

https://dspace.org/

Enhanced feature acquisition through the use of infrared imaging for human detection

Kunene, Dumisani

2020
https://hdl.handle.net/10539/30977

Downloaded from a DSpace repository. https://dspace.org/

Enhanced feature acquisition through the use of infrared imaging for human detection



UNIVERSITY OF THE
WITWATERSRAND,
JOHANNESBURG

A dissertation submitted in partial fulfilment of the requirements for the degree Master of Science.

Author: Dumisani Kunene (890669)

Supervisor: Dr. Hima Vadapalli

School of Computer Science and Applied Mathematics,
Faculty of Science,
University of the Witwatersrand, Johannesburg.

June 2020

Abstract

This thesis describes the implementation and review of human detection approaches over different spectral imaging. While significant progress on human detection has been made in the past, human detection in static images remains a challenging research problem. The performance of popular human detection systems remains inferior to the visual capability of people and animals [1, 2, 3, 4, 5, 6].

Most human detection methods are often evaluated over visible-light images. However, visible light-images can contain limited information in lowly-illuminated environments. Other complexities occur due to the possibility of random colour patterns on the image background regions and clothes of pedestrians. In most cases, the colour clutter contributes negatively to image representation methods that solely rely on edge information.

With infrared imaging, the heat radiated from objects is often uniform and independent of the colour texture, resulting in less cluttered images. This work evaluates the significance of using imaging-infrared (IIR) footage instead of visible-light images for the human detection problem. The basis of the supposition is that the choice of extracted information has a large impact on the robustness of statistical learning systems. To test this supposition, support vector machines (SVMs) and extreme learning machines (ELMs) and convolutional neural network (CNN) classifiers were trained and tested with three different datasets. The datasets consisted of the newly created infrared-based pedestrian dataset named Significance of Near Infrared Dataset (SIGNI) [7], along with the popular National Institute for Research in Computer Science and Automation (INRIA) and National ICT Australia (NICTA) pedestrian colour datasets [8, 9].

The classifiers were first trained with colour images to determine the optimal parameters that obtain high classification rates on unseen samples. Once satisfactory results were obtained, the same parameters were used for training the classifiers with infrared samples. This ensured non-biased classification comparisons over the different spectral images. The widely acclaimed histograms of oriented gradients (HOG) features were used as the human descriptor on the SVMs and ELMs and tested against the autonomously learned feature-maps on a CNNs. Therefore, this work provides more findings on the application of shallow learning and deep learning models in human detection and entails further experimental research on image processing methods and the classification of human beings in static images. The main rationale of this research is in addressing the lack of findings on the use of sufficiently large IIR training datasets for extracting better image features for human detection systems.

Initially, the hypothesis was tested by examining the classification rate of the classifiers on six relatively-small datasets of the same size and thereafter tested on larger

datasets. The SVMs obtained an average classification rate of 98.33% on the three infrared datasets, a performance gain of 0.6% than the average 97.73% that was obtained with the visual datasets. More apparent results were observed with the ELMs classifier as it achieved a gain of 8.4% with an average classification rate of 95.86% with infrared samples. The CNNs achieved the best overall classification rate than the other two classifiers, an outstanding score of 99.5% was obtained with the infrared images showing an average performance gain of 2.17% than with visual images.

Performance evaluation on larger datasets showed a similar outcome as all classifiers obtained performance gains with infrared samples. The SVMs obtained a 2.95% increase and the ELMs had a good 6.25% advantage. As on previous experiments, the CNNs scored insignificant gains of 1.66% from the relatively high classification rates than the other two classifiers.

Summing the average performance gain of each classifier on both small and larger datasets and diving it by two yields the overall performance gain of each classifier. The overall performance gain of the SVM classifier on all experiments was 1.78%, with the ELMs showing the largest gain of 7.20% and the CNNs obtaining the performance gain 1.92% respectively.

The best performing classifier (CNNs) was selected for assessing the human-detection problem over the different spectral images. The assessment was conducted by running a sliding window detector over image pyramids from natural-scene images with ground-truth information. The overall classification rate on the infrared testing scenarios was 4% higher than the average classification rate over the colour testing scenarios. Studying classification rates only when comparing classifiers can be misleading. Precision rates only highlight the accuracies of classifiers solely on images they retrieved as positive thus neglect the accuracy over the entire ground-truth data.

For instance, the classifiers that were trained with colour images (INRIA and NICTA) had higher precision rates, yet failed to retrieve a remarkably large number of positive ground-truth regions, achieving low recall rates of 0.043 and 0.077 respectively. This equates to only 5.98% of positive ground-truth boxes that were retrieved, versus 63% that was retrieved by the infrared classifier. The localisation experiments addressed an imbalanced binary classification problem, where one of the classes (negative background samples) had the overwhelming majority of the data samples. In such cases, precision rates are not a good measure because they can be easily obtained by classifiers that have a bias to the overwhelming class. Instead, the recall rate metric becomes a better measure as it shows the model's ability to find relevant samples from a testing scenario that has considerably more irrelevant samples.

The ground-truth results show that the precision and recall rates of the infrared model were both fair, unlike the visual models, where the classifiers had higher precision rates and substantially poor recall rates.

Therefore, throughout all experiments, better results were obtained with the use of infrared images than the use of visual images by all classifiers and the CNNs performed well than the two shallow learning classifiers.

Declaration

I, Dumisani Kunene (student number: 890669), declare that this research project is my own work, and all information sources have been referenced. Additionally, this work has never been submitted to any other university for any other degree.

Signed: _____

Date: _____

Acknowledgements

I would like to honour God for the wisdom and strength to complete this thesis.

To my supervisor, Dr Hima Vadapalli, thank you for the guidance, wisdom and the support you provided for this research and publications.

The author acknowledges the assistance and funding that was provided by the Council for Scientific and Industrial Research (CSIR) of South Africa. In addition, I would like to thank my colleagues from the CSIR: Rofhiwa Seletani for providing constructive feedback on the thesis and Vusi Skosana for holding me into account in completing this thesis.

Finally, I would like to extend my gratitude to my mother for the inspiration and motivation she provided in completing my studies in spite of my deteriorating health during this period. I thank my family, friends, colleagues, fellow students for their consistent support in making this thesis a reality.

Publications

Part of this work has been presented at the following conferences:

D. Kunene and H. Vadapalli, "Better feature acquisition through the use of infrared imaging for human detection systems," in *Proceedings of the South African Institute of Computer Scientists and Information Technologists*, SAICSIT '17. Bloemfontein, South Africa, 2017, pp. 21:1–21:10. [Online]. Available: http://doi.acm.org/10.1145/3129416.3129437

Influential publications:

D. Kunene, H. Vadapalli, and J. Cronje, "Indoor sign recognition for the blind," in *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists*, SAICSIT '16. Johannesburg, South Africa, 2016, pp. 19:1–19:9. [Online]. Available: http://doi.acm.org/10.1145/2987491.2987530

Other contributions:

D. Kunene, "Infrared human dataset for image classification and detection tasks", 2016, https://goo.gl/ugbevV.

Contents

Abstract i

Declaration iv

Acknowledgements v

Publications vi

Contents vii

List of Figures x

List of Tables xii

Glossary Terms xiii

1 Introduction 1
- 1.1 Problem Statement . . . 3
- 1.2 Theoretical Framework . . . 3
  - 1.2.1 Research Purpose . . . 3
    - 1.2.1.1 Research Questions . . . 4
  - 1.2.2 Hypothesis . . . 4
  - 1.2.3 Research Variables . . . 5
  - 1.2.4 Justifying the supposition . . . 5
- 1.3 Motive . . . 7
- 1.4 Objectives . . . 8
- 1.5 Contributions . . . 9
- 1.6 Thesis Outline . . . 9

2 Literature Review 12
- 2.1 Introduction . . . 12
- 2.2 Common approach to the problem . . . 12
- 2.3 Early work . . . 13
- 2.4 Histograms of Oriented Gradients . . . 14
- 2.5 Part Based Object Detection . . . 16
- 2.6 Object Recognition using Depth of Field . . . 19
- 2.7 Rotational Invariance HOGs . . . 20

2.8 Other descriptor features similar to HOGs . . . 22
2.9 Detection and Localisation . . . 22
2.10 Convolutional Neural Networks . . . 23
2.11 Background on Classifiers . . . 25
2.11.1 The Support Vector Machines . . . 25
2.11.2 Non linear SVM . . . 30
2.11.3 Neural Networks . . . 31
2.11.4 Extreme Learning Machines . . . 33
2.11.5 Convolutional Neural Networks . . . 35
2.11.6 Characterising Performance . . . 36
2.12 Summary . . . 37

3 Infrared Dataset Creation **39**
3.1 SIGNI Dataset . . . 39
3.1.1 The Camera Used . . . 40
3.2 ROI Extraction and Grouthtruth . . . 44
3.3 Ethics Clearance . . . 45
3.4 Summary . . . 46

4 Research Methodology **47**
4.1 Data Collection . . . 47
4.1.1 Data preprocessing . . . 49
4.2 HOG feature extraction . . . 49
4.3 Implementation of Classifiers . . . 52
4.3.1 SVM Classifier on HOG Features . . . 52
4.3.2 ELMs Classifier on HOG Features . . . 54
4.3.3 CNNs Classifier on HOG Features and Gray-Scale Images . . . 54
4.4 Metrics Used . . . 57
4.5 Summary . . . 58

5 Hyperparameter Selection **60**
5.1 Introduction . . . 60
5.2 SVM Classification on HOG features . . . 62
5.2.1 Introduction . . . 62
5.2.2 Selecting the Candidate Classifier . . . 63
5.2.2.1 Subtracting the Mean of HOG Features . . . 64
5.2.2.2 The effects of normalising HOG features . . . 64
5.2.2.3 Different Kernel Types . . . 66
5.2.2.4 Candidate SVM Classifier . . . 66
5.3 ELM Classification on HOG features . . . 67
5.3.1 Introduction . . . 67
5.3.2 Selecting the Classifier Candidate . . . 68
5.3.2.1 The depth of the network . . . 68
5.3.3 Activation Functions . . . 69
5.3.3.1 The effects of normalising HOG features . . . 70
5.3.3.2 The Biases of the Hidden Nodes . . . 70
5.3.4 Candidate ELM Classifier . . . 71

5.4 CNNs classification on grayscale images . . . 72
5.4.1 Introduction . . . 72
5.4.1.1 Architecture . . . 72
5.4.2 Choosing Parameters . . . 73
5.4.2.1 Feature Map Down Sampling . . . 74
5.4.2.2 Centralization of Data . . . 75
5.4.2.3 Depth of the Network . . . 75
5.4.2.4 Regularization methods: Overfitting Avoidance . . . 76
5.4.2.5 Learning rate and momentum . . . 76
5.4.3 Candidate CNN Classifier . . . 77
5.5 Summary . . . 77

6 Analysis of Results **78**
6.1 Evaluation A: Highly Skewed Datasets . . . 78
6.2 Evaluation B: Less Skewed Datasets . . . 80
6.3 Detection and Localisation Performances . . . 83
6.3.1 Localisation Experiments . . . 84
6.3.2 Localisation Results . . . 85
6.3.2.1 Ground Truth Validation flaws . . . 87
6.4 Results and Discussion . . . 87
6.5 Overal Results . . . 91

7 Conclusion **92**
7.1 Future Work . . . 93

Bibliography **102**

A Appendixes **104**
A.1 Introduction . . . 104
A.2 Localisation Tool . . . 104
A.3 Ethics Clearance . . . 112

List of Figures

1.1 The comparison of visual samples [1] (top) and infrared imaging samples (bottom). . . . 6
1.2 Applying 1D-centric edge filter on the visual (top) and IIR (bottom) samples. . . . 7
1.3 Applying the Sobel edge filter on the visual (top) and IIR (bottom) samples . . . 7

2.1 Generic steps that are involved in most sliding-window based human detection algorithms. . . . 13
2.2 The performance of the original HOG based human detector over the INRIA dataset, plotted as a DET curve [1]. . . . 16
2.3 The star structure model used by Felzenszwalb et al [28] to detect people by parts. . . . 17
2.4 The use of multiple instance learning (MIL) for evaluating the context of the entire image based on interesting sub-regions [28]. . . . 17
2.5 Range segmentation used for handling overlapping people in images [6]. The 3D points of the foreground regions are shown in (a). Then (b) is the projection of (a) onto a 2D plan. The result of applying Mean Shift Clustering for segmenting overlapping people is shown in (c). . . . 20
2.6 The results of mapping x, y cartesian coordintes to a polar coordinates (ρ, θ) plane [5]. . . . 21
2.7 The Gaussian weight distribution provides more weighting to foreground regions that are likely to contain people and less weight to the background regions of the image [5]. . . . 22
2.8 An illustration of the support vector machine model [44]. . . . 26
2.9 The margins between the hyperplanes H_1, H_0 and H_2 . . . 26
2.10 A basic example on how non-separable input points in a lower dimensional space can be mapped to a higher dimensional (in this case from 1D to 2D) space to make them separable. . . . 30
2.11 The perceptron, a basic form of a neural network [53]. . . . 32
2.12 A simple neural network structure with 3 layers [54]. . . . 33
2.13 Convolutional Neural network diagram [53]. . . . 35
2.14 Typical examples of ROC and PR curves. . . . 37

3.1 Sample large images from the SIGNI dataset. The images were captured in different environments and contain enough variance in the appearance of people. Human subjects were captured during winter and summer, in different poses and wore different clothes. . . . 41
3.2 Some of the positive samples from the SIGNI dataset. . . . 42

3.3 Some of the negative samples from the SIGNI dataset. 43
3.4 The compact thermal camera used for capturing IIR samples [64]. 44
3.5 Graphical user interface for the ROI extraction tool 45

4.1 Image classification process. 47
4.2 The neighbouring pixel values used for computing the centre pixel's gradient vector and magnitude value. 50
4.3 The horizontal gradient I_x (1st), vertical gradient I_y (2nd), magnitude of gradients M (3rd) and the cell division of the magnitude of gradients image (4th). 50
4.4 A typical example of a histogram of oriented gradients for a regular 8 pixel resolution cell. 52
4.5 Max pooling a simple image with a 2×2 patch [69]. 56

5.1 A diagram visualising the process of classifying images with the SVMs. . . 63
5.2 The mean image (left) and its HOG visualisation right (right). 64
5.3 The cross validation results for selecting the optimal sigma values. 67
5.4 Training accuracies of different activation functions of the $ELMs^{MIT}$ classifier as the number of neurons increased. 70
5.5 The CNN architecture used for human detection. 73

6.1 The precision and recall graphs of the classifiers. 82
6.2 Sample images from the SIGNI dataset. 82
6.3 Detection results of the joint SVM+CNN classifier. 83
6.4 Typical ROI's with complex partial occlusion of people that are observed by the sliding window classifier [1, 7]. 84
6.5 Some of the images that were classified as negative samples by the strict 50% overlap requirement when validating object detection ground truth information [1, 7]. 88
6.6 Heat reflections were not considered for ground-truth information, yet some were classified as positive by the classifier. 88
6.7 Some of the results from region-based CNNs detector. 89

7.1 *Cartooned* image, colour unification for better boundary edge extraction. Top left is the original image. Followed by its cartooned version (top-right). The corresponding magnitudes of gradients images are shown below it [1]. 94

A.1 The detection window position and size mapped to the ground truth resolution. 106
A.2 A typical example of the produced xml tree containing ground-truth information. 107
A.3 Cases A and B, when the detection window is above the ROI coordinate. 108
A.4 Cases C and D, are for cases when the detection window is below the ground truth ROI coordinate. 109
A.5 Cases E and G, are for cases when the detection windows is within or equal to the ground truth ROI. Then in case F, the detection window fully covers the ground truth ROI. 109
A.6 The necessary ethics clearance for creating the SIGNI dataset. 112

List of Tables

1.1 A distinction of independent and dependent variables. 5

2.1 Typical kernels functions of the SVM classifier. 31

4.1 Visual Datasets sample count. 48
4.2 IIR Dataset sample count. 48

5.1 Training, validation and testing partitions for finding the hyper-parameters of the CNNs. 61
5.2 SVM validation results with different kernels on the MIT Pedestrian dataset. 63
5.3 SVM validation results without block normalisation on HOG features. . . 65
5.4 SVM validation results with L1 block normalisation on HOG features. . . 65
5.5 SVM validation results with L2 block normalisation on HOG features. . . 65
5.6 ELMs validation results on normal HOG features from the MIT dataset. 69
5.7 ELMs validation results on standard HOG features from INRIA dataset. 69

6.1 Classification results of the SVMs over highly skewed datasets. 79
6.2 Classification results of the ELMs over highly skewed datasets. 79
6.3 Classification results of the CNNs over highly skewed datasets. 80
6.4 Classification results of SVMs on larger datasets. 81
6.5 Classification results of ELMs on larger datasets. 81
6.6 Classification results of CNNs on larger datasets. 81
6.7 Object localisation results of sliding classifiers: $Classifier^{INRIA}$, $Classifier^{NICTA}$ and $Classifier^{IIR}$ 86
6.8 Overall classification performance gains per classifier. 91

Glossary Terms

Abbreviation	**Meaning**
IIR	Imaging Infrared
HOG	Histograms of Oriented Gradients
SVM	Support Vector Machines
ANN	Artificial Neural Network
ELM	Extreme Learning Machines
CNN	Convolution Neural Networks
OpenCV	Open Computer Vision Library
SIFT	Scale Invariant Feature Transform
RIFT	Rotational Invariant Feature Transform
SURF	Speeded Up Robust Features
PCA	Principal Component Analysis
DET	Detection Error Tradeoff
ROC	Receiver Operating Characteristic
PR	Precision and Recall
FPPW	False Positives Per Window
ROI	Region of Interest
H_1	Hypothesis 1
H_2	Hypothesis 2
WASP	Wide Area Surveillance Prototype
CSIR	Council for Scientific and Industrial Research
SANDF	South African National Defence Force
SIGNI	Significance of Near Infrared Dataset
NICTA	National ICT Australia
INRIA	National Institute for Research in Computer Science and Automation (France)

Chapter 1

Introduction

Object detection is one of the fundamental goals in image processing and computer vision. The computer-vision research field focuses on methods of enabling autonomous image understanding for computers and robotics. There has been overwhelming progress within the field of object recognition. However, many computer-vision problems remain unsolved due to the computational complexities involved with most computer vision problems [1, 2]. Image understanding becomes challenging due to factors such as the varying environmental settings, cluttered backgrounds, occluded objects, varying view perspectives and the change in illumination in images. Even the best classification algorithms struggle to detect objects that are captured from varying viewing angles and blending background settings [10].

The main goal in this research field is to mimic human vision abilities by digitally finding image features that help computers to understand images better. Computers represent digital images by using n-dimensional matrices of numbers. As a result, it is extremely difficult recognising objects by just observing these numbers. Over the years, intelligent models that can produce symbolic information from digital images have been developed using linear algebra, geometry, statistics and machine learning principles.

Image features are extracted from image pixels to produce generic descriptions of objects. The features are then fed to machine learning models for the detection and classification of objects in images. Some of the common feature extraction methods include Haar Wavelets, scale-invariant feature transform (SIFT), speeded up robust features (SURF), histogram of oriented gradients (HOG) and rotation-invariant fast features (RIFF) [1, 11, 12, 13]. Extracting image features enable machine learning models to be trained much more effectively since the large pixel information is quantized to smaller vectors. Machine learning models such as linear regression, artificial neural networks (ANN),

support vector machines (SVM), kernel estimation, and decision trees are widely used in the literature for image feature detection tasks [14, 15, 16].

Human detection is a vital tool for most intelligent video surveillance systems and driver assistance systems on modern cars. Human detection is highly a complex problem due to the variation of appearance within the same class of humans. The difficulty is caused by the non-rigid nature of pedestrians. Unlike generic object detection for rigid objects, which can be detected by simply matching spatial features (e.g face detection), pedestrians can be photographed in various poses, size, shapes and view-angles. Matching algorithms have been proven unreliable for human detection due to the challenges mentioned earlier. Matching algorithms seek to detect people by matching image features from reference images that contain people [3, 17].

Luckily promising methods that can achieve high detection rates have been presented over the years [2, 5, 6, 18, 19]. Most of the work is based on the popular HOGs by Dalal et al [1]. Some of the methods use discriminative body parts for detecting people [2, 6, 20, 21]. The deformable-parts based models classify the global problem by combining the results of several local classifications. Zhao et al [22] used the bounding contours of people to identify the shapes of the deformable body parts of people and pictorial shape of the entire body. Similarly, a novel feature set that combines HOGs and local binary patterns (LBP) was presented by Wang et al [4] to handle partial occlusion in human detection. They used the locally distributed scores of the global window classification to determine the regions of occlusion.

Human detection is challenging on colour images because of the variation of colour in clothes and background. Unfortunately, the generic shape of pedestrians is not segmentable by colour because the colour patterns vary on clothes and people have different skin tones. In cases where there is low visible-light illumination, detecting people or objects becomes challenging whereas, imaging infrared (IIR) can overcome this restriction.

With infrared images, the uniform shape of pedestrians is more extractable and independent of the colours on clothes and background regions because infrared cameras only capture the heat-variation on surfaces. Infrared images are advantageous for human detection systems that operate throughout the day, where the temperatures of human subjects are likely to vary from the temperatures of background regions. Any object with temperatures above absolute zero emits infrared radiation that can be used to produce thermal images [23]. Colour cameras require illumination in dark environments, this may alert the observed subject to be aware that they are being observed. Whereas with infrared cameras, you can observe people or animals without illuminating the scenes to avoid alarming subjects.

The main objective of this thesis is to evaluate the significance of using infrared images over colour images. This entails the design and implementation of different methods for detecting humans in static images. Several classifiers were implemented to analyse the classification of humans in different spectral images. The first being the support vector machines, the second being the extreme learning machines (ELMs) based classifier and lastly, the convolutional neural networks (CNNs). Unlike the ELMs and SVMs, the CNNs were trained with raw grayscale images since CNNs can learn filters autonomously. Testing several classifiers can provide a comparison between the use of hand-engineered features versus machine-learned features.

This work requires an infrared pedestrian dataset to train the classifiers and make the necessary comparisons to the colour based datasets. Due to the difficulty of finding infrared human datasets online, a great deal of time was spent in creating a new dataset called the Significance of Near Infrared Dataset (SIGNI) manually [7]. Positive IIR samples were cropped from large images with their labels and ground-truth annotations. The term ground-truth refers to information that is constructed through prior observations for analysing the accuracy of machine learning classifiers. Further details on the preparation of the SIGNI dataset are discussed in Section 3.2. This work's contributions, problem statement and the theoretical framework are presented in the next subsections.

1.1 Problem Statement

There is still no deterministic approach for detecting people autonomously in images. Further experimental research on feature extraction and classification methods are vital. To the best of our knowledge, there is insufficient work on the application of shallow learning and deep neural networks on the human detection problem over sufficiently large IIR training-datasets. This work addresses this gap by creating a new infrared pedestrian dataset and by experimenting with different statistical learning methods on the classification and detection of humans in static images.

1.2 Theoretical Framework

1.2.1 Research Purpose

The primary purpose of this study is to evaluate whether feature acquisition can be improved through the use of infrared imaging. The work provides more findings on

the application of shallow and deep learning models in the human detection problem and provides contrasts between hand-engineered HOGs with SVMs and ELMs against machine-learned features that are optimised autonomously by CNNs. One of the contributions of this work is an infrared-based pedestrian dataset (SIGNI) that was created and published [7]. The findings of this research will attempt to answer the following research questions:

1.2.1.1 Research Questions

- Research Question (RQ_1): Over the exact number of the training and testing samples, which spectral type (between the visual datasets and the IIR dataset) results to better classification rates for shallow learning classifiers (SVMs and ELMs).
- Research Question (RQ_2): When evaluating the findings of RQ_1, how does the performance of deep learning CNNs compare to the shallow learning classifiers. Can the CNNs improve the detection rates?

The answers to the posed researched questions will help us prove or disprove the hypothesis presented in the next subsection.

1.2.2 Hypothesis

Infrared images have less noise-clutter than colour images due to the uniform temperatures on objects with uniform materials. The amount of noticeable clutter on inner foreground regions of objects and background regions is significantly reduced. The following hypotheses were formed from the basis that the reduced noise clutter has the potential to improve acquired image features and improve the autonomous classification and detection of people in images:

Hypothesis (H_1): The use of infrared data can improve the accuracy of shallow learning classifiers (SVMs and ELMs).

Hypothesis (H_2) Furthermore, the dominating deep learning techniques could also benefit from the same, which may result in better human detection performance over the use of colour images and shallow learning.

Null Hypothesis (H_0) There is no significant difference in accuracy when using infrared data over colour image data on the human classification problem.

1.2.3 Research Variables

The independent variables and dependent variables of this study are presented in Table 1.1. The following independent variables will be the input to the experiments and the outcomes will affect the dependent variables:

TABLE 1.1: A distinction of independent and dependent variables.

Independent variables:	Image datasets Image samples Set of image features The choice of classifiers Classifier parameters
Dependent variables:	Detection rate Number of true positives and false-positives Number of true negatives and false-negatives Precision and Recall Rates

1.2.4 Justifying the supposition

Several image processing methods that could potentially enhance the quality of edges were tested to reduce edge-noise clutter in images as part of the preliminary experiments of this work. The initial attempt was in enhancing continuous edges in hopes of discriminating them against edges that are not part of object contours, this attempt was discarded because sections of contour edges often blend with the background. The second attempt was smoothening images with the median filter or Gaussian smoothing filter to reduce noise. But the results were relatively poorer because smoothing grayscale images can degrade the quality of sharp edges.

Eventually, the use of infrared images yielded the desired noise reduction. The reason behind this is, the heat radiated from objects of uniform materials becomes evenly distributed across the object regardless of the visible colour patterns on the material. As a result, gradient images from infrared feeds are likely to have lesser noise from unnecessary edges within the inner shapes of objects and background regions. Consequently, contour edges can be extracted more efficiently.

The shortfall of most feature extraction methods is the inability to differentiate edges that show 2D object information (the x and y axis of the picture plane) from those that show the 3D appearance (the z axis spanning towards the capturing camera). However, this distinction is enhanced on IIR images causing richer outer shape information to be extractable. High-frequency gradients often outline the contour-edges of objects whereas the low-frequency edges mostly present the shading within the objects and 3D

appearance cues. Feature acquisition algorithms may confuse the latter as boundary edges and degrade the desired shape information that is crucial for computer vision applications.

The hypothesis H_1 states that "The use of infrared training data can improve the accuracy of shallow learning classifiers (SVMs and ELMs) over the use of colour images". Hypothesis H_2 states that "Deep learning techniques could also benefit from the same and result in better classification rates than with shallow learning classifiers".

Therefore, this work's hypothesis was formed from the basis that the reduced noise clutter has the potential to improve the perceived shape information of objects during the feature acquisition stage and consequently improve the recognition rates of traditionally used classifiers. The hypotheses can be addressed by evaluating the results from testing several deep and shallow learning classifiers with different datasets of the same size and varying data signals.

A comparison between the visual and infrared samples is shown in Figure 1.1 and their corresponding gradient images are shown in Figures 1.2 and 1.3. In Figure 1.2, the noise evidence from applying a $1D{-}Centric$ edge extraction filter is barely noticeable to the human eye while the *Sobel* edge extraction filter does highlight this noise [See Figure 1.3]. The *Sobel* filter thickens edges making them brighter and reveals how visual samples contain redundant low-frequency edges than the infrared samples, which leads to the assumption that machine learning models may recognise objects better in less cluttered scenes.



FIGURE 1.1: The comparison of visual samples [1] (top) and infrared imaging samples (bottom).

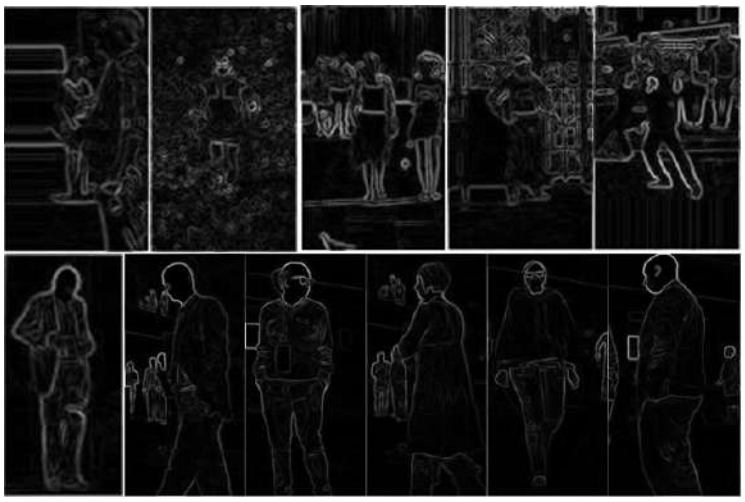


FIGURE 1.2: Applying 1D-centric edge filter on the visual (top) and IIR (bottom) samples.



FIGURE 1.3: Applying the Sobel edge filter on the visual (top) and IIR (bottom) samples .

