A Multi-Level Machine Learning System for Attention-Based Object Recognition

Ji Wan Han

A thesis submitted in partial fulfilment of the requirements
of the University of Hertfordshire
for the degree of Doctor of Philosophy

May 2010

Abstract

This thesis develops a trainable object-recognition algorithm. This algorithm represents objects using their salient features. The algorithm applies an attention mechanism to speed up feature detection.

A trainable component-based object recognition system which implements the developed algorithm has been created. This system has two layers. The first layer contains several individual feature classifiers. They detect salient features which compose higher level objects from input images. The second layer judges if those detected features form a valid object. An object is represented by a feature map which stores the geometrical and hierarchical relations among features and higher level objects. It is the input to the second layer. The attention mechanism is applied to improve feature detection speed. This mechanism will lead the system to areas with a higher likelihood of containing features when a few features are detected. Therefore the feature detection will be sped up.

Two major experiments are conducted. These experiments applied the developed system to discriminate faces from non-faces and to discriminate people from backgrounds in thermal images. The results of these experiments show the success of the implemented system. The attention mechanism displays a positive effect on feature detection. It can save feature detection time, especially in terms of classifier calls.

Acknowledgments

First and foremost I want to thank my supervisor Dr. Peter Lane. Without him, this dissertation would not have been possible. I thank him for his patience and encouragement that carried me on through difficult times, and for his insights and suggestions that helped to shape my research skills. His invaluable ideas and feedback contributed greatly to this thesis. I have improved myself on machine learning, computer vision and related domains under his supervision.

I am very grateful to my supervisors, Dr. Neil Davey and Dr. Yi Sun. They give me invaluable suggestions and guidance on my thesis. Their hard work promoted my completion of this thesis. I thank Dr. Toby Breckon and his Digital Image Processing Lab at Cranfield University for providing the thermal imagery for this thesis. It is an important part of this thesis. I thank all other staff, students and friends, who gave me support in research and life, during the course of the project.

Last but not least, I thank my family, especially my wife, for always being there when I needed them most, and for supporting me through all these years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this thesis, I present a component-based object recognition system combining computer vision and machine learning technology. The attention mechanism is one of its features. This system starts from understanding simple and isolated objects, then learns how to recognise objects from a complex real-world background. It is first tested on learning to discriminate single faces from non-faces. This is to test the system's basic structure. Then it is applied to detect people in thermal real-world images.

1.1 Problem Definition

Computer vision is the science that focuses on developing machines which can see the world [39, 98]. It has long been a dream of computer vision scientists to make such a machine with the ability of the human vision system. With the development of computers, researchers have made solid progress towards this dream. Nowadays, computer vision has developed into an integrated science and technology, which has close relations with many subjects, such as artificial intelligence, machine learning, mathematics, neurobiology, signal processing, and so on. The development of related knowledge has given computer vision researchers the chance of exploring how a machine can see like a human does in a variety of ways. In particular, psychology and neuro-science provide clues on the limita-

tions and capabilities of the human vision system. This thesis takes advantage of some of these insights, and so connects up with these human-centred sciences.

A key and classical task in computer vision is object recognition, which detects and locates specific objects from images. This can be done by humans without much effort, but for a computer vision it is still a considerable challenge. State of the art computer vision systems can process specific cases successfully: certain objects in certain environments. However it is still not possible to create a general computer vision system which can perform like humans in a wide range of environments.

Many applications would benefit from better machine vision, including medical diagnosis, safety inspection, daily life surveillance, military reconnaissance, industrial measurement, and so on. Every progress towards robust object recognition will make computer vision have a deeper influence, both on research and daily life. There are many areas of research in the computer vision domain. I will focus on object recognition by applying both computer vision and machine learning algorithms.

An object recognition system is computationally intensive and its detection performance is liable to be influenced by noise. This thesis aims to develop algorithms which can focus limited computational resources on image patches with a higher likelihood of containing target objects. By applying the developed algorithms, less will be processed, therefore less noise will be input into the system and less information will be processed. Object recognition speed will be improved or with the same time consumed, a larger scene can be processed.

1.2 My Approach

This is a machine learning based algorithm. In this approach, an object is divided into individual components. These components form an object, according to a hierarchy. This algorithm learns to identify individual components and the hierarchical and geometrical relations among them. The attention mechanism emulates a characteristic of human vision that we look to gather information

on what we expect to see. It is applied to improve object detection speed. A dual-layer hierarchical system is created to verify the ability of this algorithm. It aims to be a general object detection system, but in order to limit the research effort within limited time, I chose only to detect a limited number of objects from images.