1.3 Motive

A strategy to integrate a human detection system to a wide-area surveillance system for armour protection (WASSAP) that was developed at the *Optronic Sensor Systems* department at the CSIR was the motivation for this work. The WASSAP system provides situational awareness inside armoured military vehicles through a panoramic view in both IIR and colour footages [11, 24]. It does this by stitching matching pixels from several cameras to create a larger image. The system is aimed at providing an automated all-weather & 24-hour surveillance system for the *South African National Defence Force* (SANDF). Each IIR camera is placed with its synchronised visible-light camera, meaning each pair of visible and infrared camera point towards the same direction so that

thermal footage can be merged with visible-light footage. This is achieved by a novel autonomous camera photogrammetric calibration system [11] that uses BRISK and SIFT features. Photogrammetric calibration enables the stitching of stereo IIR footage because the common coloured checkerboards do not emit heat radiation [24]. As a 3D computational platform, the WASSAP is equipped with an image stabilisation mechanism to enhance support for background subtraction and foreground tracking tasks [25]. This is ideal for fulfilling the main objective of this project, which is to implement a system that can automatically detect nearby people and conduct the relevant research to fulfil the necessary requirements of the aforementioned degree. The following objectives were set to fulfil the work required for this thesis. The scope of this work does not cover the incorporation of this research to the WASSAP system.

1.4 Objectives

The main objective is to study the influence of using infrared images for the human detection problem on static images. To address the research questions, the following tasks must be implemented:

Procedure to probe RQ_1

1. Implement the standard HOG feature extractor.
2. Create a new infrared pedestrian dataset for supervised learning models and organise colour datasets to train classifiers.
3. Implement the two shallow learning classifiers: SVM and ELMs.
4. Optimise the parameters of the classifiers before they are tested.
5. Conduct experiments to validate and compare the performances of the candidate classifiers with the selected datasets.
6. Re-evaluate the classification performance over larger datasets.
7. Evaluate the significance of using the infrared dataset over colour datasets.

Procedure to probe RQ_2

1. Select a practical convolutional neural network architecture for human classification.
2. Optimise the parameters of the CNNs based on the datasets used.

3. Conduct experiments to test and compare the performances of the CNNs over the same datasets that were used for addressing RQ_1.

4. Compare CNN's classification results against the performance of the shallow learning classifiers (SVM and ELMs).

1.5 Contributions

The main contributions of this thesis are,

- An evaluation of the significance of using infrared images when extracting features for human detection. The evaluations are done over three different classifiers: SVMs, ELMs and the CNNs.
- Further experimental research on image processing methods and the classification of humans in static images.
- An infrared-based human dataset with annotations and ground-truth information. This dataset can be downloaded from Google Drive : `https://goo.gl/ugbevV`.

In summary, this research provides more findings on the application of shallow and deep learning models in the human detection domain. The contrasts between the hand-engineered features (HOGs) and the machine-learned features-maps is also provided.

1.6 Thesis Outline

The rest of the thesis is organised as follows:

Chapter 2: Literature Review

Related work was studied to achieve the objectives of this work and to address gaps that are found in literature. The literature review chapter includes early attempts on the human detection problem, common approaches that are followed in solving it and the recent CNN based approaches that use region proposals and bounding-box regressors to automatically localize objects in images. A clear distinction on how this work contributes to the literature is presented in the summary section of the chapter.

Chapter 3: Infrared Dataset Creation

One of the contributions of this work is the creation of an infrared human dataset named SIGNI for classification tasks. Chapter 3 discusses the rationale behind this dataset, the

approach followed when creating the dataset and the specifications of the SIGNI dataset. A subset of the positive and negative image samples from the datasets and the tools that were developed for creating the dataset are also presented.

Chapter 4: Methodology

The methodology chapter continues the theoretical framework of this research. It details the principles followed when conducting this research: the instruments that are used and how data was collected, preprocessed and analysed. The chapter also delves into the feature acquisition methods used and how the performance of the classifiers was evaluated. The software programmes used during the research are also presented. The chapter continues covering the design and implementation of the classifiers. The preliminary experiments that were conducted on each classifier to ensure that the source code works as intended and that the classifiers could be trained effectively is also discussed.

Chapter 5: Hyperparameter Selection

Machine learning models require different constraints, weights and learning rates to generalize unseen samples. This chapter describes the process for selecting the hyperparameters of the candidate classifiers for this work. Cross-validation was applied to the two shallow learning models (SVM and ELM). The chapter also describes the classification of grayscale images with CNNs.

Chapter 6: Analysis of Results

This chapter analyses the overall performance of the classifiers. Initially, the experiments were conducted with highly skewed smaller datasets of the same size and later on carried with less skewed larger datasets. The last section of this chapter discusses the localisation experiments on human detection, where the goal is to localise the position of people in relatively large images. The human detection testing scenarios consist of over a hundred annotated large images.

Chapter 7: Conclusion

This chapter concludes the findings of this work, where the advantage of using infrared imaging on the human detection problem is declared to have the potential to improve the detection results. The benefits were apparent when training the CNNs, as they converged quicker on the infrared datasets and produced smoother training loss functions. The supposition was shown to hold for all classification test cases and all three classifiers. The localisation experiments showed favourable results with the infrared testing scenarios. The results from the classifiers were scrutinized further by analysing their precision and recall rates. Future work that aims at reducing the amount of noise on colour images is

also proposed to provide a level pegging challenge to the performance of classifiers that were trained with infrared samples in this work.

Chapter 2

Literature Review

2.1 Introduction

This chapter presents a study of related work from the literature. The common approach for finding people in static images is discussed in Section 2.2. The historical attempts that led to the recent state of the art methods in human detection are covered to provide context on how the rationale behind this work was formed [Section 2.3]. A background study and the common drawbacks of HOG features and the use of sliding windows and the gaps found in the literature are also discussed [Section 2.4]. The recent CNN based approaches that use region proposals and bounding-box regressors to automatically localize objects in images are also discussed in this chapter [Section 2.10]. The chapter concludes by revealing how this work intends to address the gaps found in the literature.

2.2 Common approach to the problem

There is immense background literature on object detection, existing computer vision systems can recognise faces, vehicles and household objects with recognition rates that are above 70% [1, 26, 27, 28]. The focus of this study is on human detection literature. The vast majority of human detection solutions appear to follow a similar approach in solving the human detection problem [1, 2, 3, 5, 6]. Given an image, information on the presence of a visible human in the image is unknown to the system. The coordinates and scale of the rectangular region containing the person are also unknown. Therefore, the fundamental task for most human detection methods is to locate the coordinates and scale of the region that contains a person. Prior to the recent convolutional neural networks with region proposal networks (RPN), this was generally achieved by sliding a descriptor window over a dense scale-pyramid of the image. At every position of the

descriptor window, image features are extracted and classified as either true when the classifier predicts the presence of a human or false contrarily.

Recent methods can scan through images much quicker by using weaker classifiers to formulate a hypothesis of probable regions of interest [2, 6]. Once a hypothesis is formed, a strong binary classifier is then applied to conclude the final detection result. A generic flow-chart model on the steps required for most sliding window-based human detection algorithms is shown in Figure 2.1. To cater for scale invariance, the sliding window detector is often passed onto an image scale-pyramid, where the image resolution is scaled down until the resolution of the image equals the size of the descriptor window.

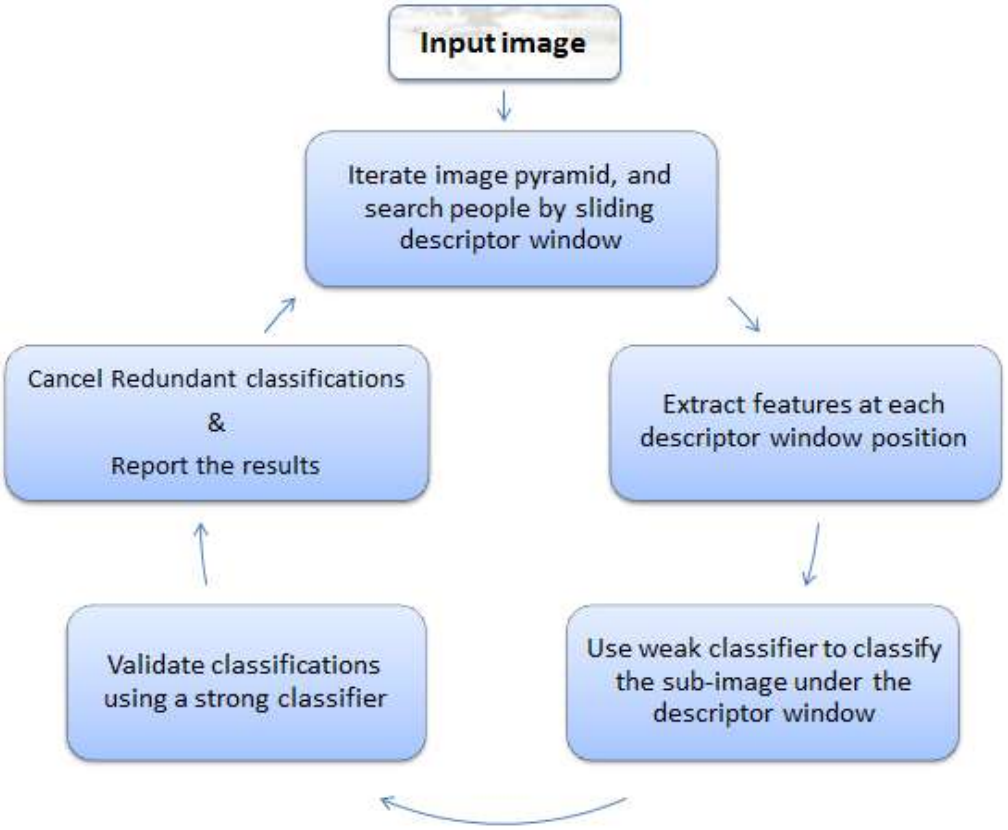


FIGURE 2.1: Generic steps that are involved in most sliding-window based human detection algorithms.

2.3 Early work

Early work tried addressing this problem by making direct comparisons of unobserved images to reference image features, however, the results were poor [3, 17, 29]. Broggi et al [17] used edge features to detect both the head and shoulders of people. They compared the extracted edges of people to a template dataset. This approach was proven less

robust because the explicit use of edge information alone becomes prone to background noise clutter, making effective edge-noise reduction methods attractive for this work.

In 1997, Oren et al [18] were the first to present a non-background-subtraction method for full-body human detection. They used Haar-wavelets features for representing the basic topology of the human body and an SVM classifier for identifying features. The SVM classifier was trained with 128x64 resolution images that either contained pedestrians at the centre of the image or just background information. Although the positive samples were explicitly defined, the negative samples consisted of random natural scenes that did not contain pedestrians. This leads to a common problem for most binary classifiers, to learn features of a negative class that cannot be explicitly defined as a single distinguishable-object causing false-positive classifications to be common.

Oren et al [18] used boot-strapping to overcome the unknown-class problem, where an SVM classifier was initially tested on a large set of negative images to identify and append false positives to the negative training set. This approach improves classification rates on specific conditions and does not help much on random unseen data that is not part of the dataset.

Haar features became popular in the past for detecting faces and other rigid objects successfully. They represent the change of intensity over neighbouring regions. When equipped with integral images, Haar features are quicker and easier to evaluate because they require basic algebraic operations. This makes them an ideal option for real-time detection systems. Unfortunately, Haar features have been proven to be less reliable on non-rigid objects like people compared to HOG based features discussed in the next section [3].

2.4 Histograms of Oriented Gradients

In 2005, Dalal et al [1] compared various types of existing digital-image features that were used for human detection and presented a breakthrough method that provided reliable human detection capabilities. This novel feature extraction method was named histograms of oriented gradients (HOG). At that time, HOG features were the most robust features for human detection. The idea behind HOGs came from the basis that the basic shape of objects can be characterised with the local distribution of intensity gradients or edge directions without the precise knowledge of their positions.

The HOG feature extraction method first applies an edge extraction filter to an input image to produce a gradient-magnitude image. This image is divided into 8×8 cells and a histogram of magnitudes is created for each cell. The frequencies of magnitudes are

populated into histograms to represent 64 pixels from each cell with significantly smaller vectors of nine values. Dalal et al. [1] experimented with various sizes of histograms but obtained the best results with $9 - bin$ histograms. The combined histogram entries of the entire image form the representation of the object features. Since gradient strengths vary over image regions due to the contrast variation, 4×4 cell-blocks were normalized to improve the accuracy of the system. During the normalisation process, blocks were overlapped with step sizes equal to a single cell to improve the performance. Two different types of block geometries were tested: the rectangular HOGs (R-HOGS) and the circular HOGS (C-HOGS). Slight performance gains were obtained with the C-HOGs.

The advantage of HOG features is on their ability to capture shape information regardless of slight photographic and geometric transformation of objects. The HOGs apply coarse spatial samples, fine orientation sampling and the photographic normalisation, this allows robust human detection when human-body segments change in appearance and pose. The HOGs also withstand complexities from slight changes in pose and successfully work as long as the overall body maintains an upright orientation.

Dalal et al selected a linear SVM classifier for shorter training times and simplicity of use. The classifier was initially trained with the existing MIT pedestrian dataset with 509 training and 200 testing samples [1]. This dataset contained perfectly upright people with their left-to-right reflections making HOG features to perform exceptionally well than existing datasets. To generate hard testing criteria, they produced a significantly more challenging dataset named the INRIA Person Dataset [8] with 1805 images of cropped people. The training set consisted of 1239 positive samples (doubled with horizontal reflections to 2478) and 12180 negative samples cropped from person-free background images. The testing set had 566 images that were also doubled with horizontal reflections to 1132 positive samples.

Dalal et al tested both RGB and grayscale images and reported about 1.5% performance gains when using RGB images. For the RGB colour images, they calculated gradient images for each channel and selected a channel with the highest norm on the gradient vectors. The detection error trade-off (DET) curve was used for performance evaluation at 10^{-4} miss-rate. The DET outlines small probability differences between highly competitive classifiers. The results were compared to the existing features sets: *Generalized Haar Wavelets*, *PCA-SIFT* and *Shape Contexts*. The DET curve Figure 2.2 shows the performance comparisons of feature sets that Dalal et al presented, where HOG features achieved lower miss-rates and outperformed the existing features sets when equipped with the same SVM classifier.

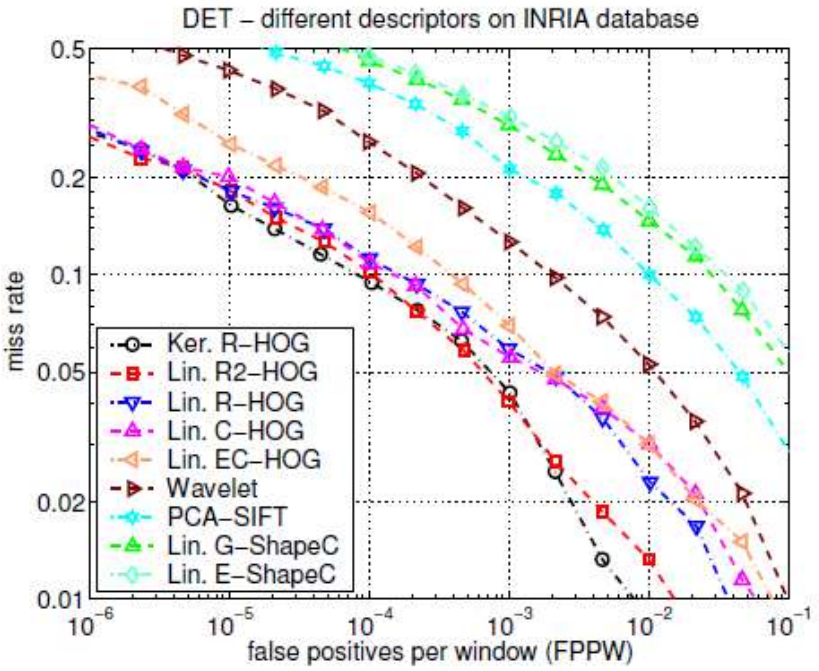


FIGURE 2.2: The performance of the original HOG based human detector over the INRIA dataset, plotted as a DET curve [1].

2.5 Part Based Object Detection

In 2010, Felzenszwalb et al [28] presented a system that improves the standard HOG classifier [1]. This novel approach used a set of visual grammars to model the appearance of objects. Objects are visualised with hierarchical structures that are formed by deformable parts of the object. The system can recognise objects without relying on their consistent overall pictorial structure, making it more robust to varying deformations of objects. The visual grammar-models cater for instances when the object's separable parts are placed at varying positions relative to the root location of the object. A star object structure was used to detect the body shape of humans, where a human's head was assigned as the root position, followed by the two blocks that separate the person's torso and the arms and two more blocks that capture the person's upper legs and lower legs [Figure 2.3]. Visual grammar models for people are difficult to train because some of the rich information is treated as latent information. For example, most human detection datasets only include the labels and coordinates of the bounding boxes that contain the entire object, but do not provide the labels and coordinates of deformable parts that make up the object. As a result, Felzenszwalb et al [2] treated the non-labelled parts as hidden information.

A multi-instance learning (MIL) SVM was used for evaluating the context of the entire image based on the contents of multiple non-indexed subregions. The entire image is

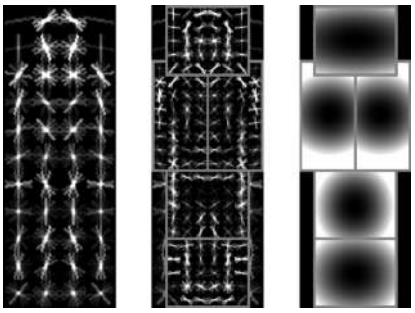


FIGURE 2.3: The star structure model used by Felzenszwalb et al [28] to detect people by parts.

viewed as a set of indexed local sub-regions without relying on manually annotated sub-regions of the image. Thus the context of the image is described using the results of multiple searching detectors on the image. These detectors produced indexes to the sub-regions of interests. This limits deep classifications to only good hypothesised regions and less processing on regions that are less likely to contain the targeted objects [Figure 2.4].

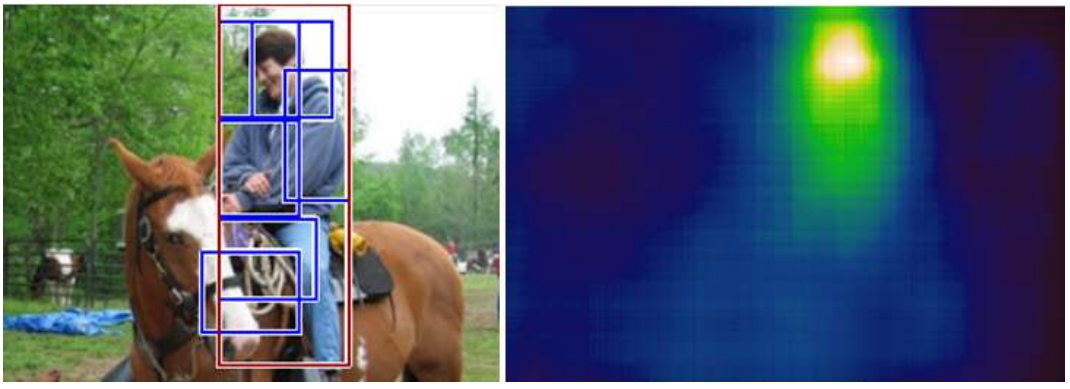


FIGURE 2.4: The use of multiple instance learning (MIL) for evaluating the context of the entire image based on interesting sub-regions [28].

This cascade of classifiers analyses the global appearance of objects using the contributions from separable part detectors. Meaning, the classifiers use dynamic programming to reuse the results of initial weak classifiers when computing the score of the richer ones. A single non-deformable but descriptive part of the objects is pre-selected as the root position, hence the head is selected as the root position for pedestrians. Root positions determine the extent to which deformable object parts are displaced from their natural position. The relative position of objects from the root position determines the significant contribution of each weak classifier to the final score.

Felzenszwalb et al [2] showed how 1×36 dimensional vector of the HOG feature block

could be reduced without significantly affecting the detection rates. The principal component analysis (PCA) of the training data was taken for reducing the dimension of the appearance-vectors. It was discovered that most of the HOG feature-energy lies within the projection of the top 13 eigenvectors. They obtained competitive detection results when applying 13-dimensional feature vectors instead of the normal 36-dimensional vectors.

The performance of the system was evaluated using the PASCAL VOC 2006, 2007 and 2008 challenges and protocols [30]. These were widely used benchmarking datasets that offer ground-truth information and the datasets are widely known for having difficult testing criteria. The goal is to detect all the bounding boxes that contain a person where classifications are considered true if the detection window overlaps at least 50% of the ground-truth bounding box and false otherwise. Multiple detections that overlap existing detections are eliminated.

The traditional method for evaluating the overall performance of computer vision systems is to examine the precision-recall statistics of the system's classification results. The precision determines how accurate and useful the system is and the recall determines how complete are the classifications compared to the total number of positive samples in the testing set. Felzenszwalb et al's system obtained the best average precision (AP) score for 9 out of 20 categories in the 2006 and 2007 PASCAL challenges. For human detection, they obtained the second-best score on the published results of the PASCAL Challenge [2]. Furthermore, they tested various versions of their system by testing different pictorial grammar structures and established that the use of classification by parts and effective bounding box prediction has a great potential to improve the detection accuracy of generic objects.

It is clear that Felzenszwalb et al [2] made significant contributions to HOG based human detection systems since the work by Dalal et al [1]. The ability to detect various objects from different classes with partial occlusion and deformation was the highlight of their work. This technique brings a step further in making machines that can recognise objects despite latent pictorial information.

This approach heavily relies on edge information to characterise the shapes of objects by parts, therefore the idea to enhance the quality of extracted edges through the use of IIR images may also improve the accuracy of part based detectors. The methods discussed thus far use two dimensional (2D) information to detect people. That is, only the visual information is utilised without any depth information. There has been increasing interest in studying ways of acknowledging depth-of-field information for more robust computer vision applications. Some attempts use stereo cameras or a combination of depth-of-field sensors with visual optics to improve the recognition accuracy of objects.

2.6 Object Recognition using Depth of Field

The appearance of most objects in real scenes changes with respect to the view perspective. Hence generic object detection methods struggle discovering objects from 2D images. Various methods have been established for considering the three-dimensional appearance of objects in scenes [31, 32, 33]. Toru et al [6] used stereo cameras for extracting the foreground regions of objects in scenes. With the depth information, they could evaluate the size of the required detection window without using scale-down image pyramids and sliding windows. This saves time by restricting image regions that need to be scanned and reduces chances for false-positive classifications.

Based on Wu et al's [34] work on using stereo cameras for segmenting the faces of people for fast detection, Toru et al [6] used the same approach to segment people from the background scene. The focal length and position of two cameras are used to estimate the distance of people from the cameras. This segmentation is done using stereo subtraction and background subtraction, where the disparities of pixels from the two cameras are matched to find foreground objects. Initially, background images are taken as reference points for subtracting the background on images taken at later stages. The background reference images are updated periodically to handle the changes in daytime illumination. Considering that the cameras are placed at different positions and viewing angles, disparities are also apparent for pixels of stationery foreground objects.

To handle cases when people overlap, Wu et al [34] projected the pixels of foreground objects to a 3D space in order to separate different people in scenes automatically. The points are further projected from the x, y, z scene to a 2D plane to provide an upright vertical view of the scene. This plane is then divided into 5×5 blocks. A histogram was computed for each block to threshold histograms that contain few pixels [Figure 2.5]. Mean Shift Clustering (MSC) was used to estimate the number of foreground objects on the scene. The MSC clusters neighbouring pixels on the vertical view so that occluded and overlapping people are apparent and treated separately [Figure 2.5] . Only the extracted regions were used for the final stage classification of Joint HOG features based on [35]. Joint HOG features are known to consider the co-occurrence of symmetric shapes. The weak and strong classifiers were trained with the large NICTA [9] person dataset. When tested against the reference system [35], Wu et al [34] obtained remarkable performance gains due to the consideration of partial occlusion.

The reviewed systems so far only work on upright people. Hence their corresponding descriptor windows were always vertical boxes. Due to the limitations of HOG features, rotational invariance is still yet to be fully realised on human detection. Section 2.7 discusses one of rotational invariance attempts for human detection [5, 13].

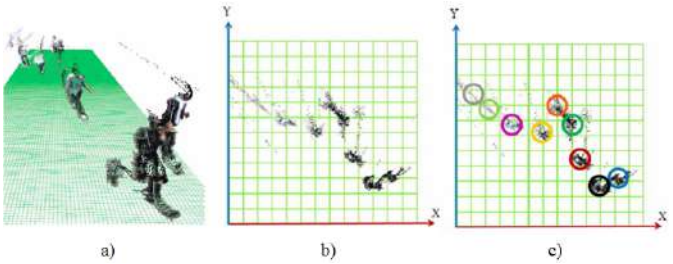


FIGURE 2.5: Range segmentation used for handling overlapping people in images [6]. The 3D points of the foreground regions are shown in (a). Then (b) is the projection of (a) onto a 2D plan. The result of applying Mean Shift Clustering for segmenting overlapping people is shown in (c).

2.7 Rotational Invariance HOGs

The common drawback for HOG based human detection systems is the lack of outright support the rotational invariant detection. The standard HOG based detector is trained using a vertical rectangle bounding box of 128×64. The positive samples only contain HOG features of upright people. This makes it impossible to cater for different camera tilt and poses of people. People are likely to be photographed from slightly skewed angles due to several factors: the slope of the ground, camera lenses and position.

Wang Li et al [5] proposed rotational invariance support for HOG based human detection systems, through mapping pixels from cartesian coordinates to polar coordinates. Their algorithm first transforms the rotation of the object into circular sectors and thereafter mitigates the effects caused by the rotation using a reverse mapping method. This is based on a hypothesised human angle that is obtained with a double-scale estimation method. Therefore, the degree of a person's tilt is first evaluated before applying the pre-processing step that corrects this tilt to a vertical pose. After this process, polar HOG features are then extracted to train an SVM classifier.

The polar-coordinates-mapping starts with a $w \times w$ square region of interest [Figure 2.6]. From this ROI, a circular region with a diameter equal to the width of the square is extracted. This circular region is then divided into sectors of radius $r = w$ and 30° angles. This totals into 12 sectors that cover the entire 360° circular region. Each sector is then decided into 3 sub-sectors of equal radiuses of $r/3$. For every point (x, y) in the cartesian plane, a cyclic translation is applied to map it to its polar coordinate position (ρ, θ), where the point (x_c, y_c) is the location of the polar origin in the cartesian plane, $\rho = \sqrt{(x - x_c)^2 + (y - y_c)^2}$ and $\theta = \arctan(y - y_c/x - x_c)$. The cyclic translation maps the cartesian circular detection region into a rectangle that eliminates the rotation of a person by representing it with a horizontal position. A problem arises due to smaller inner sub-sectors, as the pixels from the circular region cannot be enough to fill the new

rectangular space. The author mitigated this problem by using bilinear interpolation to generate values for the missing pixels based on their neighbouring pixels.

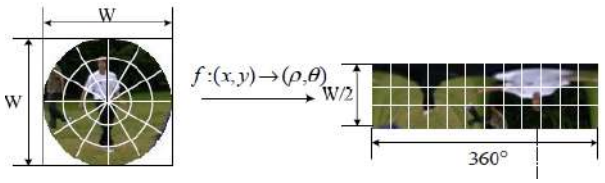


FIGURE 2.6: The results of mapping x, y cartesian coordintes to a polar coordinates (ρ, θ) plane [5].

To increase the accuracy of the algorithm, Li Chengdong et al [5] proposed a double-scale direction estimation algorithm to reinforce the statistical data from the oriented-gradients obtained from ROIs. They discovered that one of local maxima points on the histogram of all oriented-gradients represents the direction of the person from head to foot. This occurs if and only if the person is central to the round detection window. To find the correct candidate direction, they iterated across all local maxima points and stored them in an array of candidate directions. These were used for producing a Gaussian weight distribution for the ROI windows to suppress the background information, a trait that is similar to using infrared imaging for suppressing background edge information. Nevertheless, this approach from Li Chengdong et al [5] motivated the main objective of this thesis since they improved the accuracy and quality of the extracted HOG features.

The candidate directions were also used to correct the rotation of people [Figure 2.7]. So given one of the candidate direction angles, a Gaussian weight distribution is given by this function:

$$e^{-\frac{\left(\frac{x\cos(\vartheta)-y\cos(\vartheta)}{d}\right)^2+\left(\frac{(y\cos(\vartheta)-x\cos(\vartheta)}{2d}\right)^2}{4\sigma^2}}, \tag{2.1}$$

where x, y are the coordinates of the cartesian plane, d is the axis length of the square window, ϑ is the candidate direction angle used and σ is the rate of decay of the weights. The figure from Li Chengdong et al's [5] publication, shows the weight distribution of a candidate direction of 45° and its resultant $\rho - \theta$ map [Figure 2.7]. Polar-HOG features are extracted using the weights of this map and the feature extraction process is similar to the methods used on standard HOG features. At this stage, the images are ready for training SVMs for classification. The underlying success of this method is in its ability to cancel the rotation of any person as long as their groin is placed near the origin of the circular extracted region.

Due to the time scale of this research, applying rotational invariant HOG features will not be consider. But the reported results on the significance of using IIR images on upright HOG features will strongly apply to polar HOG features since they also depend on edge information from gradient images.



FIGURE 2.7: The Gaussian weight distribution provides more weighting to foreground regions that are likely to contain people and less weight to the background regions of the image [5].

2.8 Other descriptor features similar to HOGs

Since humans have identical structural body shapes, Yang et al [36] presented a novel feature set that was inspired by the HOG features to compute *Histograms of Silhouette Direction Code* (HSD) for human detection. The features differed from [1] because they collected directional code histograms instead of oriented edge gradients. The directional codes seem to further quantify the features into smaller value sets than the actual angles used in HOGs. This could be one of the reasons why they achieved better results with HSD features than HOG features. Furthermore, they proposed a background subtraction method that combines colour and intensity differences to extract the features of moving people more accurately. Similarly, Mitsui et al [35] combined joint HOG features to improve the detection accuracy. Ada-Boosting was used to combine multiple HOG features with spatiotemporal features of moving pedestrians. These are some of the methods that could benefit from the reduction of edge-noise clutter in images.

2.9 Detection and Localisation

Finding the exact location that contains a person in large images is challenging. After training a classifier that can predict the presence of people in images, most researchers often rely on sliding windows to find people on large images. As seen with standard SVMs on HOGs detectors, sliding windows are extremely time-consuming, especially on single-threaded processing units. The problem of finding a region of interest (ROI) through shifting a sliding classifier is bounded by $\bigodot(m \times n \times \frac{m \times n}{S_w \times S_h} \times L)$ complexity, where n and m are the width and height of the image, S_w and S_h are the width and

height of the sliding window and L is the number of times the original image is scaled down to cater for the scale invariances of people.

While background subtraction methods can segment moving targets from a sequence of images much faster than sliding windows, but they only work well with video footage that is captured from a stationary platform [37]. Given a sequence of images, background subtraction methods analyse a group of neighbouring pixels that changed from the previous images. While this can be efficient for detecting moving people, useful autonomous applications should detect both stationary and moving targets. For human detection, further operations are required to recognise background-subtracted objects because moving objects can be anything in natural scenes. This is how the problem of detecting people in *static images* arises. Therefore, this research is still applicable even if background subtraction can be employed to quickly find moving objects.

Sheik et al. [38] presented an effective background subtraction method for free moving cameras. The authors leveraged the fact that all trajectories corresponding to static areas in the scene lie in a three-dimensional subspace to discriminate between background and foreground regions in the scene.

Recently, Toro Obokuta et al. [6] proposed a fast human detection method that uses stereo triangulation for searching foreground objects at run-time. Depth of field information was computed from stereo cameras, to facilitate the discrimination of foreground objects from the background setting. Therefore, the number of image regions to scan is reduced and the size of the detection window is obtainable without resizing the image into a dense scale-pyramid. This approach gradually reduces the processing time and the possibility of false-positive classifications. In addition to the benefits brought by the depth information, they were able to apply a computationally demanding human detection algorithm that classifies segmented local features. Implementing the aforementioned background subtraction and stereo triangulation methods is beyond the scope of this work, the methods were discussed in the interest of related work.

2.10 Convolutional Neural Networks

The CNNs have been highly successful in image recognition tasks during the past few years [26]. Krizhevsky [26] showed the power of CNNs with a 1000 multi-class recognition project, where CNNs were trained to classify a large number of different objects ranging from household objects, vehicles, different animals etc.

Although SVMs are commonly used for human detection research, Fukui et al [39] successfully applied CNNs for pedestrian detection in 2015. The authors improved the accuracy and generalisation of standard CNNs by implementing a Random Dropout method during the training process, and an Ensemble Inference Network (EIN) for the classification process. Fukui obtained a performance gain of 10.5% than conventional CNNs. Fukui used the Caltech pedestrian dataset [40] to train the system and augmented the number of positive samples from the original 4000 samples to 101 000 through shifting, rotation, mirroring and scaling.

The dominance of CNNs can be attributed to the fact that the convolutional layers reduce the traditional overfitting requirements of most classifiers and allow objects to be recognised regardless of slight tilt and transformation. With a further bounding box regression step, the CNNs can predict both the bounding boxes of the detected objects and their classes. Region-based CNNs like the regions with convolutional neural networks (R-CNNs), the single-shot detector (SSD) and you only look once (YOLO) models can automatically predict the bounding boxes of the detected objects [41]. This eliminates the need for sliding windows and other alternative localisation methods.

Due to above evidence of effective object-recognition by CNNs, a CNNs classifier will be trained with raw grayscale images instead of HOG features to obtain relative performance evaluation against two shallow learning classifiers (SVMs and ELMs) in the current work. The two shallow learning classifiers will be trained with the standard HOG features from [1]. The datasets used in this work are not suitable to train a CNN with region-proposal layers, hence a standard CNN classifier was used instead, however, a pre-trained region-based CNN detector will be showcased to show contrast from sliding-window detectors.

Marcus [42] argues that deep learning models should be supplemented with other techniques to reach artificial general intelligence. Hence, the current research studies whether better feature acquisition (with reduced noise) aids the characterisation of non-rigid shapes in images by classifiers.

The CNNs also struggle to find relatively small objects in images and people wearing clothes that blend with the background [39, 42]. Moreover, even with transfer learning, large datasets are required to successfully train CNNs. CNN architectures may have millions of parameters, the performance of CNNs is dependent on the method in which initial parameters are tuned, therefore the amount of work required to train them is computationally expensive.

2.11 Background on Classifiers

This section presents a concise explanation of the different classifiers used in this research. The theories behind the classifiers used in this research are vital for understanding how classifiers can be trained to discriminate structures that symbolise objects in images. Initially, a background on support vector machines (SVMs) is provided in Section 2.11.1, this is followed by the background of extreme learning machines (ELMs) in Section 2.11.4 and convolutional neural networks (CNNs) in Section 2.11.5. Considering that neural networks (NNs) form the fundamental behaviour of ELMs and CNNs, a summarised discussion of feedforward neural networks is also provided. Section 2.11.6 discusses how the performance of binary classifiers can be characterised.

2.11.1 The Support Vector Machines

The SVMs are supervised learning models that are widely used for binary classifications and regression analysis. The original SVM was introduced by Vladimir Vapnik and Alexey Chervonenkis in 1963 [14]. The original SVMs used linear separation planes for binary classification. Modern versions now support non linear decision functions for classification. These were first introduced in the early 1990s by Vladimir Vapnik, Bernhard Boser and Isabelle M. Guyon [43].

Linear SVMs map labelled input examples in linear vector form as points onto a higher dimensional space Z. Mapping points into higher dimensional spaces is an attempt to make them separable by their class labels. Therefore, a support vector machine constructs a hyperplane that expands the margin between distinctively labelled points.

The hyperplane is simply a linear decision function that projects a maximum margin between the two data classes using the so-called support vectors [Figure 2.8]. This means support vectors are data points closest to the hyperplane (the decision surface). Consequently, these are the most difficult points to classify as they form the margin between the hyperplane and the nearest data points on both positive and negative samples sides.

In Figure 2.9, the red squares are the positive samples and the blue circles are negative. The H_0 and H_1 vectors are the marginal hyperplanes formed by the support vectors. The vector H_0 between the marginal hyperplanes is called the optimal hyperplane. The length of the margin between the two classes and the optimal hyperplane is determined by the number of input training vectors (the support vectors). These support vectors are trained with the training samples to optimise the maximum distance from the hyperplane to the nearest data points in both classification groups, resulting in an optimal

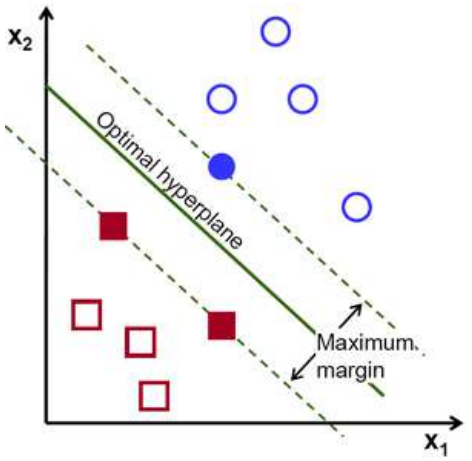


FIGURE 2.8: An illustration of the support vector machine model [44].

hyperplane. The larger the margin, the lower the generalisation error becomes. A formal definition of linear SVMs is described below, where the hyperplanes and their corresponding optimisation algorithm are defined.

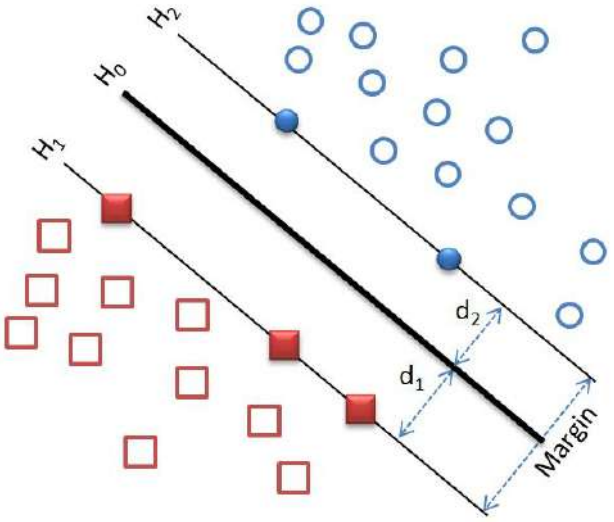


FIGURE 2.9: The margins between the hyperplanes H_1, H_0 and H_2 .

Given a labelled training set T equal to $\langle (y_1, x_1), (y_2, x_2), ...(y_n, x_n) \rangle$ that consists of pairs (y_i, x_i), where y_i is the class label (that is either 1 or -1) of the corresponding ith input example x_i, an SVM classifier can be trained by finding a hyperplane H_0 with a maximum margin that separates all examples x_j with $y_j = 1$ from all examples x_k with $y_k = -1$, where $j \neq k$.

The dataset T is proven to be linearly separable if there exists a vector $\tilde{\mathbf{w}}$ and a scaler b such that the following inequalities are true for all samples in the training set:

$$\vec{w}^T\vec{x}_i + b \quad \geq +1 \quad when \quad y = +1, \tag{2.2}$$

$$\vec{w}^T\vec{x}_i + b \quad \leq -1 \quad when \quad y = -1. \tag{2.3}$$

The two inequalities can be combined to $y_i(\vec{w}^T\vec{x}_i + b) \geq 1$ *for all* $1 \leq i \leq n$. The vectors H_1 and H_2 represent the boundary margins formed by the support vectors [Figure 2.9]. Between the two boundaries, lies the unique vector H_0 which maximises the margin between H_1 and H_2 to divide the different training samples. This vector is called the optimal hyperplane, it determines the direction of $\vec{w}/||\vec{w}||$ and is defined as:

$$\vec{w}^T\vec{x}_i + b \quad = 0 \quad for\ 1 \leq i \leq n. \tag{2.4}$$

The distance between the optimal hyperplane and the nearest example x_i from each group (negative and positive samples) has to be maximised. Therefore, the problem at hand is to maximise the margin such that $d_1 + d_2$ is maximal [Figure 2.9]. This is the sum of the distance between the optimal hyperplane and the positive sample's margin d_1 and the distance from the optimal hyperplane to the negative sample's margin d_2. Recall that by using the projection of any vector from the point $P(x_0, y_0)$ to the line $ax + by + c$, the distance from point to the line can be computed as:

$$\frac{|ax_0 + bx_0 + c|}{|a^2 + b^2|} \quad a,b,c\ are\ constants, \tag{2.5}$$

Therefore the distant from the positive support vectors x_i to the optimal hyperplane is

$$= \frac{|\vec{w}\vec{x} + b|}{||\vec{w}||} \quad , \tag{2.6}$$

$$= \frac{|1|}{||\vec{w}||} \qquad since\ \vec{w}\vec{x} + b \geq 1. \tag{2.7}$$

This follows that the distance from the negative support vector hyperplane to the optimal hyperplane is also $\frac{|1|}{||\vec{w}||}$. Thus the combined distance between the negative support-vector-hyperplane and the positive support-vector-hyperplane is $\frac{|2|}{||\vec{w}||} = \frac{2}{\sqrt{w^2}} = \frac{2}{\sqrt{\vec{w}\cdot\vec{w}}}$. Therefore in order to maximise the distance, we must minimise $\vec{w}\cdot\vec{w}$ such that no input

points lie between the marginal hyperplanes. This becomes a constrained optimisation problem for a quadratic function where the goal is to:

minimise $(\vec{w})^2$ subject to the inequality constraint $y_i(\vec{w}^T\vec{x_i} + b) \geq 1 \quad$ for all $1 \leq i \leq n$.

Thus if we let $f(x) = (\vec{w})^2$ and $g(x) = y_i(\vec{w}^T\vec{x_i} + b) - 1$, then the problem can be solved using Lagrange Multipliers. To do so, both $f(x)$ and $g(x)$ should have continuous first derivatives and a Lagrange multiplier variable λ is denoted to define the following Lagrange function:

$$\begin{aligned} \mathscr{L}(x, \lambda) &= f(x) - \sum_{i}^{n} \lambda_i g(x), && (2.8) \\ &= (\vec{w})^2 - \sum_{i}^{n} \lambda_i [y_i(\vec{w}^T\vec{x_i} + b) - 1], && (2.9) \\ &\approx \frac{1}{2}(\vec{w})^2 - \sum_{i}^{n} \lambda_i [y_i(\vec{w}^T\vec{x_i} + b) - 1]. && (2.10) \end{aligned}$$

where $\lambda = \langle \lambda_1, \lambda_1, , ..., \lambda_n \rangle$ is a vector of non negative Lagrange Multipliers and corresponds to the i'th constraint. The half in $\frac{1}{2}(\vec{w})^2$ simplifies the differentiation of $\mathscr{L}(x, \lambda)$ with respect to $\vec{w}$. It is proven that the solution to this optimisation problem is to find the saddle point of the Lagrangian for the $2n + 1$ dimensional space of λ, $\vec{w}$ and b, where the minimum is evaluated with respect to $\vec{w}$ and b. The saddle point is where the derivatives of $\mathscr{L}(x, \lambda)$ are equal to zero:

$$\begin{aligned} \frac{\partial \mathscr{L}}{\partial w} &= \vec{w} - \sum_{i}^{n} \lambda_i y_i \vec{x_i} = 0, && (2.11) \\ \frac{\partial \mathscr{L}}{\partial b} &= \sum_{i}^{n} \lambda_i y_i = 0. && (2.12) \end{aligned}$$

This implies that, the minimum is where $\vec{w} = \sum_i^n \lambda_i y_i \vec{x_i}$. We have shown that the vector of weights $\vec{w}$ is a linear combination of λ, $\vec{x}$ and the labels y_i. To solve for the values of λ we substitute the determined equation for w into the equation (3.9) to eliminate the dependency of $\vec{w}$. And also substitute b for $\sum_i^n \lambda_i y_i$. Thus equation (3.9) becomes:

$$\mathscr{L} = \frac{1}{2}(\sum_{i}^{n} \lambda_i y_i \vec{x_i})^2 - \sum_{i}^{n} \lambda_i [y_i((\sum_{j}^{n} \lambda_j y_j \vec{x_j})^T \vec{x_i} + (\sum_{i}^{n} \lambda_i y_i)) - 1]. \quad (2.13)$$

From this we can derive the Dual Lagrangian Problem which allows us to find the solution of equation (3.13) by just computing the inner products of x_i and x_j. This method is also useful for computing non-linearly separable classification classes. The Dual Problem becomes:

$$\mathscr{L}_D = \sum_{i}^{n} \lambda_i - \frac{1}{2} \sum_{i}^{n} \sum_{j}^{n} \lambda_i \lambda_j y_i y_j (x_i x_j), \qquad for\ i \neq j. \tag{2.14}$$

From this, we can solve for λ to find the weights $\vec{w}$ by differentiating the Dual Lagrangian Problem with respect to λ and setting it to zero for the saddle points [Equation 2.15]. It is known that the λs become 0 for all the λs that correspond to non-support vectors and non zero constants C to all the λs that correspond to the support vectors. Thus:

$$\lambda_i = \left\{ \begin{array}{ll} 0 & if\ \vec{x}_i\ is\ not\ a\ support\ vector \\ 0 \leq c \leq C & if\ \vec{x}_i\ is\ a\ support\ vector \end{array} \right\} \quad for\ 0 \leq \lambda_i \leq C\ . \tag{2.15}$$

The weights of $\vec{w}$ that ensure the maximum margin between the separating hyperplanes can now be calculated using:

$$\vec{w} = \sum_{i}^{n} \lambda_i y_i \vec{x_i} \tag{2.16}$$

Finally, after training the support vector machine and deriving the Lagrange multipliers to find the values for the $\vec{w}$, an arbitrary input value x_a can be classified by evaluating the sign of the function:

$$SVM(x_a) = sign(\vec{w} x_a^T + b), \tag{2.17}$$

$$= (\sum_{i}^{n} (\lambda_i y_i \tilde{\mathbf{x}}_\mathbf{i} \cdot \mathbf{x}_\mathbf{a}^\mathbf{T}) + b) \tag{2.18}$$

In simple terms, the linear combination of $\tilde{\mathbf{w}}$ an x_a is used to predict the value of $\mathbf{y_a}$, a trait similar to neural networks (NNs).

2.11.2 Non linear SVM

At times, the input data examples can be difficult to separate with a linear hyperplane. In such cases, the original data is mapped to a higher dimensional space in order to separate the data examples. In Figure 2.10, the points on the 1-dimensional line cannot be separable by using a straight line. Whereas, the same input points can be separable when mapped to a 2-dimensional space i.e $x :\rightarrow x^2$ [See the parabola at the bottom of Figure 2.10]. For non-linear classifications, a kernel trick is applied to the SVM. Which is similar to the original SVM classifier but the dot products on the dual Lagrange problems are replaced with a non-linear kernel function K. So equation Equation 2.14 becomes:

$$\mathscr{L}_D = \sum_{i}^{n} \lambda_i - \frac{1}{2} \sum_{i}^{n} \sum_{j}^{n} \lambda_i \lambda_j y_i y_j K(x_i, x_j) \qquad \textit{for } i \neq j. \tag{2.19}$$

Where K is function that maps x to a higher dimensional space. Some of the commonly used kernel functions are: polynomial (homogeneous), Gaussian radial basis function (RBF) and a hyperbolic tangent function [Table 2.1].

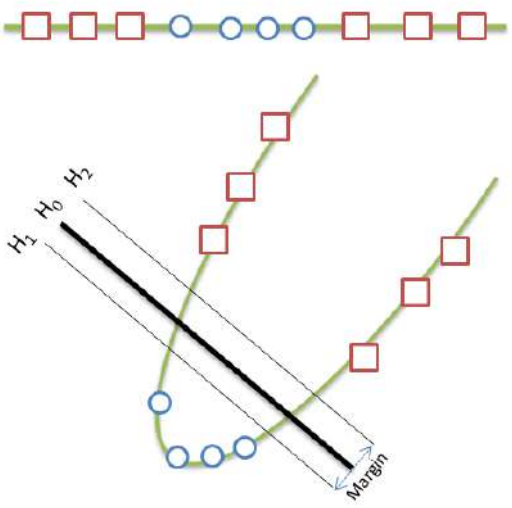


FIGURE 2.10: A basic example on how non-separable input points in a lower dimensional space can be mapped to a higher dimensional (in this case from 1D to 2D) space to make them separable.

The SVMs have been noted to have common issues. The fist is the fact that they are mostly suitable for binary classification tasks. However, there are various methods for reducing a multi-classification problem to binary classification to perform multi-classification tasks. Another issue arises from the dependency of higher-dimensional

TABLE 2.1: Typical kernels functions of the SVM classifier.

function name	**Example**
Polynomial	$k(\vec{x}, \vec{x}) = (\vec{x}, \vec{x})^d$
Gaussian radial basis function	$k(\vec{x}, \vec{x}) = exp(-\gamma \|\|\vec{x_i} \cdot \vec{x_j}\|\|^2)$ for $\gamma \geq 0$
Hyperbolic tangent function	$k(\vec{x}, \vec{x}) = \tanh(\vec{x_i} \cdot \vec{x_j} + 1)$

feature spaces, which increases the generalization error of the SVMs. Fortunately, this can be mitigated by using a sufficiently large training set. Of which, makes the SVMs to still perform well.

The SVMs can achieve high classification rates when trained well. Yu-Dong Cai et al [45] obtained detection rates that were over 90% when predicting Membrane Protein types. Cuingnet et al. [46] achieved a high classification rate with SVMs when detecting the differences of group levels in magnetic resonance imaging (MRI) brain scans. Support vector machines are also highly preferred for text recognition, Datong Chen et al presented a text-extraction algorithm that selects text from complex backgrounds in images and applied SVMs to recognise the text at an identification rate of 98% [47]. Due to the possible high classification results, Dalal et al [1] also used SVMs when they presented the original HOG-classifier.

2.11.3 Neural Networks

Neural networks are mathematical models that can approximate functions that depend on large input data. Their ability to learn from known observations have attracted a lot of attention from researchers [48, 49, 50]. The neural networks were inspired by the biological networks of neurons in animal brains. The term network refers to the interconnections of nodes between adjacent layers. The research behind neural networks dates back to 1943, Warren S. McCulloch [51] was the first to describe the conceptual model of neural networks. This lead to further research by Rosenblatt [52], who was first to model the basic function of a perceptron. By then neural networks had limitations to the problems they could solve, they were unable to solve simple logical operations such as XOR until the introduction of backpropagation by Paul Werbos [48].

Similarly to biological neurons, nodes are the individual elements of the neural network. They simply read input, processes it and generate an output [Figure 2.11]. The connections between nodes have adaptive weights that are optimised by learning algorithms. A layer is a sub-ground network or a list of nodes that pass information to other nodes

when activated. For feed-forward neural networks, information is passed in a single direction from input layers, through hidden layers until it reaches the output nodes. The input layer normally takes input-data in a vector form. The network's weights are randomly chosen during the network's initial state. For each neuron, the linear combination of input data and their weights are passed to an activation function [Figure 2.11]. A node can have an arbitrary number of inputs. On fully connected neural networks, each neuron has input from all nodes in the previous layer [Figure 2.12].

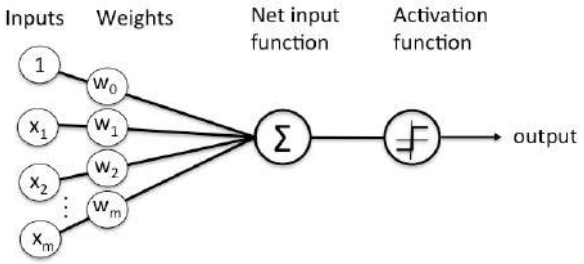


FIGURE 2.11: The perceptron, a basic form of a neural network [53].

For the perceptron to predict desired values, supervised training is applied. Which means, the perceptron is given input data to which labels are known. The predictions are then analysed to evaluate the error rate. The perceptron learns from the previous classification mistakes by adjusting the input weights according to the error rate, this process is called backpropagation. A perceptron is formally defined as follows, given a set of N observations $\mathbf{x} = \langle x_1, x_2, ..., x_n \rangle$ of dimension L, a function of a neuron $H(x)$ is defined as:

$$H(x) = \sum_{i}^{L} g(\mathbf{w_i x} + b_i). \tag{2.20}$$

where $g(x)$ is the activation function and b_i is the bias to the i'th hidden layer. The activation function $g(x)$ can be a sigmoid functions, radial basis, sine, cosine or tan etc. The vector $\mathbf{w} = \langle w_1, w_2, ..., w_n \rangle$ contains the input weights of the neuron. The sigmoid function $g(wx + b_i) = \frac{1}{1+\exp^{w^T x}}$ is often preferred as the activation function. Traditionally, gradient descent algorithms are used to train single layer feed-forward neural networks. Gradient descent algorithms minimise the classification error through optimisation techniques. That is, if $\mathbf{T} = \langle t_1, t_2, ..., t_n \rangle$ is the set of known labels for N observations, a perceptron is trained by finding the $\mathbf{w}$ such that

$$\mid H(x_i) - T \mid \quad = \quad arg_{\mathbf{w}} min \mid \sum_{i}^{L} g(\mathbf{w_i x} + b_i) - T \mid . \tag{2.21}$$

the local minimum of the function $\mid H(x_i) - T \mid$ leads to the optimal weights w for the prediction performance of the perceptron. Although a single perceptron can solve a series of linearly separable problems. The power of neural networks comes from connecting a sufficiently large number of perceptrons to work together. A network of perceptrons can solve non-linear classifications and other intelligent tasks.

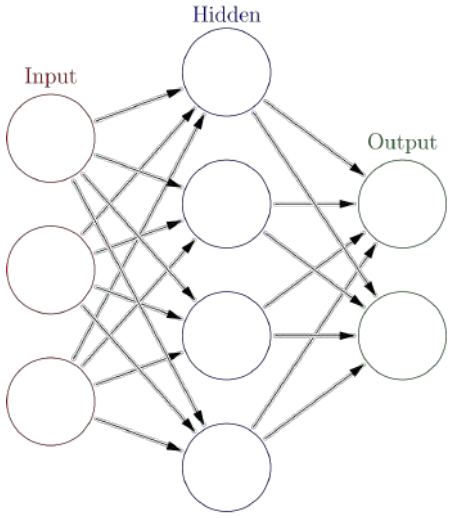


FIGURE 2.12: A simple neural network structure with 3 layers [54].

2.11.4 Extreme Learning Machines

The neural networks have dominated machine learning and data analysis applications for some time. They have been criticised for requiring long periods of time to train. Huang et al [49] presented a new variant of feed-forward neural networks called ELMs. The ELMs can be trained faster than traditional neural networks. The weights between the hidden nodes are randomly selected and never updated. The weights from the hidden layer to the output layer are learned in a single step using activation functions. To achieve this, the activation function must be differentiable on all intervals in the set of real numbers R. Huang et al [49] presented a theorem that proves for any given training set there exists an ELM that gives a minimum training error with hidden nodes that are not more than the actual number of distinct training samples.

Unlike traditional neural networks, Huang et al [49] proved that a neural network can be trained in a single step without iteratively tuning the weights of hidden neurons. They showed that the initial randomisation of hidden nodes is sufficient if you train the hidden layers output weights using a non-linear mapping function. Thus the ELM's principle is, the training process of the output weights of the hidden layer should be independent of the weights of the inner hidden layer. Furthermore, the norm of these output weights should be as small as possible, to maintain the generalisation capabilities of the ELMs. The biases are traditionally applied to the output hidden nodes, however, Huang et al [55] proposed that the output layer should be void of biases as they lead to sub-optimal solutions.

Although ELMs are faster to train, they require a large number of hidden nodes than conventional neural network since the hidden nodes in ELMs are generated randomly [55]. Forwarding data on large networks can be slow for runtime applications. Quite a large number of ideas have been published to address this problem [55, 56, 57]. An incremental ELM (I-ELM) adds nodes dynamically during the training process to make the hidden layer network more compact than the original ELMs. The nodes are added to the network until the desired generalisation is achieved. The selection is based on the contributions of nodes to the training performance.

With adaptive ELM (A-ELM) [56], the amount of the hidden nodes can change during the training stage. This avoids the common requirement to restart the training process to change the number of nodes. The A-ELMs apply an elastic method to improve incremental learning whenever there is a change to the network. Due to the added operational overhead, Zhang et al [56] used Map Reduce to speed up matrix calculations.

A two-stage ELM (TS-EML) that automatically determines the ideal size of the network was proposed by Yuan Lan et al [57]. A termination criterion for the final prediction error is initially set. This is followed by a "forward selection" process that randomly generates multiple groups of hidden nodes. Each time, the group with the highest net contribution is selected and added to the network. This process terminates when the network reaches converges to a minimum prediction error. The second stage does a backward elimination task of removing insignificant nodes using the leave-one-out (LOO) cross-validation technique. This completes the brief background on ELMs. The next section looks at another variant of neural networks called convolutional neural networks.

2.11.5 Convolutional Neural Networks

Convolutional neural networks are mostly applied for analyzing visual imagery because they perceive visual information similarly to the visual cortex of animals. The biological visual cortex can process overlapped visual information from two eyes to better understand the 3-dimensional world. The connections of convolutional layers are organised in overlapping layouts to respond to overlapping tiling on visual information [58], a trait similar to sliding descriptor windows. Thanks to the overlaps, shifts and space invariances are handled automatically [Figure 2.13].

The capability of CNNs in perceiving image data lead to a large number of computer vision applications [26, 58, 59]. Facebook uses CNNs for face recognition [59]. Vehicle manufacturers use CNNs for autonomous feedback systems in smart vehicles. Pedestrian detection and traffic-sign recognition tasks have become effective on some of the latest smart vehicles [39].

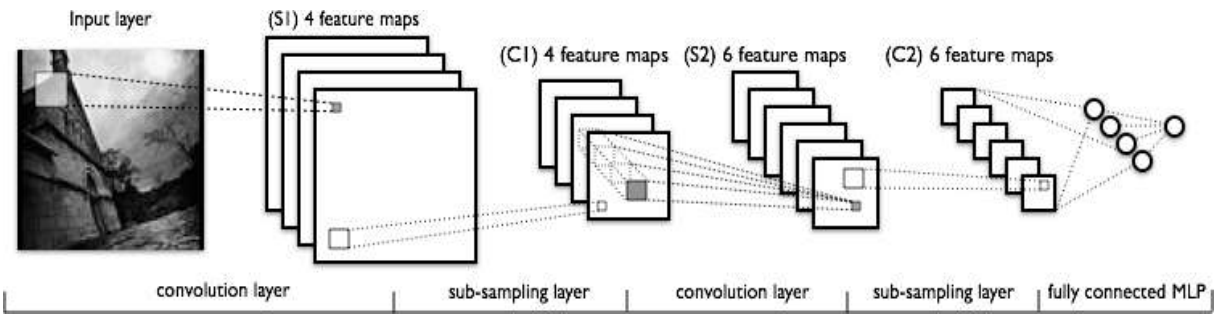


FIGURE 2.13: Convolutional Neural network diagram [53].