1.3 Challenges

Machine vision is a challenging task, as different objects have different visual properties. This thesis attempts to construct a dual-layer trainable system, and the nature of the system adds its own challenges:

1. How to select key individual features to represent an object effectively;
2. The proper classifiers. It is important to choose suitable classification algorithms to detect individual features;
3. How to organise individual classifiers together to recognise a compound object effectively;
4. A proper training method to train this multi-layer system. It is straightforward to train a single classifier, but to train a multi-layer system requires an effective connection between layers. The substance of this challenge is how to generate proper training sets for each layer;
5. How to evaluate the performance of the system.

How to solve these challenges is crucial to the success of the proposed algorithm and its application, an object recognition system. These challenges will be addressed in the following chapters.

1.4 Contributions

This thesis focuses on discussing a novel object recognition algorithm and a trainable object recognition system applying this algorithm. The completion of it

makes several contributions to the computer vision domain.

- A component-based object recognition algorithm. A system applying this algorithm is constructed. Several experiments have been conducted to verify and test the viability and performance of both the algorithm and the system. This system already can discriminate faces from non-faces and detect people in thermal images even when there is occlusion on the targets;
- Application of the attention mechanism to help locate features, therefore speed up the detection of these features. The system learns the geometrical and hierarchical relations among features composing the object and utilizes these relations to locate other features when some features are found;
- Comparison between different algorithms. These algorithms include single-layer classifiers, dual-layer classification system without attention mechanism and dual-layer classification system with attention mechanism;
- An effective method of representing an object in a feature map;
- An effective training method of training the multi-layer object recognition system;
- Another contribution is the databases created for the component based face classification and person detection within thermal images;

1.5 Thesis Outline

I give the background of this thesis in Chapter 2 by introducing concepts, algorithms, and related research.

The system created for this thesis is introduced in Chapter 3. This chapter describes the structure and training method of my system.

Chapter 4 describes the experiments applying this system to discriminate single faces from non-faces. The performance of the system will be investigated by comparing it with traditional approaches and its different configurations. The

multi-layer training method is applied to train the system and training sets for this system to classify faces are created. The aim of this chapter is to verify the viability of the proposed algorithm and system.

In Chapter 5, I apply my system to detect people in more complex infrared images. This chapter aims to investigate the generality of my system. The system will have the same basic structure like that face classification version introduced in Chapter 4 but will have different feature search strategies.

The experimental results are analysed further in Chapter 6. I summarize what has been achieved and what I have learned from completing this thesis. There will be discussion about the performance of the system. There is always space to improve a computer vision algorithm and the system applying it. I will draw a conclusion and describe the future directions of improving the algorithm and system introduced in this thesis in this chapter as well.

The databases used in this thesis are described in Appendix A and some key source code are included in Appendix B.

Chapter 2

Background

The object recognition system developed in this thesis is built upon previous work and knowledge by adding my contributions. In this chapter, I will review related concepts, algorithms and methodologies with a discussion of other researchers' applications which are relevant to this thesis. Through these reviews and discussions, the background of the developed system will be explained and readers will be clear about the inspiration and basis of this system.

2.1 Object Recognition

Object recognition is the technology of determining if an object exists in an image or video sequence, if it does, the system should report its location and measurement. In some parts of the literature, object recognition also refers to detecting and naming the detected objects, for example, to detect a person's face from background and produce the name of this detected person [92, 115]. In this thesis, object recognition has two main functions, indicating the existence of an object and locating the object. It is the basis of a scene understanding system where the general aim is to generate a reasonable explanation of an image, reporting what objects are there and the relations, both spatial and logical, among these objects. Obviously, the most important job of object recognition is to decide whether an object exists within an image.

For a human vision system, it seems effortless to discriminate and locate an object within a visual source even when this object is viewed from different view points, different distance, or even when the object is partially occluded by other objects. More challenging, this object can be in different shapes, colours, or sizes under different contexts [7, 89, 104]. In his PhD dissertation, Stanley [13] argued that a robust object recognition system needs to compensate not only for changes in the position, scale, and pose of the object, but also for variability in lighting, possible occlusions, and especially for the changes in the particular example of an object class.