Convolutional neural networks were first applied by Yann Le Cunn [58] in 1995. Yann demonstrated the ability of CNNs with a handwritten character recognition task using the famous LeNet 5. The idea was to eliminate the need for fully connected layers in the initial stages of neural networks because fully connected layers often result in overfitting. The initial layers were replaced with layers that learn the desired filters for characterising the descriptive shapes of objects.

Moreover, fully connected networks require large amounts of memory when dealing with high-resolution images. The convolutional layers force the extraction of local features by restricting the connections of nodes from the previous layers because different types of filters are computed by the convolutional layers.

After each convolutional layer, a sub-sampling layer performs the local averaging of the weights and the subsampling of the feature maps. This reduces the resolution of the feature maps. This is done to reduce that classifier's sensitivity to scale and distortion. The combination of convolutional and subsampling layers are repeated densely until

their output can optimise the generalisation of a fully connected neural network or other similar classifiers.

The structure of the convolutional neural network is designed to first map images into sequences of feature maps. At the beginning of the network, separate neurons process different portions of the image. The first layers handle the simple outer shapes of the objects separately. The layers after extracting local complex shapes. Eventually, the last convolutional layers, select the overall pictorial shapes that can distinguish the objects.

2.11.6 Characterising Performance

In binary classification tasks, the receiver operator characteristic (ROC) or precision and recall (PR) curves are often used for comparing the performances of different algorithms. The trend shifted away from simply presenting classification accuracies for validating performance as they can be misleading [60].

While the ROC curve and PR curve share some properties, the ROC curve can show highly optimistic evaluations of classifiers when the number of samples between the two classes is largely skewed. This situation is common with most binary classification tasks where one of the classes is unknown. For instance, when a distinctively definable object has to be distinguished from other generic objects, the second class becomes random background images. Thus in oftentimes, the number of background images should be larger than the number of images with the known class to train classifiers effectively. The databases used in this current research also possess this trait, as a result, the PR analysis was selected for evaluating performance than the ROC.

To plot the PR graph, the precision is computed using the formula in Equation 2.22 and Equation 2.23 is determined at varying thresholds of the classifier's decision functions. Decision functions assign a threshold to select rules for declaring samples to be positive or negatives.

$$\rho = \frac{TruePositives}{TruePositives + FalsePositives}, \tag{2.22}$$

whereas the recall is calculated by taking the number of true positives and diving it by the total number of positive samples on the scene, stated by the ground-truth information. Therefore, the recall is,

$$\gamma = \frac{TruePositives}{TotalNumberOfPositives}. \tag{2.23}$$

In Figure 2.14, the two curves were plotted from the results of the same classifier. The ROC curve simply compares true positives with true negatives. Whereas with PR curves, they compare false positives with true positives to capture the effects of using larger negative sets, this measures the robustness of classifiers. The goal of classifiers on PR graphs is to reach the top right corner, whereas the goal on ROC graphs is to reach the top left corner. Meaning, a good classifier would get a PR curve that has the properties $precision \approx 1$ and $recall \approx 1$.

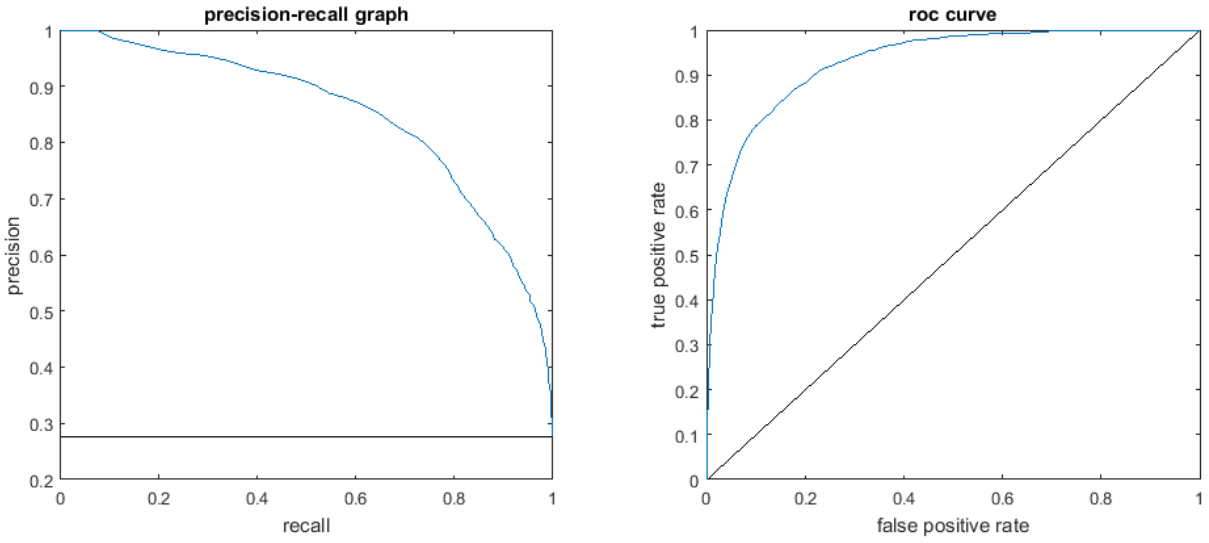


FIGURE 2.14: Typical examples of ROC and PR curves.

2.12 Summary

The reviewed literature shows how the research on human detection systems require further experimental research, analysis and the review. To the best of our knowledge, none of the research in literature has evaluated the significance of using infrared-based human-datasets instead of colour images for human detection systems. Instead, more effort is placed in tuning statistical learning classifiers rather than improving the quality of acquired features. There is still a necessity to find better feature descriptors that are robust in finding patterns of non-rigid objects. This research addresses this gap by investigating whether image features can be improved by reducing the clutter of unnecessary edges in gradient images.

In hopes of finding new ways of improving human detection, this research intends to contribute to the research field by (1) producing a new infrared dataset (with annotations), (2) evaluating the significance of using infrared images when extracting image features of people, (3) test the application of shallow-learning and deep learning models

and (4) providing further experimental research on image processing methods and the classification of people in static images.

Compared to other hand-engineered features, the HOG features were selected due to extensive use and promising results in the literature. The SVM classifier was selected as a baseline to compare the performance of previous work and the other two classifiers (ELMs and CNNs) used in this work. The SVMs and ELMs are faster to train, the objective was to test the significance of using infrared samples on both shallow learning classifiers and deeper convolutional neural network. Testing three different classifiers provides findings that are sufficient to a limited extent to address the supposition of this work. The following chapter presents the infrared dataset that was created for this research.

Chapter 3

Infrared Dataset Creation

The lack of sufficiently large, labelled and high-resolution infrared human datasets led to the creation of the SIGNI dataset for this research [7]. Existing infrared datasets from the literature had low-resolution images which were mostly captured from elevated platforms and longer distances. For instance, the people from [61, 62] are relatively small and are represented with a low vertical resolution because the images were photographed from long distances and aerial positions. These were some of the motives for creating a new dataset with clearer and higher-resolution images.

The specifications of the SIGNI dataset and the infrared camera used for capturing samples are discussed in Section 3.1. Section 3.2 presents the region-of-interest extraction tool and the ground-truth recording tool used in this research. Figure 3.1 shows a sample of large images that were used for extracting the regions of interests. Some of the positive image samples from the SIGNI are shown in Figure 3.2 followed by some of the negative images in Figure 3.3.

3.1 SIGNI Dataset

The SIGNI dataset was created for evaluating the significance of using infrared images when extracting image features of people and includes images that can be used to train supervised classifiers. The dataset also includes human detection testing scenarios with ground-truth annotations and can be downloaded from [7].

Creating large datasets can be time-consuming. To speed up the process of creating the IIR dataset, a tool for extracting ROI sub-images from large images was implemented. The application includes an option for recording ground-truth information and provides a graphical user interface that equips users with click and drag functionalities to select

image regions that contain the desired samples. Further details of this tool and a link to download it are available in Section 3.2.

The SIGNI dataset contains 5462 distinct negative samples and 1916 distinct positive samples. The number of positive samples was doubled to 3832 by adding horizontal reflections of the images. The negative infrared samples were extracted from the background images of the Infrared ASL Dataset [62]. Unlike the positive images, the background images consist of generic objects and vary considerably. When such cases occur, the number of generic images is often set to be higher than the number of definable-class (positive set) images to improve the effectiveness of the classifiers in detecting foreground objects. The datasets used in this work also have this trait, as discussed in Section 4.1. As a result, more background-images were selected for training the classifiers.

The dataset samples were created as follows: image sequences were extracted from recorded videos to manually label and obtain the bounding boxes that contain people. The coordinates of image-subregions with recognisable human beings were annotated and stored in an extensible markup language (XML) tree. Each subregion was labelled positive if it contains a recognisable human and negative otherwise. This process produced a new infrared human dataset named the SIGNI Human Dataset.

To the best of our knowledge, an alternative infrared dataset used by [63] was unavailable online. Combining samples from this dataset with the SIGNI dataset may improve the generalisation accuracy of classifiers. Images from [63] have a low resolution of 24×54, whereas, samples from the SIGNI dataset have a 64×128 dimension. A selection of positive and negative samples from the SIGNI dataset are shown in Figures 3.2 and 3.3 respectively.

3.1.1 The Camera Used

The IIR footage was captured with the *Xeneth Gobi-640-GigE* camera shown in Figure 3.4. This camera is a non-cooled thermal camera that is capable of transferring high-resolution infrared images through a GigE interface at 50 Hz. With a spectral-range of $8\mu m$ to $14\mu m$, this camera is capable of sensing low-temperature bodies (like people) accurately.

This camera is ruggerized and suitable for industrial applications. The camera is reliable for non-contact temperature measurements and offers high thermal resolution quality of 0.05 Celsius for accurate thermal analysis. Infrared cameras create images from infrared radiations and show contrasts between warm and cold materials [Figure 3.1].



FIGURE 3.1: Sample large images from the SIGNI dataset. The images were captured in different environments and contain enough variance in the appearance of people. Human subjects were captured during winter and summer, in different poses and wore different clothes.



FIGURE 3.2: Some of the positive samples from the SIGNI dataset.

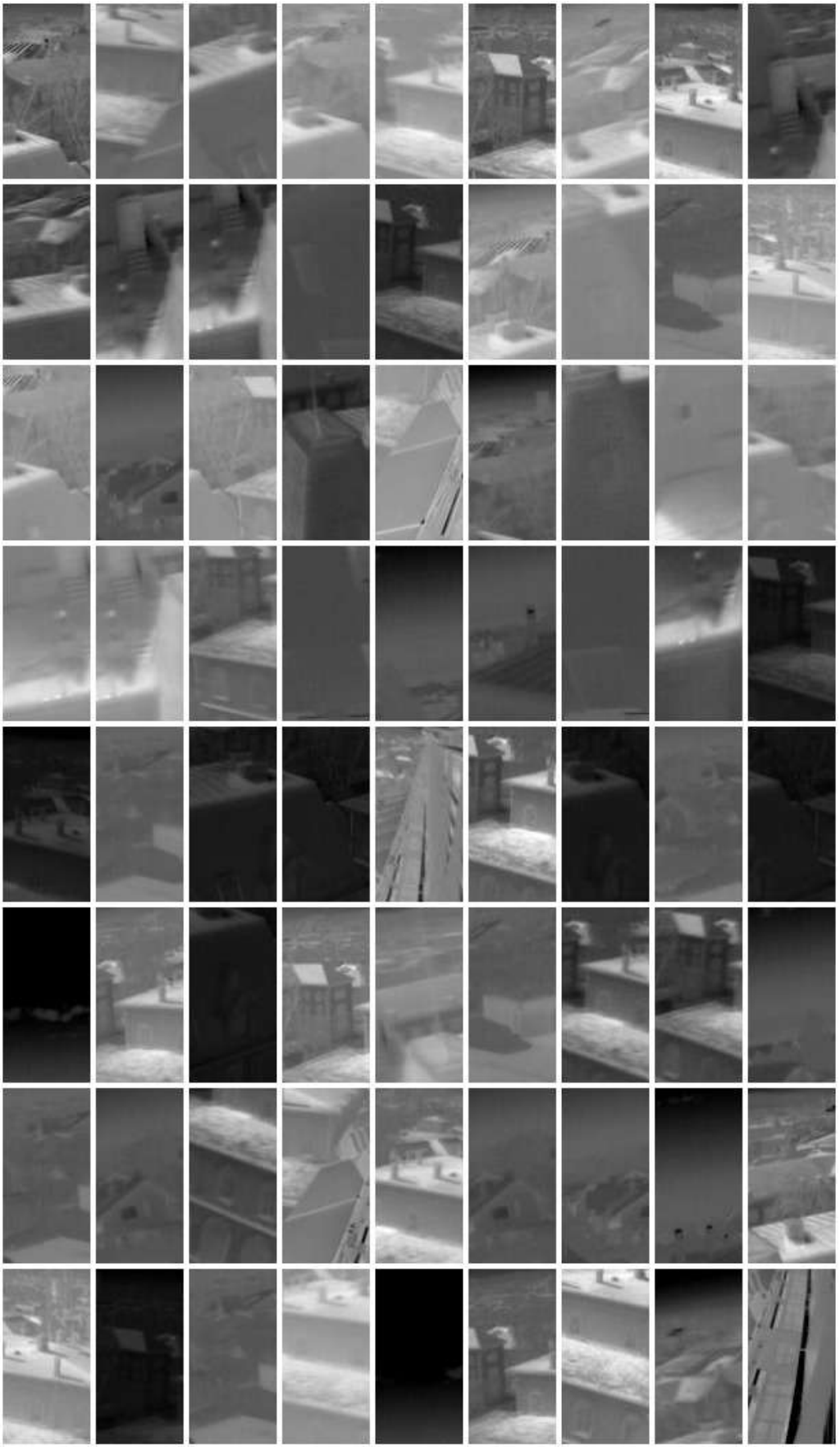


FIGURE 3.3: Some of the negative samples from the SIGNI dataset.



FIGURE 3.4: The compact thermal camera used for capturing IIR samples [64].

3.2 ROI Extraction and Grouthtruth

The ROI extraction application was implemented to speed up the process of collecting training samples from a large number of collected images. The application works as follows, a user first defines the type of samples to load which can either be class names, class indexes, positive or negative samples. Thereafter, the application looks up all images in the given directory and stores their relative path information into a basic XML tags. Through a graphical user interface, the user can then iterate through each image and use the mouse-drag functionality to select ROIs they intend to extract. Multiple ROIs can be extracted from the same image. The image coordinates of the extracted ROIs are stored in a simple XML file for the validation of ground-truth information once a statistical learning classifier has been trained. Since the size of people varies in images, the extracted ROI's resolution is snapped to a ratio of $2:1$ for the height and width to keep the correct scale of people when the images are resized to the 128×64 resolution required by the standard HOG feature extractor.

Unlike the common use of text files for storing annotation data, XML files were selected for this work because they provide an architectural structure that is easier to read and understand. Computers process data much easier when it is stored in memory, thus looking up information in XML files becomes simpler. Unlike with text files, where the operating system's level code for reading files has to be executed rapidly.

The XML tree structure allows annotation data to be extended or modified to other annotation formats. And most importantly, different tags can be used to describe discrete images. This organises data more effectively, a trait useful for the verification of stored information through XML schemas. A typical example of the XML files used in this work is shown in Figure A.2 and the graphical user interface of the ROI extraction application is shown in Figure 3.5. This tool was created with Python 2.7 and OpenCV. The tool's source-code and documentation are available from the link in [65].

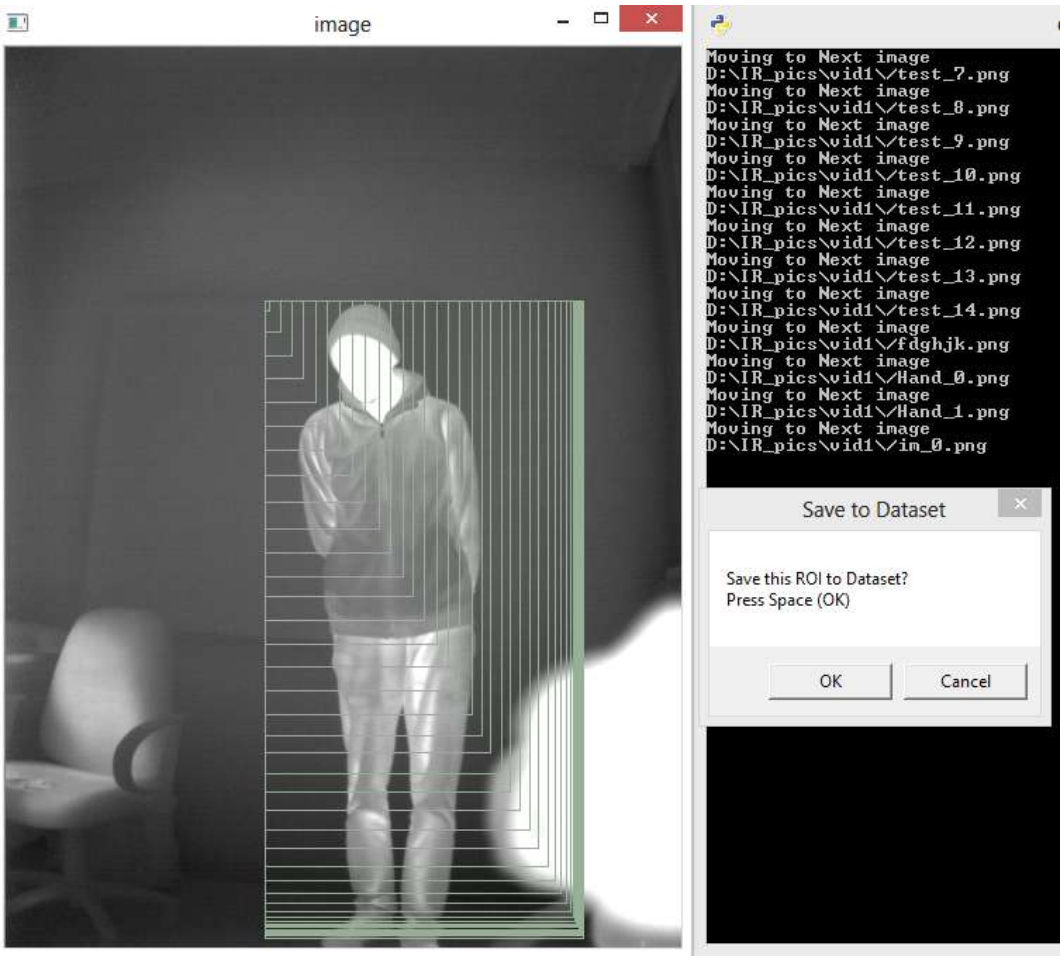


FIGURE 3.5: Graphical user interface for the ROI extraction tool

3.3 Ethics Clearance

The positive samples from the SIGNI dataset were captured from a *noncontrolled background setting*, where people were photographed anonymously from random viewing angles. Since infrared images lack colour information and the textures that are useful for identifying people, the University of the Witwatersrand provided permission and the ethical clearance to capture the image samples for the SIGNI dataset. The clearance letter is shown in Figure A.6.

None of the names or affiliations of the individuals from the image samples is mentioned. The identity of people was not recorded nor published as part of the dataset.

3.4 Summary

This chapter presented the details of the infrared SIGNI dataset. The rationale for creating this dataset was discussed, also with the tools that were used for creating the dataset. Some of the large images that were used for extracting the training samples are shown in Figure 3.1 and some of the positive image samples from the SIGNI are shown in Figure 3.2 followed by some of the negative images in Figure 3.3. The research methodology of this work is presented in the next chapter.

Chapter 4

Research Methodology

This chapter presents the methods that were followed when carrying out the research. Section 4.1 reviews the collection and selection of the pedestrian datasets used in this research, this is followed by a detailed discussion on the feature extraction methods used for producing meaningful information from the raw image pixels in Section 4.2. The design and implementation of the classifiers is presented in Section 4.3, this includes references to the libraries used. The chapter concludes by examining how the research questions can be addressed and how the hypotheses can be tested [Section 4.4]. The standard process for classifying images is shown in [Figure 4.1].

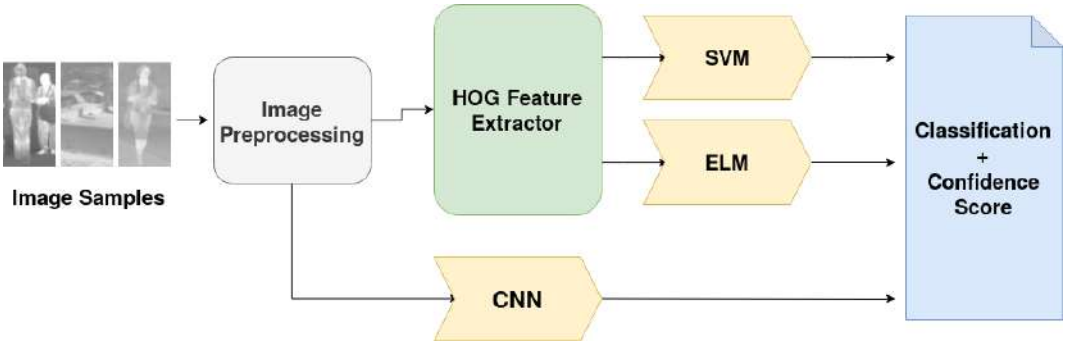


FIGURE 4.1: Image classification process.

4.1 Data Collection

Sufficiently large image datasets are required to train supervised learning classifiers. Since large images can contain regions of the desired positive samples within large areas of background information, the classifiers have to be trained to distinguish between background information and foreground information. Therefore, images that contain people

and the randomly generated background images which some were cluttered scenes were collected. Unlike positive foreground samples, negative samples can be easily extracted from significantly small datasets automatically, to produce way more negative samples than positive samples. As a result, the common imbalance between negative and positive samples should be handled with care. In this work, the number of negative samples was restricted at a reasonable size that is less than three times the number of positive samples to avoid biases.

To generate **negative** colour samples, image subregions of size 128×64 were iteratively cropped from the large background images of the INRIA training dataset to extract approximately 15 000 negative subregions. The negative samples were cropped without any overlap to keep them discrete. The background images were natural scenes that contained anything except people. The images had landscapes, mountains, rivers, plants, animals and vehicles etc.

A similar approach was followed when generating 5462 negative infrared images. Most of the **positive** samples were captured from a *noncontrolled background setting*, where people were captured anonymously from random viewing angles. For the colour images, three publicly available datasets were collected online, the INRIA [1], MIT [66] and NICTA [9] human datasets. With the IIR images, a new dataset was created as part of this study [See Chapter 3]. The images were extracted from recorded video footages, captured at two different settings with a high frequency of pedestrians. Further details of datasets are shown in Table 4.1 and Table 4.2.

TABLE 4.1: Visual Datasets sample count.

Visual Datasets	**Positives Samples**	**Negatives Samples**
MIT Pedestrian Dataset	924	18000
INRIA Pedestrian Dataset	3610	18000
NICTIA Pedestrian Dataset	15000	18000

TABLE 4.2: IIR Dataset sample count.

IIR Dataset	**Positives Samples**	**Negatives Samples**
SIGNI Human Dataset	3832	5462

When considering the number of positive samples alone, the SIGNI dataset is approximately the same size as the INRIA dataset that has 1805 distinct positive samples [See Section 3.1]. Comparing classifiers that are trained with the SIGNI dataset to classifiers trained with the INRIA dataset becomes ideal. During the experiments, the SIGNI dataset was tested against the INRIA [8] used by Dalal et al [1] for the original HOG human detector, and a smaller subset of the NICTA dataset that was reduced to the same

size as the other dataset [9]. The MIT pedestrian dataset was only used for validating the hyperparameters of the classifiers before training the final candidate-classifiers used for this work.

If larger datasets are required, then data augmentation may be necessary to virtually increase the number of positive samples on the smaller datasets to match the larger NICTA dataset. This can be achieved by augmenting each sample with 8 more representations through scaling, shear and slight translations to produce approximately 15000 images.

Data Selection:

On each dataset, eighty percent of the samples were randomly selected for training and validating the classifiers and twenty percent were reserved for testing the classification performance. About 100 unseen images were used for evaluating the ground-truth performance of the sliding window detector. The number of images used for ground-truth validation was kept small to regenerate experiments at reasonable times because applying a sliding window mechanism with scale-pyramids on large images of 640×480 can consume a great amount of time.

4.1.1 Data preprocessing

In this research, image pre-processing is referred to as the operations done on images prior to their use. On each extracted region of interest, horizontal and vertical edge extraction filters were applied to produce the magnitudes of the gradient images. This was followed by a HOG feature extraction step described in Section 4.2. Since data may contain duplicates, noise clutter, inconsistencies and distortions, the mean of the training set was computed and subtracted from all training and testing samples to centralise the image-feature. Traditionally, centralising datasets avoids the over-fitting requirement from most classifiers.

4.2 HOG feature extraction

The fundamental idea behind the histogram of oriented gradients is that the basic shape of objects can be described using its contour edges and that the intensity information that makes up these edges can be quantized to fewer values by distributing their frequencies on histograms. For this work, a HOG feature extractor was manually built similarly to the state of the art human detection system that was first presented by Dalal et al [1].

The HOG feature extractor module converts images to grids of HOG cells. This is done as follows: for each pixel the gradient vectors are calculated by taking the difference of the vertical and horizontal neighbouring pixels. With the example in Figure 4.2, the gradient vector on the centre pixel will be calculated as $\vec{v} = \langle x2 - x1,\ y2 - y1\rangle = \langle 113-35,\ 75-45\rangle = \langle 78,\ 30\rangle$. Letting D_x and D_y be single-vector filters and the original image be I, the horizontal and vertical gradient images I_x and I_y respectively can be computed as $I_x = I * D_x$ and $I_y = I * D_y$, where $D_x = \langle -1, 0, 1\rangle$ and $D_y = \langle 1, 0, -1\rangle^T$. Therefore, the magnitude of the gradients image is $M = |\langle I_x, I_y\rangle| = \sqrt{I_x^2 + I_y^2}$. Thus in Figure 4.2, the centre pixel's value from the *magnitude of the gradients* image is $M = \sqrt{78^2 + 30^2}$. The image illustrations for each of these variables is shown in Figure 4.3.

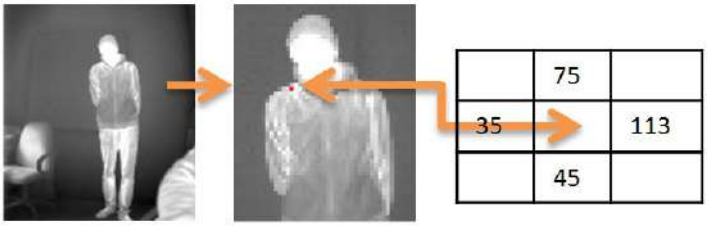


FIGURE 4.2: The neighbouring pixel values used for computing the centre pixel's gradient vector and magnitude value.



FIGURE 4.3: The horizontal gradient I_x (1st), vertical gradient I_y (2nd), magnitude of gradients M (3rd) and the cell division of the magnitude of gradients image (4th).

After computing the *magnitudes of gradients* for all pixels, depending on the applications, there exist rectangular (R-HOG) bins or circular (C-HOG) bins such that the shape information of objects can be characterised. Dalal et al [1] obtained similar performances from both bin types. Consequently, the rectangular bins were used in this work for simplicity and easier implementation. With R-HOGs, the image is normally divided into 8×8 pixel cells where the corresponding nine-bin histograms are created for each

cell. The histograms represent the frequency of the *magnitudes of the gradient* vectors within the cells.

A split partial-contribution between neighbouring bins is considered based on each pixel's gradient-vector angle. This depends on the centering of the bins which are traditionally centred at $\{10, 30, 50, ..170\}$. For instance, if a vector angle of some pixel is 45° then $1/4$ of the magnitude is added to the bin centred at 30° and about $3/4$ is added to the bin centred at 50°. An illustration of a histogram representing a single cell is shown in Figure 4.4. Once all histograms are created, they are normalised based on a 4×4 cell-block that slides through the entire image. Except for the cells on the edges, inner cells are normalised more than once since the window is slid per cell-block. Normalisation is done by putting four histograms into arrays of length 36, this equates to the number of values in four histograms. The purpose of the normalisation is to make histograms invariant to the varying contrast in images [1], this step completes the feature extraction process.

Traditional HOG based pedestrian classifiers are trained with a collection of 64×128 resolution images. This work followed the same approach, as image samples were converted into the same resolution without affecting their scale proportions. The size of the ROIs remained the same for both IIR and visual images, thus a HOG feature matrix for a single ROI had a size of $16 \times 8 \times 9$ since 8×8 image cells and 9 bin histograms were used. These matrices were then reshaped to 1×1152 feature vectors. At this stage the features are ready to be used to train a machine learning classifier.

Once the classifiers are trained, human detection can be performed on larger images of any resolutions by applying a 64×128 sliding detection-window, where the detection window is used as a descriptor on all image sub-regions where the sliding window halts. Thus on each new sliding window position, the classifier predicts whether the sub-region contains a person or not. To deal with the scale invariance of people, the resolution of the detection window is kept constant and the input image is scaled down by a constant factor on every completed convolution, the scaling repeats until the large image's size is less than or equal to the detection window.

Once the HOG feature extraction method was implemented, the next task was to implement the required classifiers to conduct the necessary experiments.

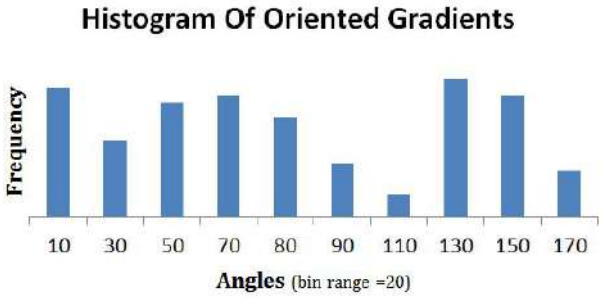


FIGURE 4.4: A typical example of a histogram of oriented gradients for a regular 8 pixel resolution cell.

4.3 Implementation of Classifiers

This section covers the design and implementation of the classifiers and brief code references to the library used to implement them to allow this research to be reproducible.

4.3.1 SVM Classifier on HOG Features

The SVMs can perform non-linear classification by mapping data into a higher degree polynomial space to improve the performance. For the current work, the SVM classifier was programmed with Microsoft Visual C++ 2010 programming environment and OpenCV's version 2.4.8 library which inherits tools from the common LibSVM library. The standard HOG feature extractor was developed from first principles with the OpenCV library and this was based on the work by Dalal et al [1]. Training the SVM classifier for binary classification was done as follows:

Given a vector containing the labels of samples and their corresponding training samples stored in OpenCV's matrix datatypes cv:Mat, one can create an SVM instance of type CvSVM that can be trained using the function CvSVM::train(TrainingData, Labels, ..., params), where the *params* are options that specify the type of SVM to train, the SVM kernel to use and also the criteria to limit the classifier's maximum training iterations. The OpenCV library supports most common SVM kernels. The linear, polynomial and radial basis function (RBF) kernels were tested in this work. The sigmoid kernel function and other user-defined custom functions can also be used as kernels. The following code snippet shows how to typically set parameters for the SVMs structure.

In this case, the SVM type CvSVM::C_SVC caters for n classifications ($n \geq 2$) and allows imperfect separations of labelled classes with the penalty multiplier c for outliers. Other SVM types include (but not limited to) CvSVM::ONE_CLASS and the

CvSVM::EPS_SVR. The former does distribution estimation to build a boundary that separates a single class from the rest of the feature space, where all training data is from a single class. The latter is called Support Vector Regression (SVR), this model requires the distance between feature vectors and the hyperplane to be less than some constant p and similarly to C_SVC, the penalty multiplier c for outliers is also used. With the penalty multiplier, class weights can be assigned to affect the misclassification penalty for certain classes by setting a value to the parameter *parram.class_weights*.

```
1 CvSVMParams params;
2 params.svm_type    = CvSVM::C_SVC;
3 params.kernel_type = CvSVM::LINEAR;
4
5 params.term_crit.type = CV_TERMCRIT_ITER  + CV_TERMCRIT_EPS;
6 params.term_crit.max_iter = 1000;
7 params.term_crit.epsilon = 1e-6;
```

Once the SVMs are trained, the function CvSVM::predict(sample) can be applied to test the performance of the SVM when classifying samples. This function can either return the predicted labels or can be set with an additional boolean parameter to return the signed distance of the testing sample from the hyperplane's margin. Since SVMs determine the best hyperplane that separates distinct classes on the condition that the distances from the hyperplane to the first training samples from both classes are maximised, this implies that the further the mapped sample is from the hyperplane the more confident the SVM is on its classification decision. Hence, this distance can be used to model the confidence score (probability) in the range of $[0, 1]$. To do this, the following distance to a sigmoid function is considered:

$$\frac{1.0}{(1.0 + \mathbf{e}^{-decision})}. \tag{4.1}$$

For analytical purposes, the library provides methods that provide information about the trained support vectors. The function CvSVM::get_support_vector_count() returns the total number of support vectors used in the problem after training the SVM. And with CvSVM::get_support_vector() one can obtain each support vector by simply supplying its index to determine which training examples were used as support vectors. This summarises the library references and implementation of the SVM classifier.

4.3.2 ELMs Classifier on HOG Features

For the second classifier, the open-source Matlab code-repository from [67] was used to implement the extreme learning machines. Unlike the convolutional neural networks, there was no difficulty in getting this source code to compile as only standard Matlab libraries are used. Nevertheless, the ELMs are much easier to implement since they resemble the behaviour of basic neural networks. The MIT training data used for validating the parameters of the SVMs was restructured to be compatible with this library. The ELMs classifier was then tested to ensure it worked correctly. During the initial tests of ELMs, a classification rate of 89.8% was obtainable without pruning the hyper-parameters of this classifier, with the default parameters of the sigmoid activation function.

The library contains two main scripts, the *elm_train.m* and the *elm_predict.m*. The *elm_train* function first preprocesses the data before the training process, then automatically finds and sorts the samples based on their labels. Thus finding the unique labels in the dataset to determine the number of output nodes needed for the final layer of the ELM network.

Once the data has been prepared then training ELMs can be done as follows, the input weights and the biases of the hidden neurons should be randomly generated. Each training sample is then passed to the network by multiplying its values to the input weights. The hidden output matrix can be calculated using the sigmoid function, $sin()$ or the hard limit transfer function. Other activation functions like the *Triangular basis* and *radial basis* can also be used. The final step calculates the output weights and determines the network's training and testing accuracies.

The *elm_predict* function loads the trained ELM and just applies the weights and the biases of the neural network to the testing data. The activation function is then applied to activate the nodes and the weights of the output layer are multiplied to classify the data. At this stage, the ELMs may predict if image regions contain a person or not.

4.3.3 CNNs Classifier on HOG Features and Gray-Scale Images

Convolutional neural networks often require large amounts of data to learn from, hence efficient and parallel computing is often required. Unlike the ELMs classifier, libraries that implement CNNs are often not trivial due to the efficient matrix or tensor computation requirements. Graphical Processing Units are often ideal to train CNNs at reasonable times. The initial layers of CNNs also involve notable iterations of convolving image-filters and subsampling, which can be computationally expensive. Due to such

reasons, it is often recommended to use some of the efficient libraries such as CAFFE, MatConvNet, TensorFlow or PyTorch to implement CNNs.

When selecting a CNNs library, the priority was to consider one that allows researchers to focus on the study instead of dealing with the challenges involved with the low-level implementation of CNNs. The TensorFlow or PyTorch libraries were not available yet when this research commences in 2015.

Therefore, the open-source MatConvNet library [68] became the toolbox of choice for this research due to the simplicity of use and flexibility to modify CNN architectures. With the support of NVIDIA's GPGPU processing, it is easy to interchangeably train the CNNs with CPU and CUDA. Since efficiency is important with CNNs this became another highlight when choosing this library. Furthermore, CNNs can be trained with either grayscale or colour images and non-square images are also supported as input. The documentation of this library includes several examples that show how networks can be modified or trained.

Due to the small size of datasets used in this research, and the restriction not to use transfer learning to discriminate the effects of reducing noise on human detection, the CNN architecture used is based on the *LeNet-5* architecture from [58]. This example shows how CNN can be trained to recognise handwritten digits between 0 and 9.

To build a simple CNN that learns from each arbitrary image img of resolution $(w \times h)$, the first task is to create several random filter banks. One can do so by running f = randn(3,3,D,K, ¡single¿), which creates K random filters of size 3×3 and dimension D. The second task is to convolve each image with the filters by using the function $y = vl_nnconv(img, f, [])$. This yields an array y with K channels, each $channel_i$ represents the output of filtering the i-th image img_i with the i-th filter f_i. The convolutional blocks have options for: (a) padding the input images before applying filters, (b) selecting the step size of the filter strides over the input image and (c) down-sampling features after the pooling process. The filters are applied in a sliding window manner. Input image channels can be grouped and applied with different filter subsets.

Depending on the architectural choices of the researcher or the intended application, the third task is often spatial pooling or applying the ReLu operation discussed in the paragraph below. Spatial Pooling (also called downsampling) reduces the dimensionality of each feature map while retaining most of the important information [See Figure 4.5]. In MatConvNet, spatial pooling can be applied using the $vl_nnpool()$ function which applies max or sum pooling. The max-pooling computes the maximum response of each feature channel under the filtered areas and sum-pooling computes the average value of the responses instead. The network is organised such that a spatial pooling layer follows

each convolving layer. Any number of network layers can be added in this pattern to make the network deeper. The FC layer from Equation 4.2, denotes a fully connected neural network, SVMs or any capable classifier.

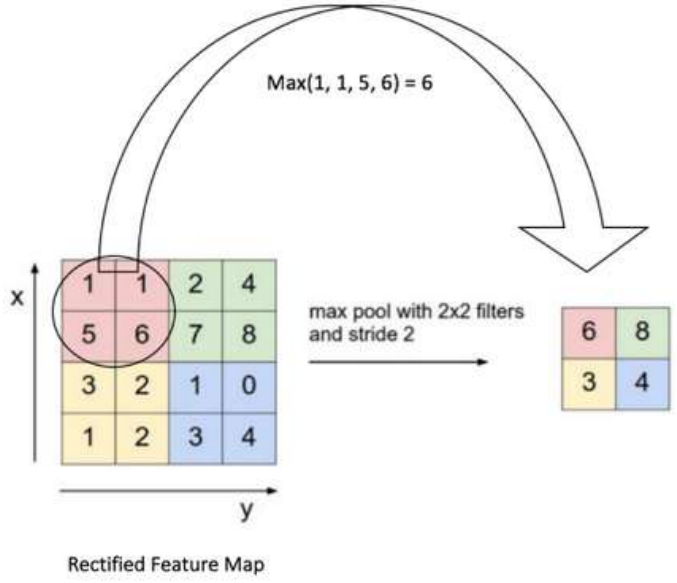


FIGURE 4.5: Max pooling a simple image with a 2×2 patch [69].

A non-linear operation called Rectified Linear Units (ReLu) is traditionally applied after each convolution operation to replace all negative values in the feature map by zero [See Equation 4.2]. The ReLu function can be applied before or after the pooling of features to introduce non-linearity to the CNNs, this allows CNNs to solve non-linear and real-world problems. Other non-linear functions such as the *tan* and *sigmoid* functions are also supported in MatConvNet but the ReLu is known to perform better.

$$[Input] \rightarrow [Conv] \rightarrow [ReLu] \rightarrow [Pool] \rightarrow ... \rightarrow [Conv] \rightarrow [ReLu] \rightarrow [Pool] \rightarrow [FC]. \tag{4.2}$$

The original authors of the CNNs, Lecun et al [70] showed that CNNs can perform better with diverse deeper architectures. For instance, the original *LeNet-1* architecture could classify ten different digits from the MNIST dataset with an error rate of 1.7%, however, the slightly deeper *LeNet-5* architecture could reduce the error rate further to 0.9%.

The CNNs can also perform well even with different architectures. MatConvNet's MNIST example uses an architecture similar to the *LeNet-1* to classify ten different handwritten digits [68]. In the example, the ReLu operation is applied near the end

of the network, just before the fully connected layers instead of applying it after each convolutional layer as done by [71].

The initial layers of the CNNs extract the high-level features from images to study the simple shapes of objects. The complex shapes of objects are discovered at the deeper layers into the network. At the end of the network, the classifier learns specific shapes that can be used to distinguish objects from different classes.

The neurons from the first layers of the network exploit the spatial correlations of features, due to their local connectivity to adjacent neurons. Whereas, the neurons at the end are connected to produce the global shape information of objects. Therefore, the first layers of the network are more susceptible to the local noise clutter found on most gradient images.

The FC layer at the end of the model in Equation 4.2, denotes a machine learning classifier. This can be a fully-connected neural network, SVMs or any capable classifier. The loss function processes the output of the fully connected layers to predict the labels of the classes. A softmax function takes a vector of arbitrary values and squeezes it to values that are between zero and one to ensure that the fully connected layer produces probabilities that sum to 1. Thus providing the ability to evaluate the confidence scores of the classified data. With the softmax function, it becomes possible to also distinguish each predicted class from K mutually exclusive classes, for $K \in \mathbb{Z} \geq 1$. This can be implemented using the procedure $vl_nnsoftmax$ from the MatConvNet library.

4.4 Metrics Used

Having experimented with various types of parameters for each classifier, where the classifiers were trained and validated with varying subgroups of the MIT dataset [66]. On each training scenario, the results from each classifier were compared against previous results to determine the optimal hyper-parameters that produced the best results on each classifier.

Thereafter, the performance of the SVMs was tested against the performances of ELMs and CNNs to output **Classification Results**. The results were saved to an external file with the corresponding technical specifications used. The three classifiers were tested over the same testing criteria. The ground-truth information from the proposed infrared dataset was used to obtain accurate performance results of each classifier. So the answers to the proposed research questions were obtained by evaluating these classification results. Therefore, **hypothesis** H_1 and **hypothesis** H_2 can be tested and proven true if the following cases hold respectively:

- The SVMs and ELMs obtain better classification results over the infrared imaging testing criteria than colour images (RQ_1).
- The deep learning CNNs obtain better classification results than the shallow learning classifiers SVMs and ELMs (RQ_2).

Further performance analysis (PA) was conducted by evaluating the precision and recall rates of the classifiers, where the precision measures how useful the search results are and the recall measure how complete the detections are. The *precision* is computed by taking the number of true positives and dividing it by the total number of positive detections. That is,

$$\rho = \frac{TruePositives}{TruePositives + FalsePositives}, \tag{4.3}$$

whereas the recall is calculated by taking the number of true positives and diving it by the total number of positive samples on the scene, stated by the ground-truth information. Therefore, the recall is,

$$\gamma = \frac{TruePositives}{TotalNumberOfPositives}. \tag{4.4}$$

4.5 Summary

In this chapter, the method of conducting this research was presented. The details of the three visual datasets and the infrared SIGNI dataset used were presented. The data preprocessing step to prevent classifiers from overfitting was also discussed. The feature extraction methods used was reported in detail, where the HOG features and the machine-learned CNN feature maps were introduced as the method for producing meaningful information from raw image data.

The design and implementation of the classifiers were reported. This included brief references to the library environments that were used to allow this work to be reproducible by readers. The chapter also delved into how data will be analysed, how the performance of the classifiers will be evaluated and how the research questions can be addressed. The approach for testing the hypotheses was also discussed.

The next chapter presents the work done for selecting the hyperparameters of each classifier.

Chapter 5

Hyperparameter Selection

5.1 Introduction

This chapter describes the process for selecting the hyperparameters of the three candidate classifiers that were used for addressing this work's supposition. Machine learning models require different constraints, weights and learning rates to generalize on unseen samples. One of the popular methods for estimating optimal parameters in literature is cross-validation [72]. This was applied for the two shallow learning classifiers, SVMs and ELMs. The experimental setup of this work is discussed in Sections 5.2, 5.3 and 5.4.

The objective of the experiments is to evaluate the effects of using an infrared-based dataset when training several human detection systems. To do so, the classifiers were initially trained with colour based human datasets to select the optimal parameters for each classifier. The parameters were pruned further with larger datasets at a later stage. Once the optimal results were obtained, instances of the best-performing candidate classifiers were trained with the infrared dataset and compared to the results obtained with the colour datasets.

The classifiers were only trained with targeted data where interesting objects appear at the centre of the training images. These images are already cropped for training supervised-learning models. Only the performance analysis (PA) stage examined the classifiers' ability to localise and detect people, where large natural-scene images were used.

A cross-validation (with $k = 5$) was conducted on the SVM classifier that was trained with the MIT dataset (denoted as SVM^{MIT}), as well as the SVM^{INRIA} and SVM^{SIGNI}. The training datasets are reasonably small so the classifiers could be trained and tested

in reasonable time. The cross-validation was left-out on the large NICTA dataset as it takes a long time to load several subsets of this dataset. The best parameters for the SVMs and ELMs were selected by iteratively classifying different random sets of the training data and selecting parameters that score the best validation results. In the case of the CNNs, about 20% of the dataset was reserved for validating its performance each time this classifier was trained [Table 5.1]. The general procedure for applying cross-validation is as follows:

Algorithm 1: The general procedure to apply cross-validation.

Data: Partition the training dataset into k folds randomly.

Result: The model's results on selected parameters

1 **for** *each fold* F_i $1 \leq i \geq k$ **do**
2 Reserve F_i as the validation data;
3 Select the remaining k-1 folds as training data;
4 Fit the model on the training data and evaluate it on F_i;
5 Retain the evaluation results and observe the parameters used;
6 Discard the model;
7 **if** *Observed parameters are optimal* **then**
8 Select the parameters and exit;
9 **else**
10 Try different parameters ;
11 Go back to the beginning and repeat line 1;

TABLE 5.1: Training, validation and testing partitions for finding the hyper-parameters of the CNNs.

	Training	**Validation**	**Test**
MIT	60%	20%	20%
INRIA	60%	20%	20%
SIGNI	60%	20%	20%

Merging overlapping boxes is only presentable to the human eye and is mostly useful for presentations. However, this research focuses on the analyses of precision and recall. Therefore overlapping detection boxes are not merged but evaluated separately to measure recognition rate scores. Another concern is that merging overlapping detection boxes through algorithms such as Non-Maxima-Suppression or Mean-Shift algorithms may result in the loss of vital information when either deleting low scoring detections or taking an average of clustered overlapping detections.

This chapter presents the processes and outcomes of training the classifiers and the methods used for selecting the parameters of the final classifiers. These classifiers are

referred to as candidate classifiers because they are the final representatives that will be used to evaluate the significance of using IIR samples. The chapter is divided and organised into three sections that present the selection of hyperparameters for each classifier.

The experiments on support vector machines are discussed in Section 5.2, extreme learning machines in Section 5.3 and lastly, the convolutional neural networks in Section 5.4. The testing criteria for each experiment is stated in detail.

5.2 SVM Classification on HOG features

5.2.1 Introduction

The SVMs were the first classifier to be trained. The process of classifying images with SVMs can be visualised with the flow diagram in Figure 5.1. Suppose an image is fed to a sufficiently trained SVM classifier in vector form [Consider the diagram Figure 5.1 from the left], the SVM will first map this image-vector to a higher dimensional feature space. This feature space was determined during the training process of the SVM, such that the data can be separable with the kernel function in use.

The SVM evaluates the location of a given image relative to the reference images used as support vectors in the same feature space. The evaluated location is used to determine the side of the optimal hyperplane which the image that is being classified belongs. Images that are mapped closer to the positive support vectors are classified as positive. The inverse is also true for images that are mapped onto the negative side of the hyperplane.

The farther the distance from the support vectors, the higher the SVM's classification confidence. Given a trained SVM classifier with a margin formed by the optimal weights $\vec{w}$, the linear combination between a mapped image in the feature-space and the weights can be used to predict the class of this newly inserted image. This is done by evaluating the sign produced by the formula in Equation 2.15.

Initial experiments were done on the MIT pedestrian dataset. The importance of using HOG features over raw grayscale pixels was evaluated by temporarily testing the SVMs with grayscale images. The SVMs managed to classify 71% of the grayscale samples correctly.

Reintroducing HOG features improved the classification rate to 97.0% with the RBF kernel selected for the SVMs and the L1 normalisation step for HOG feature extraction [See Table 5.2, for the pre-experimental results of other kernel-functions tested].

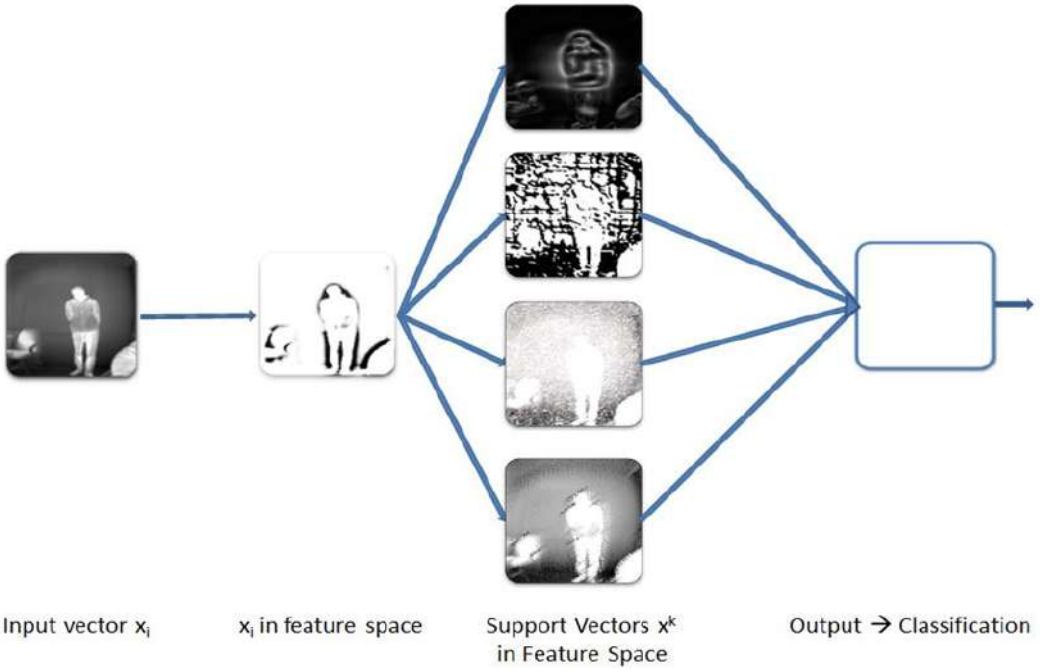


FIGURE 5.1: A diagram visualising the process of classifying images with the SVMs.

Statistical training-data can be re-engineered to improve the accuracy of the classifiers, where false-positives and false-negatives are extensively searched whenever the classifiers are validated such that the hard-sample images are duplicated and re-added to the training set. The repetition of *hard-samples* has been shown to improve the generalisation accuracy of most classifiers, however, this approach would not be suitable for this work since the goal is to discriminate the performance obtained when using different spectral-type images and datasets of the same size.

TABLE 5.2: SVM validation results with different kernels on the MIT Pedestrian dataset.

MIT Pedestrian Dataset [924 pos, 924 neg]	
Variable	**Quantity**
SVM Type: Linear	Recognition Rate 94.3%
SVM Type: Poly (deg=2)	Recognition Rate 74.2%
SVM Type: RBF sigma=31	Recognition Rate 97.0%

5.2.2 Selecting the Candidate Classifier

This section, describes the steps that were involved when selecting the parameters that make up the best SVMs candidate. This candidate will be used for comparing the performance of this classifier to the other two classifiers.

5.2.2.1 Subtracting the Mean of HOG Features

Subtracting the mean of known samples from every sample of the dataset often improves the classification accuracy [1, 2]. When doing so, the common shape of objects in the dataset becomes more apparent on the visualisation of the mean image. For instance, the common shape of people becomes more apparent in the mean image than the varying background elements on most pedestrian datasets. Because the background images consist of random images with inconsistent patterns and such images cannot be visualised as a single class object.

In hopes of improving the performance, the mean of HOG features was produced from the training set and subtracted from the features of every training sample. This was done with a pixel-wise summation of all images, such that the pixel value at position (x, y) on the first image $image_1$ is summed up with all pixels at the same position (x, y) from all the remaining images $image_j$ (for $j \neq 1$). The resulting sum-image was then divided with the total number of samples in the training set to obtain the grayscale mean-image similar to the one in Figure 5.2. Unfortunately, subtracting the mean of HOG features from the training set's features did not improve the performance of the SVM^{MIT}. This was the case for both L1 and L2 normalisation methods.



FIGURE 5.2: The mean image (left) and its HOG visualisation right (right).

5.2.2.2 The effects of normalising HOG features

It became worthwhile to test the impact of normalising HOG features by comparing the performance of the SVMs when normalisation was discarded. Removing the normalisation step improved the training results on the MIT dataset. With the linear kernel scoring a 100% validation rate on the 370 validation samples (evenly split for both classes). This was unexpected because normalisation often improves the quality of

HOG features and is reported to introduce invariances to the change of pixel intensity in images [1]. It is not clear why the normalisation-step yields poorer results than in [1]. Dalal et al [1] used SVM-Light for implementing the SVMs, whereas, the LibSVM library was used in this work. The performance degradation could be a floating-point accuracy bug in the library. The reliance of estimated real numbers by numerical software often leads to nonreproducible results across different libraries. Numerical bugs can also be caused by sensitivity to rounding-off precision, overflow/underflow of floating-points when estimating real-numbers or by random-number generators [73]. The parameters and the maximum number of iterations that were applied when training the SVMs were different from [1].

Considering the two commonly known normalisation methods, the L2 normalisation often gave slight performance improvements of about 1.3% on average than the L1-norm. While the results from both the Linear and RBF kernels were indifferent after the L1 and L2 normalisations, the polynomial kernel benefited from applying the L1 normalisation than the L2 [See Table 5.3, Table 5.4, Table 5.5].

TABLE 5.3: SVM validation results without block normalisation on HOG features.

MIT Pedestrian Dataset [924 pos, 924 neg]	
Variable	**Quantity**
SVM Type: Linear	Recognition Rate 100%
SVM Type: Poly (deg=2)	Recognition Rate 83.7%
SVM Type: RBF (sigma=31)	Recognition Rate 99.5%

TABLE 5.4: SVM validation results with L1 block normalisation on HOG features.

MIT Pedestrian Dataset [924 pos, 924 neg]	
Variable	**Quantity**
SVM Type: Linear	Recognition Rate 96.7%
SVM Type: Poly (deg=2)	Recognition Rate 86.4%
SVM Type: RBF (sigma=31)	Recognition Rate 96.2%

TABLE 5.5: SVM validation results with L2 block normalisation on HOG features.

MIT Pedestrian Dataset [924 pos, 924 neg]	
Variable	**Quantity**
SVM Type: Linear	Recognition Rate 96.7%
SVM Type: Poly (deg=2)	Recognition Rate 61.4%
SVM Type: RBF (sigma=31)	Recognition Rate 96.2%

The effects of the normalisation can only be concluded after applying it on a different dataset. Similar to the MIT dataset, removing normalisation on the INRIA dataset improved the validation rate of all three SVM kernels. The same was applied to the

infrared *SIGNI* dataset, where the L1 and L2 normalisation method produced a training accuracy of about 96% on the 1400 samples from the IIR dataset, yet removing the normalisation increased the validation rate to 99.3% on the same data. Due to the performance degradation on SVM^{MIT}, SVM^{INRIA} and SVM^{SIGNI} classifiers, the normalisation step was discarded during the extraction of HOG features.

5.2.2.3 Different Kernel Types

Different types of kernels were experimented with to select the best performing kernel. The SVM's linear kernel often performed better than the polynomial kernel and the RBF outperformed both kernels. The polynomial kernel of $degree = 2$ only produced competitive results when removing the normalisation step during the extraction of HOG features. In this case, a validation rate of 84.7% was obtained from the MIT dataset. Any degree higher than 2 caused the SVMs not to train at all and classify everything as one of the classes.

Due to consistently high validation rates, the RBF kernel was selected as the kernel of choice for the SVM classifier. In the OpenCV library, the values of σ correspond to the γ values in the gaussian radial basis function shown in Table 2.1. To select the optimal value of sigma, cross-validation over several random training sets was conducted. During the cross-validation, the number of samples from both classes were kept equal. The best validation rates were obtained when $25 \leq \sigma \leq 32$ and the validation rate converged when $\sigma \geq 25$. In Figure 5.3, the cross-validation results for varying values of sigma are shown. The approximate out-of-sample misclassification rate was 0.0005422, thus the generalisation error rate was less than 0.1%.

5.2.2.4 Candidate SVM Classifier

This completes the brief summary of the experimental processes that lead to the selection of the optimal SVMs used in this research. Thus, the candidate SVM classifier used for performance evaluation has the following parameters, a Radial Basis Function kernel, with sigma=31. The next section covers the experimental work done for training the optimal ELMs classifier.

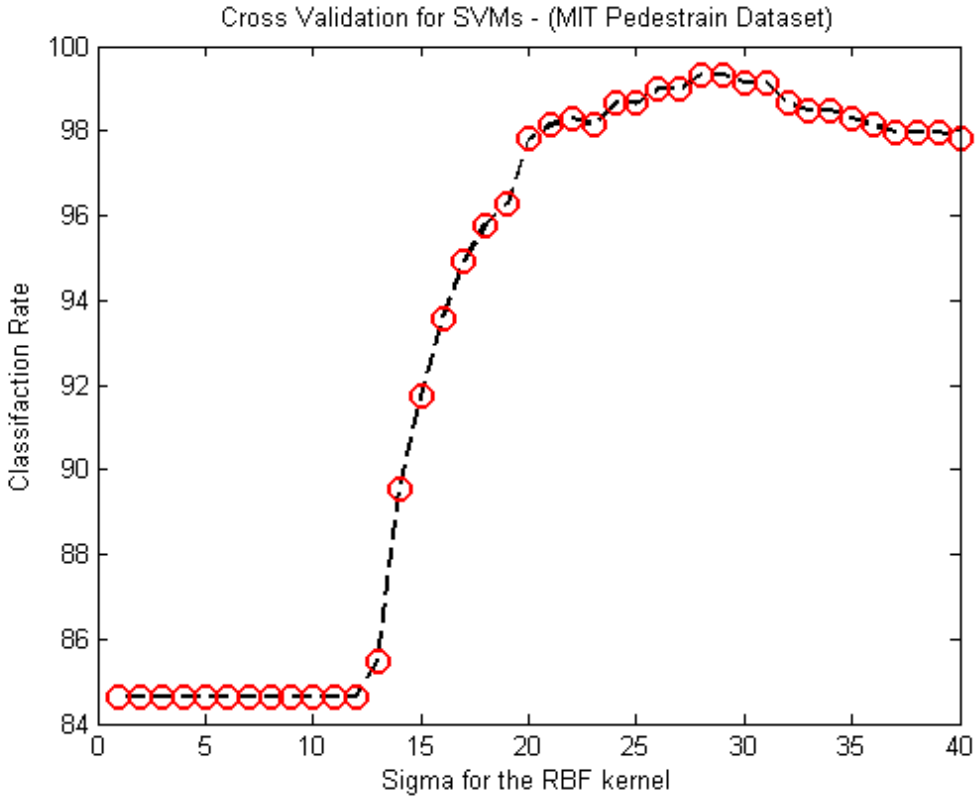


FIGURE 5.3: The cross validation results for selecting the optimal sigma values.

5.3 ELM Classification on HOG features

5.3.1 Introduction

Moving on to the parameter validation experiments on the second classifier, the extreme learning machines. The ELM models are based on artificial neural networks. Similarities between neural networks and SVMs have been debated in the past, but these are different models. With the SVMs, the optimal separating hyperplane that optimises the distance between the two classes of samples is computed. Whereas, neural networks are trained on by finding the network of weights that best predict the classes of samples.

The ELMs were first trained with the smaller MIT Pedestrian dataset as done with SVMs from the previous section. To train ELMs for image classification, the images were first converted to a single column vector. Thereafter, every pixel value was fed to every node on the first layer of the hidden nodes to form a fully connected layer. The pixel values were then multiplied by the random weights of the hidden layer and the bias inputs were added to the result. This linear combination between pixel values and the random weights is supplied to an activation function. The different types of activation

functions that can be used are discussed in the following paragraphs. The depth of the hidden layer should be sufficiently large to classify the image vectors and restricted to prevent over-fitting conditions. The final step is to compute the output matrix of the hidden layer and optimise the output weights. The ELMs are successfully trained once the training error is reduced to a sufficient minimum. Once trained, unknown images can be forwarded to the model to predict their classes through inference.

The challenge with this classifier is in finding the best performing activation function, the sufficient number of network neurons to use and the right bias to the hidden layers of the network. The available activation functions that could be tested were: the *sigmoid* function, *sin*, and *hardlimit* function. Since neural networks obtain their predictive power from connecting a large number of cooperating neurons, the number of neurons used are expected to have an impact on the ELM's validation accuracy. The use of biases is optional, though they can help shift the activation function horizontally on the Cartesian plane. This often improves the learning and classifying ability of neural networks on certain data. The following sections discuss the experiments done on the ELMs classifier to find the optimal hyperparameters.

5.3.2 Selecting the Classifier Candidate

Finding the optimal parameters for the ELMs is necessary to compare this classifier to the other two classifiers in this research. The ELMs set new random weights each time they are trained, therefore the initial weights of the network are may influence to the speed in which the ELMs fit data.

5.3.2.1 The depth of the network

Section 2.11 discussed how neural networks can approximate mathematical functions. Neural networks differ from other machine learning approaches by using multiple layers to solve non-linear operations altogether. Getting back to the debate on the similarity of support vector machines with neural networks, SVMs optimise the output of the linear combinations of data points with a single kernel function. Researchers [74] argue that this process is similar to what neural networks do when they minimise the validation error produced by the activation functions of neurons altogether. The similarity lies in the fact that the neurons also process the linear combinations of the weights and input data. Therefore, SVMs are in essence neural networks that have a constant number of layers [74]. It is well known that functions with fewer oscillations tend to poorly approximate functions with many oscillations. This also applies to neural networks, because networks with more depth tend to be more likely to classify complex data like

images effectively than those with less depth. Noting that this tends to work when neural network models are trained with enough data samples.

As expected, the number of neurons correlated with the performance of the ELMs network. The performance of the network is likely to improve as the depth is increased. For instance, Figure 5.4 shows the training accuracy of the ELMs as the number of neurons increases to an extent that is limited by the size of the training data. The classifier used for generating Figure 5.4 was trained with 924 positive samples from the MIT pedestrian dataset and 3300 negative samples that were generated from INRIA's negative images.

5.3.3 Activation Functions

The *sigmoid* activation function converged faster than the *hardlimit* and *sin*() functions. The *sin*() function required more neurons to converge, requiring 3400 neurons for the error function to converge to an acceptable error rate that is below 3%. The performance order of the activation functions remained the same when training with the larger INRIA and NICTA dataset. The only difference is that the size of the network has to be increased as the number of training samples increases.

The number of neurons was set to 4500 and the ELMs were evaluated on standardised validation sets. Up to 15 000 neurons could be used on a computer with an 8 gigabyte of memory. With standard HOG features, the sigmoid function had the best training accuracy, followed by the hard limit and the sin graph [See Table 5.6 and Table 5.7].

TABLE 5.6: ELMs validation results on normal HOG features from the MIT dataset.

MIT Pedestrian Dataset [924 pos, 3500 neg]	
Variable	**Quantity**
ELM Activation Function: sigmoid	Recognition Rate 99.9%
ELM Activation Function: sin	Recognition Rate 98.9%
ELM Activation Function: hard limit	Recognition Rate 99.8%

TABLE 5.7: ELMs validation results on standard HOG features from INRIA dataset.

INRIA Dataset [3300 pos, 5000 neg]	
Variable	**Quantity**
ELM Activation Function: sigmoid	Recognition Rate 99.2%
ELM Activation Function: sin	Recognition Rate 96.1%
ELM Activation Function: hard limit	Recognition Rate 99.7%

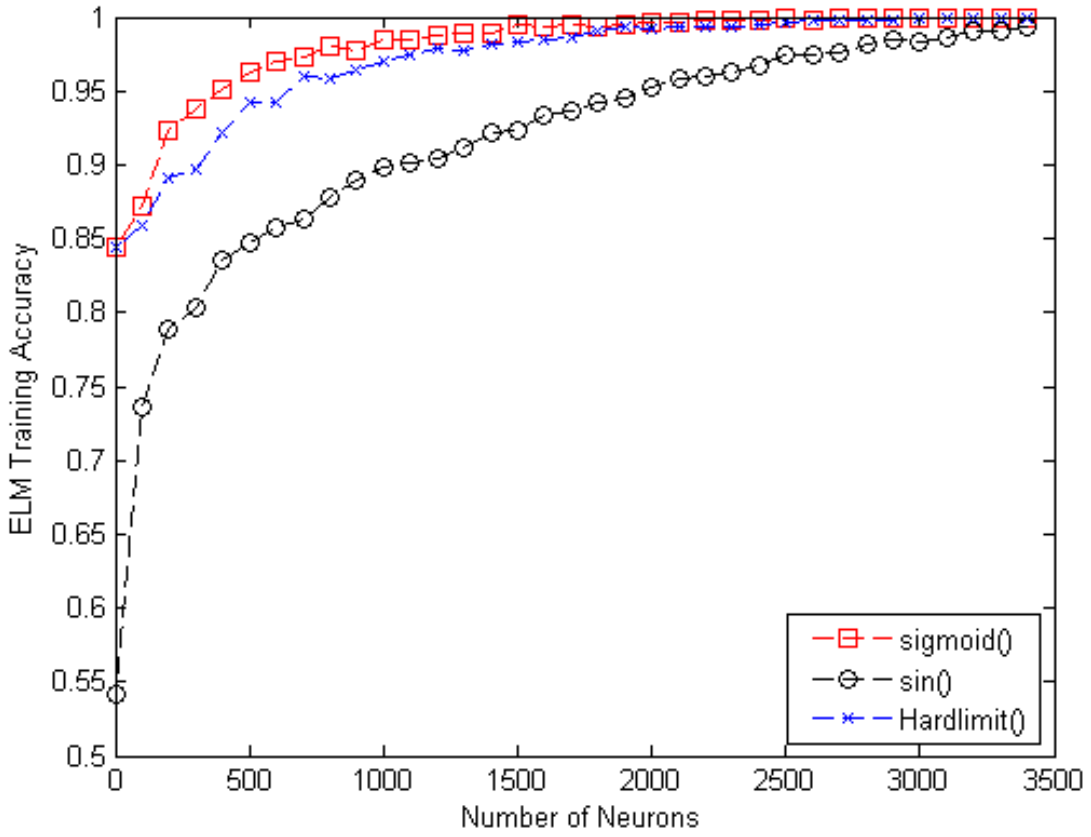


FIGURE 5.4: Training accuracies of different activation functions of the $ELMs^{MIT}$ classifier as the number of neurons increased.

5.3.3.1 The effects of normalising HOG features

To compare the finest details between classifiers, the performance of ELMs was also evaluated on the different block normalisation methods of HOG features. This was done to minimise biases that may be favourable to one of the classifiers. A cross-validation over 5 different subsets of the MIT dataset could not separate the two normalisation methods in terms of training accuracies. So we tried centralising the data by subtracting the mean from each image example as done with the SVMs. Centralising the entire training set did not affect the validation results, perhaps the block normalisation process had already subjected each sample to a $zero^{th} - mean$ by scaling the values to the range 0 to 1. Unlike the SVMs, the normalisation of features was not effective with the ELM classifier.

5.3.3.2 The Biases of the Hidden Nodes

The neural network's ability to learn and generalise can be conditioned to partially apply favouritism to certain classes of data with the use of network biases. When setting a bias, the aim is to shift the activation functions so that the network converges to a

better local minimum. By default, random values between 0 and 1 were assigned to the hidden layer biases. Constant biases to the entire network were set to the first few layers of the hidden layers or the last layer of the hidden layers in hopes of obtaining better validation rates. Slight performance improvements were apparent when setting the entire bias vector to values of 1, with the *sin*() and *hardlimit* activation functions showing more gains.

The performance did not change much after trying other values such as 0.25, 0.5 and 0.75. The same applied when using different biases on different portions of the hidden layers. This demonstrates the ability of ELMs to learn data regardless of the weights used for hidden layers. To determine the exact biases to use, activation functions have to be plotted so that one can observe the effects of shifting the functions. Biases can shift decision functions to be more favourable when classifying specific data, slight performance gains were achievable with this method. The size and quality of training data have a larger effect than setting biases. Nevertheless, the required network biases vary per dataset used.

5.3.4 Candidate ELM Classifier

Unlike the SVMs, the candidate ELMs cannot be assigned with only fixed optimal parameters. Some of the parameters like the number of neurons to use vary with each training dataset because the network depth that can obtain good training accuracies is initially unknown. As a result, such parameters must be learned through trial and error while other parameters like the activation function to use can be fixed.

Due to better performance, the sigmoid function was selected as the activation function of choice for the candidate ELMs. This candidate classifier had promising validation accuracies that were up to 97.6%.

This concludes the preparations of the extreme learning machines classifier. The steps involved in selecting the parameters for the third classifier are discussed in the next section.

5.4 CNNs classification on grayscale images

5.4.1 Introduction

The CNNs have been dominating the performance benchmarks of image recognition challenges lately [26]. The CNNs can autonomously learn the filters required for recognising objects, making it worthwhile to test whether CNNs can perform better when trained with HOG features or grayscale images.

The implementation of CNNs can be quite complex, hence the MatConvNet library was used for implementing the CNNs in this work. The CNNs could not train well initially, despite efforts of trying various parameters and CNN architectures. It turned out that the training problems occurred due to MATCONVNET's inability to train from rectangular images. The images were reshaped into squares and their resolution was reduced to (64×64) to mitigate this. The reduction of resolution helped support a larger number of convolutional filters to be used and essentially allowed the CNNs to be trained and tested at reasonable times. The selected resolution was more than sufficient because CNNs are capable of classifying multiple objects at low resolutions of 32×32.

5.4.1.1 Architecture

Several architectures of convolutional neural networks were considered for finding the ideal architecture for this research, some were small classifiers that identified a few classes and others were large to classify a thousand objects. A similar problem of classifying ten handwritten digits from the MNIST dataset was addressed with the popular *LeNet Architecture* by [75]. The current work evaluates a binary classification problem, the design of the architecture used in this research was also based on *LeNet-5*. But the size and the number of filters used, pooling methods and filter strides were modified to support the larger 64×64 images used.

The initial architectures that were employed are similar to the one populated in Figure 5.5, where the number of filter-banks used on each layer, filter dimensions, strides and the feature-map dimensions are shown. Notice that the library does not distinguish fully connected layers and convolutional blocks at the end of the network, instead fully connected layers are automatically handled internally when the dimension of the output map is $W = H = 1$ [68].

The MatConvNet library provides two types of network structures, the $SimpleNN$ wrapper provides building blocks for implementing simple CNN structures with layers that pass information in a linear direction. The second wrapper $DagNN$ has building

layer	0	1	2	3	4	5	6	7	8	9	10
type	input	conv	apool	conv	apool	conv	apool	conv	relu	conv	softmxl
name	n/a	layer1	layer2	layer3	layer4	layer5	layer6	layer7	layer8	layer9	layer10
support	n/a	5	2	5	2	5	2	4	1	1	1
filt dim	n/a	1	n/a	40	n/a	80	n/a	160	n/a	400	n/a
num filts	n/a	40	n/a	80	n/a	160	n/a	400	n/a	2	n/a
stride	n/a	1	2	1	2	1	2	1	1	1	1
pad	n/a	0	0	0	0	0	0	0	0	0	0
rf size	n/a	5	6	14	16	32	36	60	60	60	60
rf offset	n/a	3	3.5	7.5	8.5	16.5	18.5	30.5	30.5	30.5	30.5
rf stride	n/a	1	2	2	4	4	8	8	8	8	8
data size	64	60	30	26	13	9	4	1	1	1	1
data depth	1	40	40	80	80	160	160	400	400	2	1
data num	100	100	100	100	100	100	100	100	100	100	1

FIGURE 5.5: The CNN architecture used for human detection.

blocks for organising CNNs into more complex acyclic graphs. The *SimpleNN* structure has been shown to classify multiple objects from larger image-datasets like the ImageNet and CIFAR [26]. Therefore, the CNN architecture in Figure 5.5 is sufficient for a binary classification problem.

Even though convolutional neural networks are trainable with colour images, all colour datasets were converted to grayscale images to match the number of channels with the infrared images. Additionally, this limits the required computer-memory, adds support for more convolving filters at the convolutional-layers and lessens the time required for training the CNNs.

The split-ratio between the training, validation and testing sets remain at $60 : 20 : 20$ throughout this research. The CNNs were first trained with 3000 positive and 3000 negative samples from the NICTA dataset. Without any revision of parameters, the CNNs obtained a validation rate of 91.67% on 600 positive and 600 negative images. This was done without centralizing the data by subtracting the mean of training samples from each sample.

5.4.2 Choosing Parameters

When selecting parameters for CNNs, the process of selecting the number of layers to use, convolutional layers, the right size of filters to use, the amount of stride and padding is often not trivial initially. There is no single approach to follow because such parameters are largely dependent on the type of data used. Data can vary in dimension, dynamic range and complexity. The type of image processing methods that are required for extracting information in images also varies per application. The recommended

approach is to first study the datasets and find the right combination of parameters that can abstract image information at a reasonable scale.

The initial parameters that were used for training the CNNs were as follows, the filter sizes were set to 3×3. A stride of a single-pixel was selected for the convolutional layers and a stride of 2-pixels was selected for the pooling layers. The stride controls the step sizes of filters when convolving around the feature map. In this work, the filters were shifted by a single pixel each time to consider all the information from feature maps.

Zero-Padding adds rows and columns of zeros on the edges of features maps and is often used for retaining the information at the edges. The information at the edges is retained by keeping the spatial output dimensions equal to the input dimensions. Most people in the datasets used are often positioned at the centre of the images, perhaps, this could be the reason padding did not improve the performance.

The number of random filters is traditionally increased during each consecutive layer as the spatial pooling layer down-samples feature maps [Figure 5.5]. The smaller resolution of feature maps allows more filters to be used. This is apparent in Figure 5.5, where the feature map dimensions were reduced from (64×64) to (30×30) after applying spatial-pooling in layer 2. Note that the size of the filters was reduced whenever the dimension of feature maps was less than or equal to the dimension of filters, this assures filters can fit within feature maps.

5.4.2.1 Feature Map Down Sampling

Conventionally, the max-pooling approach is applied such that the information concerning the exact location of features is made less significant than the location of features relative to each other. This helps with capturing the shape information of objects. The different downsampling methods showed slight performance outcomes, taking the "max" instead of the "average" at the pooling layers increased the validation rate to 94.83% with the network converging around twenty epochs.

The convolutional and pooling layers were applied iteratively, where the dimension of feature maps was reduced to 4×4 and the number of filters was increased to 400 filters after each iteration. A *ReLu* operation was applied for introducing non-linearity to the classifier. It is customary to apply a non-linear activation function right after the convolutional layers. Unlike the $tan(h)$ and *sigmoid* functions, the network trained faster with the *ReLu*. After the ReLu layer, the feature maps were fully connected to obtain a high-level view of features. Eventually, a loss layer was applied to define how the network was penalised for making wrong predictions. The error obtained is then used to re-train the network.

5.4.2.2 Centralization of Data

The centralization of data samples had an unexpected outcome, the CNNs converged faster and became more prone to overfitting. The centralization caused a jittered loss curve during the training process, whereas the removal of centralization produced a steady convergence towards the objective error. Slight performance gains were also apparent and the training process took longer as expected. So far, the best results were obtained after 40 epochs. An epoch represents a single forward pass and a single backward pass of all training samples to the network.

Oftentimes, the network's performance decreased exponentially after the 40th epoch on the validation data. After 40 epochs, the training accuracies kept on increasing, indicating an overfitting classifier. More performance gains can still be achieved by enlarging the depth of the network and considering various patterns of padding at the initial stages of the network to preserve the low-level information during the early stages of feature extraction. Padding would also allow deeper network structures.

5.4.2.3 Depth of the Network

The depth of the network can be adjusted by selecting the number of filters to use for each convolutional layer. The introduction of graphical processing units enabled deeper networks to be tested and allowed additional data samples to be used.

Since denser CNNs require longer epochs to train, the number of epochs was increased whenever the network's architecture was made denser. The decision towards terminating the training process was dependent on the convergence of the minimum error and the avoidance of overfitting.

On the architecture in Figure 5.5, the number of filters on all convolutional layers was a multiple of the number of filters at the initial layer. The number of filters was set as follows: if K is the number of filters on the first convolutional layer, the second layers had $K \times 2$, the third $K \times 4$, then $K \times 10$ at the second last layer. $K \times 2$ was set for the last convolutional layer since this research addresses a binary classification problem. Increasing K to 100 lead to slight performance gains and increased the validation rate to 96.67%. Setting K to 128, increased the performance by just 0.16% and no performance gains were observed when $K \geq 192$ on smaller datasets.

The CNNs scored 98.38% when increasing the number of samples to 5000 for each class. Using a larger dataset of 15000 samples increased the validation rate to 98.92%. For the candidate classifier, $K = 192$ was selected for the initial filers.

5.4.2.4 Regularization methods: Overfitting Avoidance

The most intuitive regularization method for neural networks is the avoidance of over-fitting. The easiest way to perform this is halting the training process before the convergence of the classification error. This approach should be done with caution to avoid limiting the network from learning. Hence a balance between training accuracy and the amount of over-fitting should be maintained.

In this work, the training process was halted whenever the training error converged to a reasonable minimum. The epoch that scored the highest testing accuracy was selected. Allowing further ten epochs drastically reduced the classifier's ability to generalise unseen data.

Another method for avoiding overfitting is limiting the quality of parameters used. In the case of neural networks, the depth of the network can be limited to restrict the predicting power of the network. This was considered whenever overfitting occurred. Other methods such as the weight decay, dropout, stochastic pooling, L1 and L2 regularisation also exist but were not tested in the current work.

5.4.2.5 Learning rate and momentum

Convolutional neural networks have more hyper-parameters to consider than the generic neural networks, but the rules for adjusting the learning rates and the momentum for stochastic gradient descent remain the same. A high learning rate will decay the error-loss function faster and the function may converge to a local minimum. This is often apparent with a jittered convergence curve. The higher the learning rate the harder it becomes for the network to settle to an optimal state.

The learning rate and the momentum of the network were reduced whenever poor training accuracies were observed. While experimenting with various CNN parameters, the learning rates were kept within a range of 0.01 and 0.0001 to train and validate the CNNs at reasonable times. For the final candidate classifier, a learning rate of 0.00001 was used. The threshold value that was used with our system is 0.1×10^{-8}, this prolonged training process for several days. In spite of longer training times, this did not guarantee better validation rates and the loss function would fail to converge at times.

5.4.3 Candidate CNN Classifier

Just like ELMs, the candidate CNNs cannot be assigned with fixed optimal parameters. The design of the architecture and parameters such as the downsampling options, learning rate and momentum highly depend on the datasets used. The required depth of the network is also unknown initially. Such parameters are learned through trial and error whenever the CNNs are trained.

The final parameters for the candidate CNNs were as follows: the number of initial filters used was $K = 192$. The learning rate and training momentum were set to 0.0001 and 0.5 respectively, the preferred spatial pooling method was "max pooling", the minimum number of epochs was set to 40 but often increased to a maximum of 192 epochs until the training error converged to an acceptable minimum and the validation accuracy was sufficient. From this point, the task that followed was to compare the performance of this candidate CNNs against the two shallow learning classifiers that were trained earlier. The following chapter will present the comparison and analysis of test results between the classifiers.

5.5 Summary

The experimental setup of this work was presented, where the steps involved in finding the candidate classifiers were discussed. The characteristics of processing images for binary classification were discussed for each classifier. Various critical hyper-parameters that are crucial for each classifier were also discussed.

The final performance evaluation of each candidate-classifier on unknown examples is discussed in the next chapter, where the classifiers are tested over the same scenarios to address the research questions and hypotheses [Section 1.2]. The next chapter presents the overall findings of this work and addresses the supposition of this work [Chapter 6].

Chapter 6

Analysis of Results

The previous chapter performed the necessary experiments to prepare the candidate classifiers for the final performance evaluation. This chapter evaluates the overall performance of the classifiers and reports the findings of this work. Section 6.1 presents a performance evaluation of the classifiers over highly skewed datasets with fewer positive samples. Evaluation on less skewed datasets with more positive-sample variance is presented in Section 6.2 and results from the ground-truth assessment over the human detection problem are presented in Section 6.3.

The outcomes of Section 6.1 highlight the ability of the classifiers to learn from fewer positive samples and skewed data. The localisation experiments test the classifiers' ability in solving the detection problem over difficult natural-scene testing scenarios. The parameters of the candidate classifiers that were selected prior remain unchanged for the rest of the experiments onwards.

6.1 Evaluation A: Highly Skewed Datasets

To address the research questions, the classification performance of the classifiers was evaluated by comparing the average classification rates of the three colour datasets against the average scores of the three infrared datasets. To do so, the positive samples in each dataset were reduced to match the size of the dataset (MIT) with the least positive samples. The generated INRIA negative samples were used for all visual datasets. The number of visual negative samples was limited to the number of negative samples from the infrared SiGNI dataset. Therefore, 924 positive samples and 5462 negative samples were selected for each dataset. The split ratio of the training, validation and testing sets is always set to $60 : 20 : 20$ respectively.

TABLE 6.1: Classification results of the SVMs over highly skewed datasets.

Datasets (924:5462)	**Classification Results**
INRIA	96.4%
MIT	98.6%
NICTA	98.2%
Infrared SIGNI (A)	98.0%
Infrared SIGNI (B)	98.4%
Infrared SIGNI (C)	98.6%
Avg Classification Rate (Visual)	**97.73**%
Avg Classification Rate (Infrared)	**98.33**%

TABLE 6.2: Classification results of the ELMs over highly skewed datasets.

Datasets (924:5462)	**Classification Results**
INRIA	82.7%
MIT	90.6%
NICTA	89.1%
Infrared SIGNI (A)	95.7%
Infrared SIGNI (B)	96.9%
Infrared SIGNI (C)	95.0%
Avg Classification Rate (Visual)	**87.46**%
Avg Classification Rate (Infrared)	**95.86**%

To match the three visual datasets, the positive samples from the infrared SIGNI dataset were partitioned into three sub-datasets. Each of the partitions had 462 positive images and their reflections to match the total of 924 samples from the three visual datasets.

Slight performance gains were observed from the use of infrared samples over the use of visual samples. Starting with the SVM candidate classifier, the average classification rate of the infrared samples was 0.6% higher than the visual samples [Table 6.1].

The performance of the candidate ELMs classifier over highly skewed datasets was relatively low than SVMs for both spectral image data types [See Table 6.2]. The performance gain of using infrared images was more apparent with the ELMs, as the classifier obtained a gain of 8.4% with infrared images. However, the ELMs' ability to learn and classify visual images was poorer than the SVMs. The classification rate of the ELMs over the infrared data was 95.86%, compared to the classification rate of about 98.33% that was achieved by the SVMs, a 10% difference in favour of the SVMs over the classification of visual data was observed. In terms of the significance of using infrared images over visual images, the supposition (H_1) holds for both SVMs and ELMs thus far.

The performance of the CNNs over visual images was on par with the SVMs [Table 6.3]. Just like the other two classifiers, the CNNs also showed better results with infrared

TABLE 6.3: Classification results of the CNNs over highly skewed datasets.

Datasets (924:5462)	**Classification Results**
INRIA	97.17%
MIT	98.59%
NICTA	96.39%
Infrared SIGNI (A)	99.61%
Infrared SIGNI (B)	99.84%
Infrared SIGNI (C)	99.21%
Avg Classification Rate (Visual)	**97.38**%
Avg Classification Rate (Infrared)	**99.55**%

images. The CNNs obtained the best overall classification rate (supporting hypothesis H_2), with the average classification rate over the three visual datasets at 97.38%, while an outstanding score of 99.5% was obtained with the infrared images. This reflects the dominance of CNNs in image classification tasks.

It is important to note that these results only represent the basic evaluation of the classifiers, as the datasets used at this stage have relatively few positive samples. The performance evaluation of the classifiers can be more precise with larger datasets. Therefore, in the next subsection, the classifiers were tested with less skewed datasets that have more positive samples to increase the diversity of samples and reduce the chances of overfitting.

6.2 Evaluation B: Less Skewed Datasets

In this section, the hypotheses were evaluated over less skewed datasets with more sample variance. To do so, samples from the SIGNI dataset were combined to a total of 3832 positive samples and 5462 negative samples and compared to the INRIA and NICTA datasets of the exact size. The MIT dataset was discarded for this task considering that it only contains 924 positive images. To balance the size of datasets, the 3610 positive samples from the INRIA dataset were supplemented with 222 non-used samples from the NICTA dataset to obtain 3832 positive samples.

When comparing the two visual datasets alone, the classifiers obtained better results with the NICTA dataset. This may be attributed to the consistent positioning and scale of people on the NICTA dataset. The advantages of using infrared samples were apparent from the first training epoch of the CNNs, the training error was reduced to just 0.132%, compared to the 0.278% achieved by the visual datasets. A validation accuracy of 0.019% was obtained just after 6 epochs with the infrared dataset, compared to the

minimum of 50 epochs that were required by the visual datasets. The CNNs converged faster when training with infrared samples, at just 10 epochs in some cases.

TABLE 6.4: Classification results of SVMs on larger datasets.

Datasets (3832:5462)	**Classification Results**
INRIA	94.1%
NICTA	97.8%
Avg Classification Rate (Visual)	**95.95**%
Infrared SIGNI (Entire Dataset)	**98.9**%

TABLE 6.5: Classification results of ELMs on larger datasets.

Datasets (3832:5462)	**Classification Results**
INRIA	87.9%
NICTA	92.4%
Avg Classification Rate (Visual)	**90.15**%
Infrared SIGNI (Entire Dataset)	**96.4**%

TABLE 6.6: Classification results of CNNs on larger datasets.

Datasets (3832:5462)	**Classification Results**
INRIA	97.58%
NICTA	98.43%
Avg Classification Rate (Visual)	**98.01**%
Infrared SIGNI (Entire Dataset)	**99.67**%

To examine the results on larger datasets, the average classification results of the two visual datasets were compared to the results obtained with the entire infrared dataset. The hypotheses remained true for all classifiers, with the SVMs and ELMs benefiting the most. The SVMs had a 2.95% advantage and the ELMs reaching over 6% advantage thus supporting the hypothesis H_1. The CNNs already had relatively high classification rates with the visual datasets, resulting in a slight performance gain of 1.66% when using the infrared samples which supports the hypothesis H_2. Notice that the performance of the ELMs improved with larger training datasets.

The performance results over the SIGNI dataset can be attributed to the high contrast between the people and background in most samples. While human subjects can be visible during day and night without any illumination to the environment with infrared images, it is worthy to note their limitations. The classifiers would struggle predicting the presence of people in situations where the background temperatures are similar to the temperatures of people.

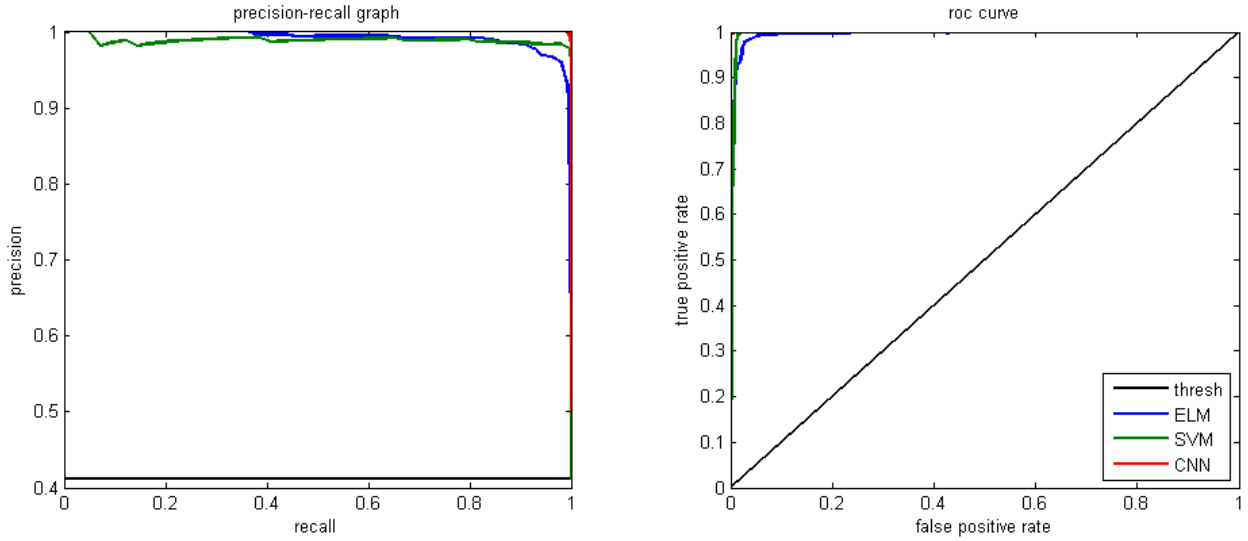


FIGURE 6.1: The precision and recall graphs of the classifiers.



FIGURE 6.2: Sample images from the SIGNI dataset.

The precision and recall graphs of the classifiers over the SIGNI testing set are shown in Figure 6.1. The graphs correspond to the performance results obtained with the larger datasets. The goal of any typical classifier is to reach the top-right corner of the precision and recall graph. Whereas, on the ROC graph the goal is to reach the top-left corner of the axis.

The classifiers performed relatively well on the testing samples [Figure 6.1]. However, measuring the performance of classifiers with discrete images from the same dataset may lead to misleading results because samples from the same dataset may be similar in nature [Figure 6.2]. For instance, the people may be scaled to the same size and the images may be photographed from similar environments. Hence it was necessary to further test the classifiers over the human detection problem with random infrared and visual images from the internet. In the next section, the best performing classifier was selected for evaluating the hypotheses over the human detection problem.



FIGURE 6.3: Detection results of the joint SVM+CNN classifier.

6.3 Detection and Localisation Performances

This section analyses the localisation performance of the classifiers. The ground-truth information consists of a collection of annotated regions from over 200 large images of about 640×480 resolution on average. The ground-truth regions were manually labelled and their coordinates were stored in XML files, to ensure effective evaluation of the localisation performances.

Each classifier was passed over a sequence of large images in a sliding window manner. During each image pass, the sliding window was convolved over a dense scale pyramid of the image. A classified ROI is considered correct if it overlaps at least 50% of any ground-truth region. The number of true-positives, false-positives, true-negatives and false-negatives was measured for results and analysis purposes. The ground truth algorithm was first tested over smaller ground-truth testing set to ensure that the implementation worked correctly, where every output was monitored manually. This testing set contained 15 large images that were converted into multiple-scale pyramids. Errors found in the algorithm were corrected until the algorithm was confirmed to work as intended. The



FIGURE 6.4: Typical ROI's with complex partial occlusion of people that are observed by the sliding window classifier [1, 7].

localisation experiments were also monitored occasionally to ascertain that the output was correct.

6.3.1 Localisation Experiments

The localisation experiments test the generalisation of the classifiers over tougher testing conditions. With the classification experiments in Section 6.1 and Section 6.2, the classifiers were given images that either contained a person or not. The images that contained people, had subjects that filled at least 80% of the images and the subjects were mostly positioned near the centre of the images. These conditions made it easier for classifiers to distinguish between the background images and those that contained people, which subsequently lead to good classification results.

Whereas the localisation experiments expose the classifiers to more complex inputs and more variety of partially occluded people on the tested sub-images. When the detection problem is converted into a classification problem, the complexities that occur are as follows: the sliding window may extract images that contain partially occluded people like in Figure 6.4. The images may only show the following sections of the human body: torso only, head and shoulders only, a single leg or both legs, mid-waist only, an arm or shoulders. This phenomenon leads to interesting results.

The large ground-truth images that were used varied in size and were not resized to avoid irregular shape deformation to the photographed people. This caused the number of images that were tested between the colour and infrared ground-truth datasets to also differ. With this in mind, the total number of regions-of-interest tested was considered during each evaluation of the classifier to obtain non-biased comparisons between the two

spectral images. The resolution of the ROIs remained 128×64 and the vertical/horizontal strides of the sliding window were 16 pixels.

The infrared ground-truth dataset contained 113 images and over 233 labelled ground-truth regions, enough to produce 167 254 of testable ROIs. Because of the availability of colour images from the internet and the INRIA dataset, the visual ground-truth dataset contained 262 large images and over 617 labelled ground-truth regions, resulting in over 358 310 testable ROIs (twice the amount of the infrared testable ROIs).

6.3.2 Localisation Results

The best performing classifier (CNNs) during the classification experiments was selected for conducting the challenging localisation experiments. For these experiments, the same architecture used for implementing the CNN in the MatConvNet library was re-implemented in Caffe and NVIDIA Digits platform for the localisation experiments. The NVIDIA DIGITS platform provides a much more simplistic user-interface for designing, training and visualizing deep neural networks. Caffe is an open-source deep learning framework that has the implementation of CNNs for classification and detection tasks. One of the reasons for migration was the speed at which the CNNs could be trained and tested with Caffe's C++ programming interface.

The results are reported with the total number of regions tested, the number of ground truth visits, true positives, false positives, true negatives, false negatives, and the precision and recall rates of each testing scenario. The testing scenarios differ by the training dataset that was used and the corresponding spectral type of images. The results are shown in Table 6.7.

The overall classification rate on the infrared testing scenario was 4% higher than the average classification rate over the colour testing scenarios. But looking at classification rates alone can be misleading. For instance, the CNNs that were tested on the colour testing scenarios had a higher precision rate than the one tested with the infrared testing scenario. However, the precision rate only highlights the precision of the classifier solely on the images it retrieved as positive. This does not show the precision over the total amount of positive ground-truth data. The recall rate of the classifiers highlights this as discussed later on.

For simplicity, let us refer to the two CNNs that were trained with colour images as $Classifier^{INRIA}$ and $Classifier^{NICTA}$ and the one trained with infrared images as $Classifier^{IIR}$. Studying the recall rates shows that both the $Classifier^{INRIA}$ and $Classifier^{NICTA}$ performed poorer than the $Classifier^{IIR}$. The low values of 0.043 and

0.077 indicate that the classifiers that were trained with colour images failed to retrieve a remarkable large number of positive ground-truth regions. The $Classifier^{INRIA}$ only retrieved 8 054 while $Classifier^{NICTA}$ only retrieved 14 364 from the total 187 447 of ground-truth regions that were visited by each classifier. Instead, the $Classifier^{IIR}$ was able to retrieve 48 214 of the total 76 479 ground-truth visits. This implies that the two colour classifiers only retrieved an average of 5.98% of the ground truth ROIs as positive samples, whereas the infrared classifier was able to retrieve 63.04% of the total ground truth regions. This is a remarkable difference in favour of the infrared classifier.

The number of false positives by the $Classifier^{IIR}$ was quite high and the number of false negatives on the colour classifiers was also high. It appears that the $Classifier^{IIR}$ was more likely to classify data as positive than the other two classifiers and the colour classifiers were more likely to predict data as negative. The high amount of false positives by the infrared classifier may be attributed to the following reasons. The training images had no instances where people had lower temperatures than the background settings while some of the ground-truth testing scenes have such cases.

For the localisation experiments, samples from the infrared testing-scenario were supplemented with infrared images from the internet to lessen the gap between the number of visual samples and infrared samples. Some of the infrared images from the internet were poor in quality (low resolution, low sharpness or blurred). Most of the infrared-images that were recorded for this work during a sunny day in May 2017, had human temperatures that blended with the background. Whereas, samples from the visual testing-scenario had fewer imperfections (better resolution and the subjects were clearer).

The traditional method of assessing the performance of sliding-window detectors appears to have some flaws, as discussed in the next subsection.

TABLE 6.7: Object localisation results of sliding classifiers: $Classifier^{INRIA}$, $Classifier^{NICTA}$ and $Classifier^{IIR}$.

Criteria	**INRIA**	**NICTA**	**SIGNI**
Total regions tested	358310	358310	167254
Total ground truth visits	187447	187447	76479
Number of True positives	8054	14364	48214
Number of False positives	4319	7696	50129
Number of True negatives	166544	163167	40646
Number of False negatives	179393	173083	28265
Classification Rate	**48.7%**	**49.5%**	**53.1%**
Precision	**0.651**	**0.651**	**0.490**
Recall	**0.043**	**0.077**	**0.630**

6.3.2.1 Ground Truth Validation flaws

The requirement for detected windows to overlap ground-truth regions by at least 50% has many flaws. It is easy to see that with any step-size used for the sliding classifier, the 50% requirement may penalise classifiers for recognising partially visible objects. The question that arises is: are classifiers not supposed to recognise partially occluded objects? Ideally, a robust classifier should recognise a human even if half of the body is visible. It is concerning that most object detection benchmarks follow this rule and are penalising classifiers unfairly.

To address this problem, the ROIs that were misclassified and their corresponding ground truth labels were saved into files for reassessment. It appears that a large number of images with noticeable full bodies of people are being marked as negative samples due to the strict 50% rule [Figure 6.5]. Some images would show a person positioned at the far horizontal sides of ROI and some would show relatively small people within the ROI. This information should be enough to recognise people, as robust classifiers should handle slight translations and occlusions.

Ground-truth validators should make provisions for detection windows that fall short of meeting the required overlap. It is evident that the classifiers are identifying vital patterns in images, but are unfairly penalised for doing so.

6.4 Results and Discussion

Human beings can easily identify people that are highly occluded, even from a picture that only shows a head, hand or boots [Figure 6.4]. If future classifiers are expected to possess such cognitive capabilities, then newer ways of evaluating their performance should be established. The use of heat maps may address this problem, where detections are penalised the further they are from ground-truth window. Meaning detections that are closer to the ground-truth region become favourable. Unlike semantic/mask segmentation, it is difficult to determine the inclusion of objects in rectangular windows without pixel-level ground-truth information. Therefore, heat map scoring/penalising methods may mitigate this problem.

Another similar problem is penalising classifiers for recognising reflections of people on reflective materials. Just like visible light, heat radiation can also be reflected on surfaces that have high specularity. Consequently, some of the image samples showed spectral ghostly reflections of people on doors and other reflecting materials [Figure 6.6]. Regrettably, these reflections were not considered during the annotation of the ground-truth information since they were barely visible.



FIGURE 6.5: Some of the images that were classified as negative samples by the strict 50% overlap requirement when validating object detection ground truth information [1, 7].



FIGURE 6.6: Heat reflections were not considered for ground-truth information, yet some were classified as positive by the classifier.



FIGURE 6.7: Some of the results from region-based CNNs detector.

As expected, the classifiers would recognise these reflections as positives at times. Which is good, because classifiers are mathematical models that see tensors (i.e multi-dimensional arrays) instead of images as animals do. Because of the time constraints of this thesis, we were unable to include these ghostly reflections to the ground-truth information. Nevertheless, they would barely affect the results because only a few images have reflections in the testing set.

Other CNN network models such as the AlexNet and GoogleNet were experimented with to try and improve the results. However, the datasets used in this work were too small to successfully train the AlexNet model without transfer learning. The larger GoogleNet model required more system memory than the $8GB$ memory on the system used for training the CNNs. Some of the recent CNN architectures can perform both detection and classification of objects in images. The choice of a sliding window detector was selected in this research to accurately assess the performance of all classifiers since two of the classifiers used in this research cannot perform detection on their own. Transfer learning was strictly not applied to discriminate the real performance differences from applying different spectral-band images. This concludes the rationale for selecting the older Lenet 5 architecture for the CNNs classifier.

Object detection is a challenging task to solve than image classification problems. With object detection, computers are not expected to just classify the context of the images but are required to predict the location of the bounding boxes that enclose the detected objects in large images. Region-based CNNs have superior performance than the sliding classifiers used in this research [76] and report less false positives due to the bounding-box regressor which best fits a rectangle onto regions that contain detected objects. The following images in Figure 6.7, show the results of applying a single-shot detector (SSD) on large images. This SSD detector employed a VGG architecture for the base convolutional layers and was trained with the 2015 COCO dataset.

The best two performing classifiers (SVMs and CNNs) were selected to form a robust cascade of classifiers that could be compared to the SSD detector. That is, a region-of-interest is only predicted to contain a person if and only if, both SVMs and CNNs classify it as positive. Through this joint SVMs+CNNs classifier, similar experiments of finding people in large images were conducted and the detection results are shown in Figure 6.3. The cascade of classifiers had fewer misclassifications than the other sliding-window-based detectors, however, the recent region-based CNNs detectors are more accurate for detection tasks and only require a single model to be trained. Other region-based CNN architectures that are capable of robust human detection are the YOLO, Fast RCNNs, SSD detector and DetectNet.

6.5 Overal Results

The average performance gains that were obtained by each classifier can be calculated by taking a sum of performance gains from both highly skewed (Experiments A) and less skewed (Experiments B) dataset experiments and dividing it by two. This makes it easier to interpret the overall results in Table 6.8. For instance, summing the performance gain of the SVM classifier on highly skewed datasets (0.60%) with its gain on the less skewed datasets (2.95%) and dividing this by two yields the overall performance gain of $\frac{3.55\%}{2} = 1.78\%$ for this classifier. Doing the same calculation for the ELMs and CNNs shows overall performance gains of $\frac{14.40\%}{2}$ and $\frac{3.83\%}{2}$ respectively.

TABLE 6.8: Overall classification performance gains per classifier.

	Small Datasets Gains	**Large Datasets Gains**	**Overall Gains**
SVMs	0.60%	2.95%	**1.78%**
ELMs	8.40%	6.00%	**7.20%**
CNNs	2.17%	1.66%	**1.92%**

The ground-truth results justified this work's hypotheses (H_1 and H_2) and show that the classifiers are more likely to recognise the appearance of people in images with reduced noise levels like infrared images. The low recall rates of 0.043 and 0.077 (from CNN^{INRIA} and CNN^{NICTA} respectively) indicate that the CNN classifier that was trained with colour images failed to retrieve a remarkable large number of positive ground-truth regions. This equates to just above 5.98% of positive ground-truth boxes that were retrieved, versus a 63% of ground-truth regions that were retrieved by the infrared classifier. A high recall rate indicates a classifier that returned most of the relevant results, whereas a high precision rate indicates a classifier that returned considerably more relevant results than irrelevant ones. The ground-truth results from Table 6.7 show that the precision and recall rates over infrared testing scenarios were more balanced compared to the results on visual testing scenarios (when using a precision rate of 0.5 as a reference point), where the classifiers had a higher precision with a substantially much poor recall rate.

Throughout all experiments, better results were obtained with the use of infrared images than the use of visual images by the two shallow learning classifiers (SVMs and ELMs) as well as the deep learning CNNs. Furthermore, the CNNs performed well than the shallow learning classifiers, making the overall results favourable to the supposition of this work. The findings of this research are concluded in the next chapter and future work is proposed.

Chapter 7

Conclusion

This work has shown the advantage of using infrared imaging for human detection. Several visual datasets were collected and compared with the newly created dataset that is publicly available online [7]. The influence of using infrared images was studied over six smaller subsets of the datasets, as well as larger datasets. Several classifiers were intensively trained to determine the optimal parameters that could be used as a standard for evaluating the extent of performance gains between the two spectral samples. The use of similar parameters reduced possible biases between the two data sample types.

The hypothesised supposition was shown to hold for all classification test cases and all three classifiers. As a result, we can conclude that the use of infrared images has the potential to improve human detection results. Furthermore, the results show that deep learning models may also benefit from the same supposition and better classification results were obtained with deep convolutional neural networks than the shallow learning SVMs and ELMs. The benefits were also apparent during the training process, as the CNNs converged faster with the infrared dataset. On the large dataset experiments, the classifiers (SVMs and ELMs) that used hand-engineered features benefited more than the classifier (CNNs) that autonomously learned the desired filters for feature acquisition.

The localisation experiments addressed an imbalanced binary classification problem, where one of the classes (negative background samples) had the overwhelming majority of the data samples. In such cases, precision rates are not a good measure because they can be easily obtained by any classifier that is biased to the overwhelming class. Instead, the recall rate metric becomes a better measure as it shows a model's ability to find relevant samples from a testing scenario that has considerably more irrelevant samples.

The results from the localisation experiments were favourable to the infrared testing scenarios as the classifier trained with infrared images showed a higher recall rate and a balanced trade-off between the precision and recall rate. Compared to the visual testing scenarios, the classifiers had higher precision rates and substantially poor recall rates proving to be biased to the negative class. The recall rates showed that the classifiers trained with colour datasets failed to retrieve about 95% of the positive ground-truth regions versus about 36.96% from the classifier trained with infrared images.

It is clear that a more precise outcome of this research would be possible should the datasets from both spectral types be captured in synchronously. In addition, the datasets should be captured at different times of the day and different environments to improve the generalisation of the classifiers.

7.1 Future Work

Our current and future work is on developing other methods for enhancing the quality of extracted edge information. Based on prior experiments, we propose the unification of similar colours in local regions for lessening the amount of low-frequency edges on foreground objects. This may improve the quality of HOG/CNN features. To implement this method, we tried *"cartooning"* the images using the Selective Gaussian Blur filter from the cross-platform image editor GIMP. Sample results of this are shown in Figure 7.1, where low-frequency edges are visibly reduced whilst the pictorial shape of the pedestrian is retained. Superior descriptive shape information may be extracted should this filter be applied to all image samples as this method reduces the clutter of edge-noise on both foreground and background regions.

Another approach could be applying a Watershed Colour Segmentation algorithm for unifying solid colours that are within object boundaries. The *Bilateral* filter has similar effects like the Selective Gaussian Blur filter but has a higher computational cost. The benefits of the proposed steps are not only limited to HOG features. Unifying colours in images may benefit other feature descriptors and classifiers. As discussed in this work [12, 77], quantising colours can assist shape detection algorithms and colour segmentation tasks. Lastly, another contribution could be creating a large synced infrared datasets with high-resolution images, where infrared samples are captured in synchronous with the visual samples. This will require special optical lenses that will allow different cameras to capture similar pictures with similar depth perception.

Developing a convolutional neural network architecture that is capable of suppressing edge-noise in images before applying object classification or detection can be a crucial

step in this research. Further work may entail the use of generative adversarial networks (GANS) for generating artistic-decoded images with reduced noise and unified colours.



FIGURE 7.1: *Cartooned* image, colour unification for better boundary edge extraction. Top left is the original image. Followed by its cartooned version (top-right). The corresponding magnitudes of gradients images are shown below it [1].

Bibliography

[1] Dalal Navneet and Triggs Bill. "Histograms of Oriented Gradients for Human Detection". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference On.* Vol. 1. IEEE. 2005, pp. 886–893 (cit. on pp. i, 1, 2, 6, 12, 14–16, 18, 22, 24, 31, 48–52, 64, 65, 84, 88, 94).

[2] Felzenszwalb Pedro F, Girshick Ross B, McAllester David, and Ramanan Deva. "Object Detection with Discriminatively Trained Part-Based Models". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 32.9 (2010), pp. 1627–1645 (cit. on pp. i, 1, 2, 12, 13, 16–18, 64).

[3] Jia Hui-Xing and Zhang Yu-Jin. "Human Detection in Static Images". In: *Pattern Recog. Tech. App.: Recent Advances* 1 (2008), pp. 227–243 (cit. on pp. i, 2, 12–14).

[4] Wang Xiaoyu, Han Tony X, and Yan Shuicheng. "An HOG-LBP Human Detector with Partial Occlusion Handling". In: *Computer Vision, 2009 IEEE 12th International Conference On.* IEEE. 2009, pp. 32–39 (cit. on pp. i, 2).

[5] Li Wang, Chengdong Wu, Dongyue Chen, and Baihua Lu. "Rotation-Invariant Human Detection Scheme Based on Polar-HOGs Feature and Double Scales Direction Estimation". In: *Photonics and Optoelectronics (SOPO), 2011 Symposium On.* IEEE. 2011, pp. 1–4 (cit. on pp. i, 2, 12, 19–22).

[6] Ubukata Tom, Shibata Masatoshi, Terabayashi Kotaro, Mora Alessandro, Kawashita Takehiro, Masuyama Gakuto, and Umeda Kazunori. "Fast Human Detection Combining Range Image Segmentation and Local Feature Based Detection". In: *Pattern Recognition (ICPR), 2014 22nd International Conference On.* IEEE. 2014, pp. 4281–4286 (cit. on pp. i, 2, 12, 13, 19, 20, 23).

[7] Kunene Dumisani. "SIGNI infrared pedestrian dataset, Electronic Dataset". In: (2017) (cit. on pp. i, 3, 4, 39, 84, 88, 92).

[8] Dalal Navneet. "INRIA Person Dataset". In: (2005) (cit. on pp. i, 15, 48).

[9] Overett Gary, Petersson Lars, Brewer Nathan, Andersson Lars, and Pettersson Niklas. "A New Pedestrian Dataset for Supervised Learning". In: *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE. 2008, pp. 373–378 (cit. on pp. i, 19, 48, 49).

[10] Hartley Richard and Zisserman Andrew. "Multiple View Geometry in Computer Vision". In: *Robotica* 23.2 (2005), pp. 271–271 (cit. on p. 1).

[11] De Villiers JP. "Real-Time Stitching of High Resolution Video on COTS Hardware". In: *Proceedings of the 2009 International Symposium on Optomechatronic Technologies*. Vol. 9. ISOT2009. 2009, pp. 46–51 (cit. on pp. 1, 7, 8).

[12] Kunene Dumisani, Vadapalli Hima, and Cronje Jaco. "Indoor Sign Recognition for the Blind". In: *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists*. (Johannesburg, South Africa). SAICSIT '16. New York, NY, USA: ACM, 2016, 19:1–19:9. URL: http://doi.acm.org/10.1145/2987491.2987530 (cit. on pp. 1, 93).

[13] Stefanou Stefanos and Argyros Antonis A. "Efficient Scale and Rotation Invariant Object Detection Based on HOGs and Evolutionary Optimization Techniques". In: *International Symposium on Visual Computing*. Springer. 2012, pp. 220–229 (cit. on pp. 1, 19).

[14] Vapnik Vladimir Naumovich and Vapnik Vlamimir. *"Statistical Learning Theory"*. Vol. 1. Wiley New York, 1998 (cit. on pp. 2, 25).

[15] Williams Christopher KI and Rasmussen Carl Edward. "Gaussian Processes for Machine Learning". In: *the MIT Press* 2.3 (2006), p. 4 (cit. on p. 2).

[16] Cover Thomas and Hart Peter. "Nearest Neighbor Pattern Classification". In: *IEEE transactions on information theory* 13.1 (1967), pp. 21–27 (cit. on p. 2).

[17] Broggi Alberto, Bertozzi Massimo, Fascioli Alessandra, and Sechi Massimiliano. "Shape-Based Pedestrian Detection". In: *Proceedings of the IEEE Intelligent Vehicles Symposium*. Citeseer. 2000, pp. 215–220 (cit. on pp. 2, 13).

[18] Oren Michael, Papageorgiou Constantine, Sinha Pawan, Osuna Edgar, and Poggio Tomaso. "Pedestrian Detection Using Wavelet Templates". In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference On*. IEEE. 1997, pp. 193–199 (cit. on pp. 2, 14).

[19] Dollár Piotr, Belongie Serge, and Perona Pietro. "The Fastest Pedestrian Detector in the West." In: *BMVC*. Vol. 2. 3. Citeseer. 2010, p. 7 (cit. on p. 2).

[20] Mikolajczyk Krystian, Schmid Cordelia, and Zisserman Andrew. "Human Detection Based on a Probabilistic Assembly of Robust Part Detectors". In: *European Conference on Computer Vision*. Springer. 2004, pp. 69–82 (cit. on p. 2).

[21] Yang Yi and Ramanan Deva. "Articulated Pose Estimation with Flexible Mixtures-of-Parts". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference On*. IEEE. 2011, pp. 1385–1392 (cit. on p. 2).

[22] Zhao Liang and Thorpe Chuck. "Recursive Context Reasoning for Human Detection and Parts Identification". In: *IEEE Workshop on Human Modeling, Analysis and Synthesis*. Citeseer. 2000, pp. 136–141 (cit. on p. 2).

[23] Clark Jim. *The Basics of Heat(Core Concepts)*. The Rosen Publishing Group, New York, 2004. 55 pp. (cit. on p. 2).

[24] Cronje J and de Villiers J P. "A Comparison of Image Features for Registering LWIR and Visual Images". In: *Proceedings of the 23nd Annual Symposium of the Pattern Recognition Society of South Africa*. Vol. 1. PRASA2012. 2012 (cit. on pp. 7, 8).

[25] Duvenhage Bernardt, Delport JP, and de Villiers Jason. "Implementation of the Lucas-Kanade Image Registration Algorithm on a GPU for 3D Computational Platform Stabilisation". In: *Proceedings of the 7th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*. ACM. 2010, pp. 83–90 (cit. on p. 8).

[26] Krizhevsky Alex, Sutskever Ilya, and Hinton Geoffrey E. "Imagenet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. 2012, pp. 1097–1105 (cit. on pp. 12, 23, 35, 72, 73).

[27] Juang Li-Hong, Wu Ming-Ni, and Weng Zhi-Zhong. "Object Identification Using Mobile Devices". In: *Measurement* 51 (2014), pp. 100–111 (cit. on p. 12).

[28] Felzenszwalb Pedro F, Girshick Ross B, and McAllester David. "Cascade Object Detection with Deformable Part Models". In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference On*. IEEE. 2010, pp. 2241–2248 (cit. on pp. 12, 16, 17).

[29] Rohr Karl. "Towards Model-Based Recognition of Human Movements in Image Sequences". In: *CVGIP: Image understanding* 59.1 (1994), pp. 94–115 (cit. on p. 13).

[30] Everingham M., Van Gool L., Williams C. K. I., Winn J., and Zisserman A. *"The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results"*. 2008. URL: http://host.robots.ox.ac.uk/pascal/VOC/voc2008/ (cit. on p. 18).

[31] Xia Lu, Chen Chia-Chih, and Aggarwal Jake K. "Human Detection Using Depth Information by Kinect". In: *CVPR 2011 WORKSHOPS*. IEEE. 2011, pp. 15–22 (cit. on p. 19).

[32] Han Jungong, Shao Ling, Xu Dong, and Shotton Jamie. "Enhanced Computer Vision with Microsoft Kinect Sensor: A Review". In: *IEEE transactions on cybernetics* 43.5 (2013), pp. 1318–1334 (cit. on p. 19).

[33] Janoch Allison, Karayev Sergey, Jia Yangqing, Barron Jonathan T, Fritz Mario, Saenko Kate, and Darrell Trevor. "A Category-Level 3D Object Dataset: Putting the Kinect to Work". In: *Consumer Depth Cameras for Computer Vision*. Springer, 2013, pp. 141–165 (cit. on p. 19).

[34] Wu Haiyuan, Suzuki Kazumasa, Wada Toshikazu, and Chen Qian. "Accelerating Face Detection by Using Depth Information". In: *Advances in Image and Video Technology*. Springer, 2009, pp. 657–667 (cit. on p. 19).

[35] Mitsui Tomokazu and Fujiyoshi Hironobu. "Object Detection by Joint Features Based on Two-Stage Boosting". In: *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference On*. IEEE. 2009, pp. 1169–1176 (cit. on pp. 19, 22).

[36] Yang Wei, Song Zhan, and Wu Xinyu. "Histogram of Silhouette Direction Code: An Efficient HOG-Based Descriptor for Accurate Human Detection". In: *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference On*. IEEE. 2012, pp. 330–335 (cit. on p. 22).

[37] Piccardi Massimo. "Background Subtraction Techniques: A Review". In: *Systems, Man and Cybernetics, 2004 IEEE International Conference On*. Vol. 4. IEEE. 2004, pp. 3099–3104 (cit. on p. 23).

[38] Sheikh Yaser, Javed Omar, and Kanade Takeo. "Background Subtraction for Freely Moving Cameras". In: *Computer Vision, 2009 IEEE 12th International Conference On*. IEEE. 2009, pp. 1219–1225 (cit. on p. 23).

[39] Fukui Hiroshi, Yamashita Takayoshi, Yamauchi Yuji, Fujiyoshi Hironobu, and Murase Hiroshi. "Pedestrian Detection Based on Deep Convolutional Neural Network with Ensemble Inference Network". In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2015, pp. 223–228 (cit. on pp. 24, 35).

[40] Dollár Piotr, Wojek Christian, Schiele Bernt, and Perona Pietro. "Pedestrian Detection: A Benchmark". In: (2009) (cit. on p. 24).

[41] Redmon Joseph, Divvala Santosh, Girshick Ross, and Farhadi Ali. "You Only Look Once: Unified, Real-Time Object Detection". In: *arXiv preprint arXiv:1506.02640* (2015) (cit. on p. 24).

[42] Marcus Gary. "Deep Learning: A Critical Appraisal". In: *arXiv preprint arXiv:1801.00631* (2018) (cit. on p. 24).

[43] Boser Bernhard E, Guyon Isabelle M, and Vapnik Vladimir N. "A Training Algorithm for Optimal Margin Classifiers". In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. ACM. 1992, pp. 144–152 (cit. on p. 25).

[44] Eliot Lance. *"Support Vector Machines (SVM) for AI Self-Driving Cars"*. 2018. URL: https://www.aitrends.com/ai-insider/support-vector-machines-svm-ai-self-driving-cars/ (cit. on p. 26).

[45] Cai Yu-Dong, Zhou Guo-Ping, and Chou Kuo-Chen. "Support Vector Machines for Predicting Membrane Protein Types by Using Functional Domain Composition". In: *Biophysical journal* 84.5 (2003), pp. 3257–3263 (cit. on p. 31).

[46] Cuingnet Rémi, Rosso Charlotte, Chupin Marie, Lehéricy Stéphane, Dormont Didier, Benali Habib, Samson Yves, and Colliot Olivier. "Spatial Regularization of SVM for the Detection of Diffusion Alterations Associated with Stroke Outcome". In: *Medical Image Analysis* 15.5 (2011), pp. 729–737. ISSN: 1361-8415. DOI: http://dx.doi.org/10.1016/j.media.2011.05.007. URL: http://www.sciencedirect.com/science/article/pii/S1361841511000594 (cit. on p. 31).

[47] Chen Datong, Bourlard Hervé, and Thiran J-P. "Text Identification in Complex Background Using SVM". In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference On*. Vol. 2. IEEE. 2001, pp. II–621 (cit. on p. 31).

[48] Werbos Paul. "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences". Harvard University, 1974 (cit. on p. 31).

[49] Huang Guang-Bin, Zhu Qin-Yu, and Siew Chee-Kheong. "Extreme Learning Machine: Theory and Applications". In: *Neurocomputing* 70.1 (2006), pp. 489–501 (cit. on pp. 31, 33, 34).

[50] Fan Jialue, Xu Wei, Wu Ying, and Gong Yihong. "Human Tracking Using Convolutional Neural Networks". In: *IEEE Transactions on Neural Networks* 21.10 (2010), pp. 1610–1623 (cit. on p. 31).

[51] McCulloch Warren S and Pitts Walter. "A Logical Calculus of the Ideas Immanent in Nervous Activity". In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133 (cit. on p. 31).

[52] Rosenblatt Frank. "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain." In: *Psychological review* 65.6 (1958), p. 386 (cit. on p. 31).

[53] Nicholson Chris. *"A Beginner's Guide to Neural Networks and Deep Learning"*. 2016. URL: https://pathmind.com/wiki/neural-network (cit. on pp. 32, 35).

[54] Glosser Ca. *"An example artificial neural network with a hidden layer"*. 2013. URL: https://en.wikipedia.org/wiki/Artificial_neural_network#/media/File:Colored_neural_network.svg (cit. on p. 33).

[55] Huang Gao, Huang Guang-Bin, Song Shiji, and You Keyou. "Trends in Extreme Learning Machines: A Review". In: *Neural Networks* 61 (2015), pp. 32–48 (cit. on p. 34).

[56] Zhang Rui, Lan Yuan, Huang Guang-bin, and Xu Zong-Ben. "Universal Approximation of Extreme Learning Machine with Adaptive Growth of Hidden Nodes". In: *IEEE Transactions on Neural Networks and Learning Systems* 23.2 (2012), pp. 365–371 (cit. on p. 34).

[57] Lan Yuan, Soh Yeng Chai, and Huang Guang-Bin. "Two-Stage Extreme Learning Machine for Regression". In: *Neurocomputing* 73.16 (2010), pp. 3028–3038 (cit. on p. 34).

[58] LeCun Yann and Bengio Yoshua. "Convolutional Networks for Images, Speech, and Time Series". In: *The handbook of brain theory and neural networks* 3361.10 (1995), p. 1995 (cit. on pp. 35, 55).

[59] Taigman Yaniv, Yang Ming, Ranzato Marc'Aurelio, and Wolf Lior. "Deepface: Closing the Gap to Human-Level Performance in Face Verification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1701–1708 (cit. on p. 35).

[60] Davis Jesse and Goadrich Mark. "The Relationship between Precision-Recall and ROC Curves". In: *Proceedings of the 23rd International Conference on Machine Learning*. ACM. 2006, pp. 233–240 (cit. on p. 36).

[61] Davis James W and Keck Mark A. "A Two-Stage Template Approach to Person Detection in Thermal Imagery". In: *Application of Computer Vision, 2005. WACV/MOTIONS'05 Volume 1. Seventh IEEE Workshops On*. Vol. 1. IEEE. 2005, pp. 364–369 (cit. on p. 39).

[62] Portmann Jan, Lynen Simon, Chli Margarita, and Siegwart Roland. "People Detection and Tracking from Aerial Thermal Views". In: *Robotics and Automation (ICRA), 2014 IEEE International Conference On*. IEEE. 2014, pp. 1794–1800 (cit. on pp. 39, 40).

[63] Zhang Li, Wu Bo, and Nevatia Ram. "Pedestrian Detection in Infrared Images Based on Local Shape Features". In: *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference On*. IEEE. 2007, pp. 1–8 (cit. on p. 40).

[64] Xenics. *"Xenics Infrared Solutions - Gobi 640 Series"*. 2016. URL: https://www.xenics.com/long-wave-infrared-imagers/gobi-640-series/ (cit. on p. 44).

[65] Kunene Dumisani. "Dataset GroundTruth Creator". In: (2016) (cit. on p. 44).

[66] Fitzgerald Mary Pat. *"MIT Pedestrian Dataset"*. 2000. URL: http://cbcl.mit.edu/software-datasets/PedestrianData.html (cit. on pp. 48, 57).

[67] Guang-Bin Huang. *"Extreme Learning Machines"*. 2004. URL: https://www.ntu.edu.sg/home/egbhuang/elm_codes.html (cit. on p. 54).

[68] Vedaldi A. and Lenc K. "MatConvNet: Convolutional Neural Networks for MATLAB". In: *ACM International Conference on Multimedia*. 2015 (cit. on pp. 55, 56, 72).

[69] Xu Danfei. *"Convolutional Neural Networks for Visual Recognition"*. 2016. URL: https://cs231n.github.io/convolutional-networks/ (cit. on p. 56).

[70] LeCun Yann, Jackel LD, Bottou Leon, Brunot A, Cortes Corinna, Denker JS, Drucker Harris, Guyon I, Muller UA, Sackinger Eduard, et al. "Comparison of Learning Algorithms for Handwritten Digit Recognition". In: *International Conference on Artificial Neural Networks*. Vol. 60. Perth, Australia. 1995, pp. 53–60 (cit. on p. 56).

[71] Krizhevsky Alex and Hinton G. "Convolutional Deep Belief Networks on Cifar-10". In: *Unpublished manuscript* 40 (2010) (cit. on p. 57).

[72] Jung Yoonsuh and Hu Jianhua. "AK-fold averaging cross-validation procedure". In: *Journal of nonparametric statistics* 27.2 (2015), pp. 167–179 (cit. on p. 60).

[73] Di Franco Anthony, Guo Hui, and Rubio-González Cindy. "A comprehensive study of real-world numerical bug characteristics". In: *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE. 2017, pp. 509–519 (cit. on p. 65).

[74] Telgarsky Matus. "Benefits of Depth in Neural Networks". In: *arXiv preprint arXiv:1602.04485* (2016) (cit. on p. 68).

[75] LeCun Yann, Jackel LD, Bottou Léon, Cortes Corinna, Denker John S, Drucker Harris, Guyon Isabelle, Muller Urs A, Sackinger Eduard, Simard Patrice, et al. "Learning Algorithms for Classification: A Comparison on Handwritten Digit Recognition". In: *Neural networks: the statistical mechanics perspective* 261 (1995), p. 276 (cit. on p. 72).

[76] Lin Che-Tsung, Santoso Patrisia Sherryl, Chen Shu-Ping, Lin Hung-Jin, and Lai Shang-Hong. "Fast Vehicle Detector for Autonomous Driving". In: *Computer Vision Workshop (ICCVW), 2017 IEEE International Conference On*. IEEE. 2017, pp. 222–229 (cit. on p. 90).

[77] Skosana Vusi and Kunene Dumisani. "Edge-Preserving Smoothing Filters for Improving Object Classification". In: *Proceedings of the South African Institute of Computer Scientists and Information Technologists 2019*. (Skukuza, South Africa). SAICSIT '19. New York, NY, USA: ACM, 2019, 16:1–16:7. ISBN: 978-1-4503-7265-7. DOI: 10.1145/3351108.3351125. URL: http://doi.acm.org/10.1145/3351108.3351125 (cit. on p. 93).

This page is intentionally left blank.

Appendix A

Appendixes

A.1 Introduction

The appendixes present extra material to substantiate the discussion of the instruments used for this research. The ground truth validation tool that was used in this work is presented in Appendix A.2 The ethics clearance for capturing the samples of the SIGNI dataset is shown in Figure A.6.

A typical example of the XML files that store ground-truth information is shown in Figure A.2. The code in Listing A.1 shows how the ROI coordinates were mapped from the scaled-down pyramid images to the ground-truth resolution space for correct ground-truth validation. The cases to consider when evaluating the extent of overlap between the detection window and the ground-truth regions are shown in Figure A.3, Figure A.4 and Figure A.5. The Matlab code Listing A.2 shows the algorithm for evaluating the detection window's extent of overlap over each Ground Truth ROIs.

A.2 Localisation Tool

Thanks to Matlab's MEX compiler, a global wrapper that applies the sliding window mechanism for all classifiers was developed for ground-truth validation when addressing the human detection problem. This allows the detection and localisation of people on images that are larger than the classifier's descriptor windows. This tool takes as inputs, a trained classifier object, the image to be tested, its resolution and XML-tag, the detection window's coordinates and resolution, and lastly the original image's resolution. The optional XML-tag argument is only required when testing images with ground-truth information.

Thanks to Matlab's MEX compiler, a global wrapper that applies the sliding window mechanism for all classifiers was developed for ground-truth validation to address. This allows the detection and localisation of people on images that are larger than the classifier's descriptor windows. This tool takes as inputs, a trained classifier object, the image to be tested, its resolution and XML-tag, the detection window's coordinates and resolution, and lastly the original image's resolution. The optional XML-tag argument is only required when testing images with ground-truth information.

An alternative to MEX files is the use of a robot operating system (ROS) environment, which allows images to be sent to the executables that implement the different classifiers for processing. Once the images are classified, the classification results can be sent back to the ground-truth assessment tool. The ROS environment facilitates data communication between different executable nodes.

Consider a scenario where the original image's resolution is 640×480 and it is scaled down by a fraction of 0.2 whenever the descriptor window completes a convolution over it. If the image is scaled down 6 times, the resolution will change from $[640 \times 480]$ to $[512 \times 384], [410 \times 307], ..., [209 \times 157]$. From this image, suppose a detection occurs at position $(x_0, y_0) = (60, 35)$ with a $[128 \times 64]$ descriptor window, and we need to determine if this a true detection or not using the ground truth information.

The problem to solve is finding the descriptor window's relative position and size to the original image's resolution [See Figure A.1]. This can be solved arithmetically as follows:

1. Determine the ratios between the (ROI position coordinates on the scaled-down image) and the (resolution of the scaled-down image).

2. Then multiply the ratios with the original image's resolution to obtain the ROI's position on the original image. The same is done for mapping the width and height.

Note that the subscript D refers to the variables from the scaled-down image resolution space hence the (D), and the subscript G refers to the variables in the ground truth resolution space hence the (G). The Matlab code snippet n Listing A.1 does this task.

After mapping the ROI's from scaled-down images to the original image size. The extent of overlap for ROIs over the ground truth subregions is evaluated. To do this, the six possible cases shown in Figure A.3 must be considered, Figure A.4, and Figure A.5. The evaluation of the detection window is only conducted if the distance from the (midpoint of the detection window rectangle) to the (midpoint of the evaluated ground

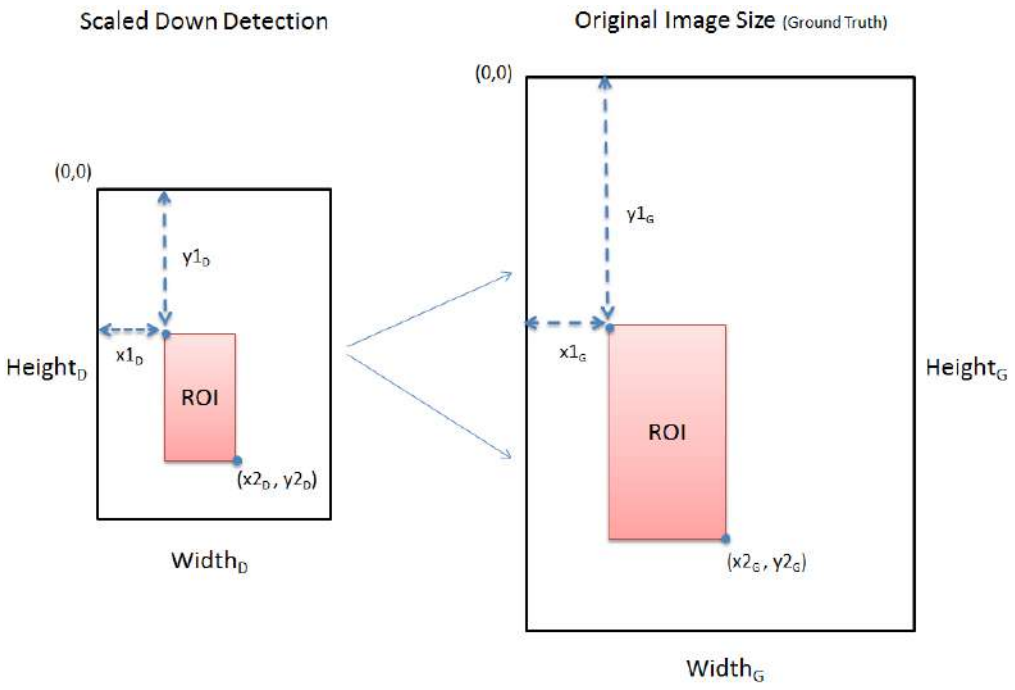


FIGURE A.1: The detection window position and size mapped to the ground truth resolution.

truth window) is less than half the length of the ground truth rectangle's hypotenuse. Therefore, the extent of overlap for detection windows that are far from the ground truth window is not applied to avoid unnecessary computation.

Once the ground-truth ROI coordinates are extracted from an XML file (similar to Figure A.2) using the code from lines 2 to 8 in Listing A.2, six conditions are evaluated using several conditional statements. For Cases A, B, C and D, the detection window is first examined whether it is above or below the ground truth ROI. From there, the height of the overlapping rectangle is calculated followed by the horizontal overlap. For cases E and G, the overlapping width and height become the actual side lengths of the detection window since the entire detection window is within the ground truth ROI. With the case of F, the overlap equals the ground truth window's sides.

The lines 17 to 55 in Listing A.2 show the self-explanatory implementation of this in Matlab. With the width and height of the overlap, the area of the overlapping rectangle can be calculated and compared to the area of the ground truth rectangle. A detection window is only regarded as a "true position" if it overlaps the ground truth window by at least 50%, such that the detection window is not twice as big as the ground truth window.

OpenCV has an easier approach for evaluating the extent of overlap between ROIs.

Given two OpenCV rectangles of types *Rect2d*, the intersection rectangle can be computed by the line in python *overlapRect = ROI_1 & ROI_2;*, and the area of overlap can be determined by simply calling the function *overlapRectable.area();*.

```
<?xml version="1.0"?>
- <PositiveSamples totalNumOfSubregions="86" numImages="100">
    <image numSubRegions="0" filename="person_001.bmp"/>
    <image numSubRegions="0" filename="person_002.bmp"/>
    <image numSubRegions="0" filename="person_003.bmp"/>
  - <image numSubRegions="1" filename="person_004.bmp">
      <subregion yStart="122" yEnd="682" xStart="70" xEnd="328"/>
    </image>
  - <image numSubRegions="1" filename="person_005.bmp">
      <subregion yStart="58" yEnd="740" xStart="29" xEnd="365"/>
    </image>
    <image numSubRegions="0" filename="person_006.bmp"/>
  - <image numSubRegions="1" filename="person_007.bmp">
      <subregion yStart="94" yEnd="494" xStart="107" xEnd="306"/>
    </image>
  - <image numSubRegions="1" filename="person_008.bmp">
      <subregion yStart="115" yEnd="582" xStart="129" xEnd="370"/>
    </image>
  - <image numSubRegions="1" filename="person_009.bmp">
      <subregion yStart="74" yEnd="629" xStart="106" xEnd="334"/>
    </image>
  - <image numSubRegions="2" filename="person_010.bmp">
      <subregion yStart="152" yEnd="606" xStart="191" xEnd="335"/>
      <subregion yStart="233" yEnd="345" xStart="33" xEnd="113"/>
    </image>
  - <image numSubRegions="1" filename="person_011.bmp">
      <subregion yStart="149" yEnd="508" xStart="154" xEnd="323"/>
    </image>
  - <image numSubRegions="1" filename="person_012.bmp">
      <subregion yStart="188" yEnd="484" xStart="199" xEnd="311"/>
    </image>
  + <image numSubRegions="1" filename="person_013.bmp">
```

FIGURE A.2: A typical example of the produced xml tree containing ground-truth information.

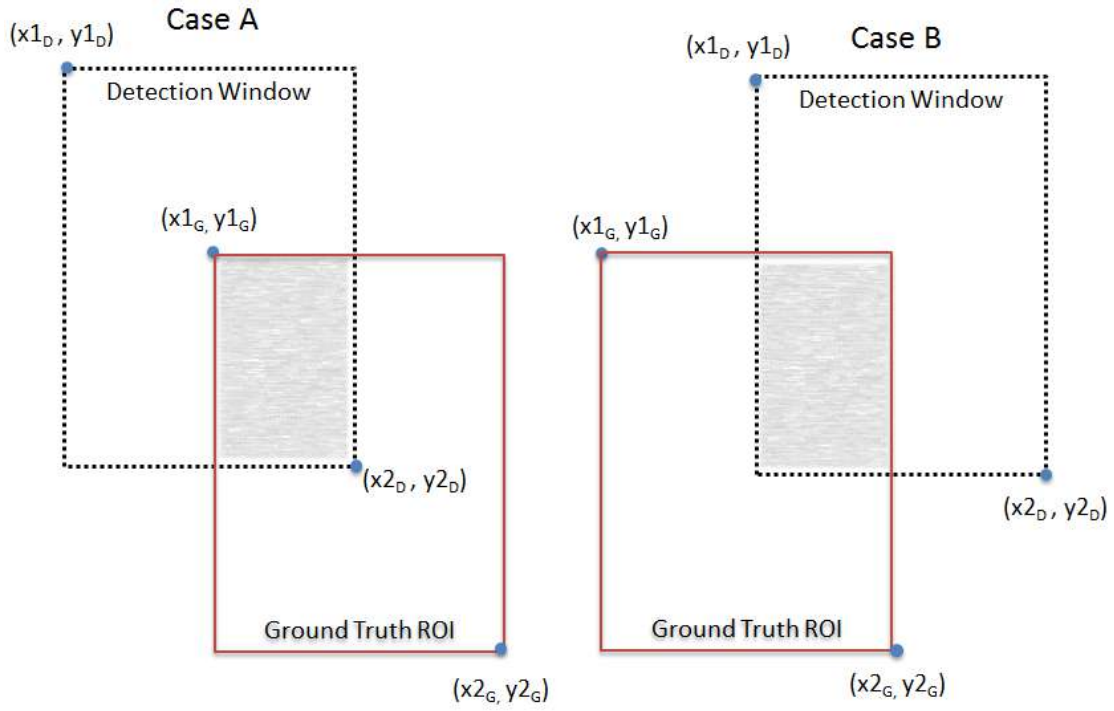


FIGURE A.3: Cases A and B, when the detection window is above the ROI coordinate.

```
%ROI position coordinates vs scaled down image resolution.
y1positionRatio = y1D / Height_D;
x1positionRatio = x1D / Width_D;

%Relative ROI position mapped to the Ground Truth resolution space.
y1G = round( Height_G * y1positionRatio);
x1G = round( Width_G * x1positionRatio);

%ROI size vs scaled down image resolution.
height = y2D - y1D;
width = x2D - x1D;
currentHeightRatio = height/Height_D;
currentWidthRatio = width/Width_D;

%Relative ROI size mapped to the Ground Truth resolution space.
originalWindowHeight = round(Height_D*currentHeightRatio);
originalWindowWidth =  round(Width_D* currentWidthRatio);
x2G = x1Original + originalWindowWidth;
y2G = y1Original + originalWindowHeight;
```

LISTING A.1: Mechanism for mapping the position and size of the scaled down descriptor window to the ground truth resolution space.

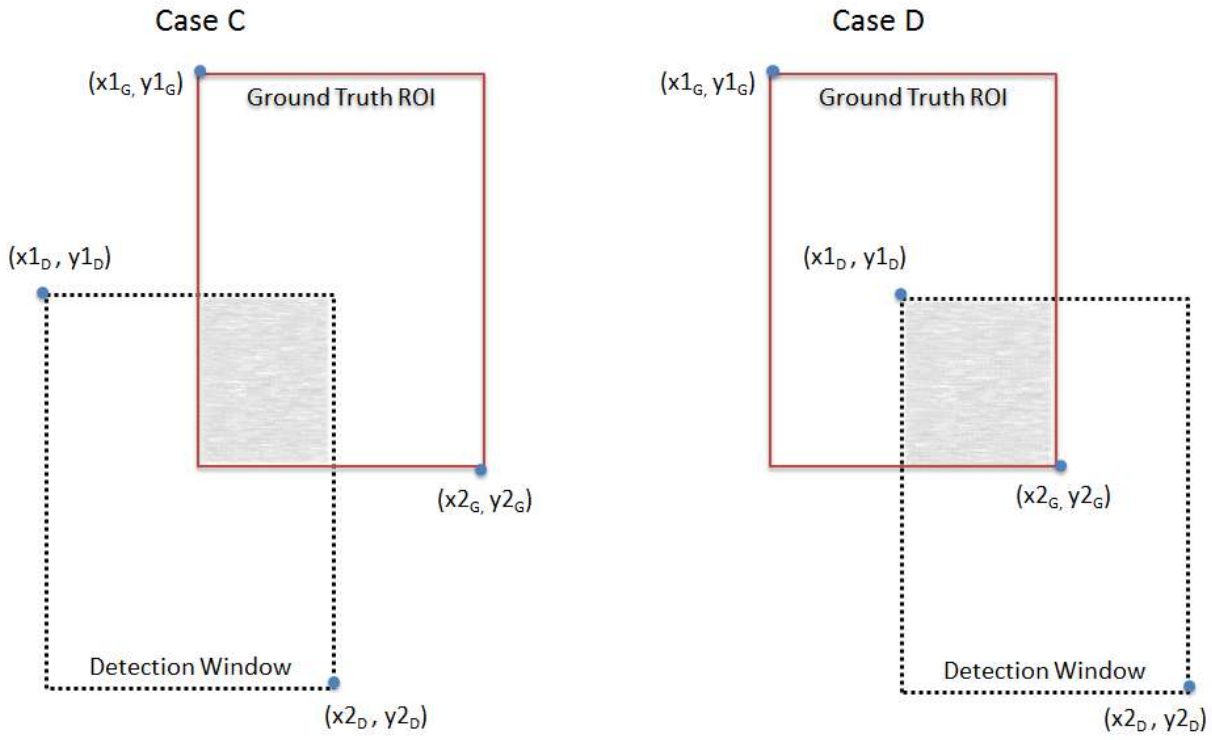


FIGURE A.4: Cases C and D, are for cases when the detection window is below the ground truth ROI coordinate.

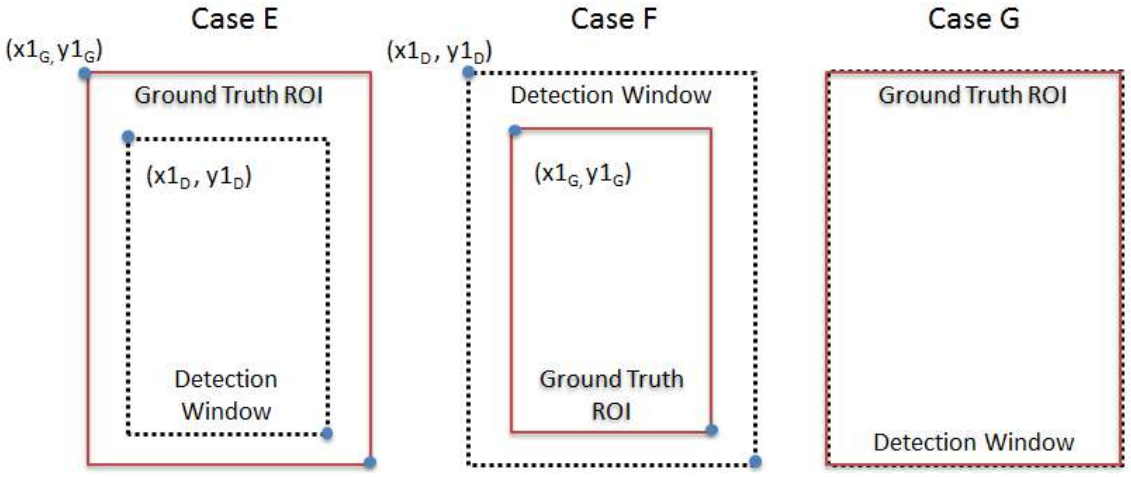


FIGURE A.5: Cases E and G, are for cases when the detection windows is within or equal to the ground truth ROI. Then in case F, the detection window fully covers the ground truth ROI.

```
%Read all subregions
subregion = xmlImageTag.getFirstChild();
for i=1:numSubRegions
  %Coorditates of the Ground Truth window
  x1G = uint16(eval(char(subregion.getAttribute('xStart'))));
  y1G = uint16(eval(char(subregion.getAttribute('yStart'))));
  x2G = uint16(eval(char(subregion.getAttribute('xEnd'))));
  y2G = uint16(eval(char(subregion.getAttribute('yEnd'))));

  %% Validate Ground Truth
  [overlappingWidth, overlappingHeight, overlapPercentange] = deal(0.0);

  % Overlaping cases between the detection and ground truth window
  % Case A and B
  if( (y1D < y1G) && ((y2D)> y1G) && (y2G>y2D)) %vertical overlap
  overlappingHeight = (y2D)-y1G;
    %Case A
    if( (x1D < x1G) &&  (x1G<x2D) && (x2D<x2G) )  %horizontal overlap
      overlappingWidth = x2D-x1G;
    end
    %Case B
    if( (x1D > x1G) &&  (x1D<x2G) && (x2G<x2D) )
      overlappingWidth = x2G-x1D;
    end

  %Case C and D
  elseif( (y1D > y1G) && ((y2G)>y1D) && (y2D>y2G) )
    overlappingHeight = (y2G)-y1D;
    %Case C
    if( (x1D < x1G) && (x1G<x2D)  && (x2D<x2G) )
      overlappingWidth = x2D-x1G;
    end
    %Case D
    if( (x1D > x1G) && (x1D<x2G) && (x2G<x2D))
      overlappingWidth = x2G-x1D;
    end
  %Case E and G
  elseif((x1D >= x1G) && (y1D>= y1G) )
    if((x2D) <= x2G && (y2D<= y2G))
      overlappingWidth = widthDetection;
      overlappingHeight = heightDetection;
    end
  %Case F
```

```
44    elseif((x1D < x1G) && (y1D<y1G) )
45      if((x2D) > y2G)
46        if (widthDetection * heightDetection)/GroundTruthAreaSize <= 2
47            overlappingWidth = widthGroundTruth;
48            overlappingHeight = heightGroundTruth;
49        else
50          continue;      % False Positive
51        end
52      end
53
54    %Validate the  overlap percentange
55    heightGroundTruth = y2G - y1G;
56    widthGroundTruth  =  x2G - x1G;
57    GroundTruthAreaSize = single(widthGroundTruth*heightGroundTruth);
58    OverlappingAreaSize = overlappingWidth*overlappingHeight;
59    overlapPercentange = single(OverlappingAreaSize) / GroundTruthAreaSize ;
60    if(overlapPercentange >= 0.5 && overlapPercentange<=2)
61      numTruePositives = numTruePositives +1;
62      return;
63    else
64      %Move to the next subregion on the xml
65      subregion = subregion.getNex1GSibling();
66      continue;    % False Positive
67    end
68  end
```

Listing A.2: Mechanism for evaluating the detection window's extent of overlap over each Ground Truth ROIs.

A.3 Ethics Clearance

A permit and ethics clearance from the University of the Witwatersrand was obtained prior to the creation of this dataset. The document is shown in Figure A.6.



UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG

OFFICE OF THE DEPUTY REGISTRAR

1 September 2016

Mr Dumisane Kunene
Student number 890669
School of Mathematics & Computer Sciences

TO WHOM IT MAY CONCERN

"The implementation and review of Human detection systems"

This letter serves to confirm that the above project has received permission to be conducted on University premises, and/or involving staff and/or students of the University as research participants. In undertaking this research, you agree to abide by all University regulations for conducting research on campus and to respect participants' rights to withdraw from participation at any time.

This notice serves as proof that the University's internal mailing system may be used as the mechanism by which potential participants can be approached.

If you are conducting research on certain student cohorts, year groups or courses within specific Schools and within the teaching term, permission must be sought from Heads of School or individual academics.

The necessary ethical clearance has been obtained.

Nicoleen Potgieter
Deputy Registrar

Private Bag 3, Wits, 2050, South Africa | T +27 11 717 1204/8 | F +27 86 553 2271 | www.wits.ac.za

FIGURE A.6: The necessary ethics clearance for creating the SIGNI dataset.