Much effort have been made towards a robust object recognition system. Researchers have been working in many different directions to achieve an improvement in object recognition [47, 114, 88, 56, 93, 64]. Object recognition has been developed into an interdisciplinary subject. It has strong links to machine learning, artificial intelligence, psychology, image processing, neurobiology as well as mathematics. These different subjects contribute to the development of the object recognition. For example, the research in psychology and neurobiology give suggestions on how a biological vision system finds an object. Their theories may help researchers of object detection develop algorithms having characteristics of a biological vision system, such as neural networks [42, 35, 86]. In return, an object recognition system may help researchers in biological disciplines have a deeper understanding of biological vision theories by verifying them via computer models [84]. Image processing methods can help an object recognition system find objects more easily by enhancing useful information and reducing noise, such as edge detection, contrast enhancement between foreground and background, and noise filtering [41, 98, 1]. Recently with the development of machine learning algorithms, object recognition systems have gained significant improvement on their classification performance [61]. Machine learning gives object recognition systems the ability of learning the object from variety of its samples. This ability is extremely useful when it is difficult or impossible to represent an object by manually programming.

There are two main categories of object recognition systems, template match-

ing and statistical classification. Early object recognition systems mainly adopt template matching [19]. This approach finds image patches matching a template image. It is relatively simple and straightforward, therefore easy to implement. But there are several drawbacks. Because it is based on pre-collected sample images, it is difficult to cover all possible appearances of an object under different environments, such as the different lighting. This can make the system need some preprocessing to improve the quality of both input and template images, and often fails to cope with the rotational and scale variety. The result of this drawback might be low efficiency and high complexity. The system may also fail when an object is occluded. The robustness of this approach is therefore influenced by these drawbacks. By contrast, most recent developed object recognition systems use statistical classification [13, 56]. Statistical classification is a kind of classification method based on quantitative and statistical information of a set of input samples with discrete categories, i.e. positive or negative [48]. These categories will be assigned to an unseen input based on a learned statistical model. These systems take advantage of the rapid development of machine learning technology. This approach provides a chance to learn and recognise an object without explicit manual input. An object recognition system should learn key characteristics of an object from a variety of samples. These samples provide the system a chance to learn the properties of the object. Therefore the system performance will be more robust on new object appearances.

2.2 Hierarchical Representation of Objects

The ultimate aim of computer vision is to construct a machine which has the ability to automatically interpret the visual world in almost all kinds of situation like a human being [75, 39, 9]. Robust object recognition is often the starting point of a computer vision system. Without accurate object recognition, it will be difficult to understand a scene correctly. A good detection strategy is vital to the object recognition performance. Here, *strategy* means the structure of the object recognition system and how the system organises useful information

presented to it. It is about how to decompose a given scene using structured knowledge which can include facts and ideas, understanding, and the totality of what is known. The knowledge is mostly about the definition or description of different objects and their possible spatial and logical combinations in the real world. An effective object recognition system can utilize learned knowledge to solve the problem of detecting objects from a target scene and combining them to form a meaningful explanation of this scene. The detection strategy is the key to achieve higher performance with less consumption of computation resources.

A general task of object recognition is to make useful decisions about real physical objects and their key features based on sensed images [39, 29]. It aims to help to generate a correct and effective representation of the real world [98]. The real world is composed of a variety of objects which are distinguished by their salient features. A key property of many complex computer vision systems that allows us to comprehend, analyze, and build such systems is their decomposition into an hierarchy [101]. Such hierarchical representations have become an important method to solve problems in computer vision [23].

This world can be described using hierarchical expressions [22, 3, 85]. There are many hierarchies existing in this world. Whether we are concerned with engineering and science research or economy research, we use hierarchical representations to express different relations among objects. The human vision system is one of the most sophisticated vision systems in the world [8]. It takes advantage of representing scenes using hierarchies to get a very effective scene interpretation [12, 2, 69]. At the same time, it can use learned knowledge to help itself find expected objects in the hierarchy. This mechanism is a closed loop. The hierarchical object detection system developed for this research is inspired by this fact and I gave it a name '*Cengji*', which is the Chinese word for hierarchy.

The human brain stores knowledge of objects in a hierarchical structure [100, 38, 2]. This is the structural basis of the highly effective human vision system. Figure 2.1 shows a hierarchy of a face (top left). The face is a structure composed from a left eye, right eye, nose and mouth. This form of structure applies to other scenes, for example, the office or the natural environment. Each node in

the hierarchy represents knowledge about an individual object. The nodes are organized by hierarchical knowledge to form a meaningful understanding of the object.

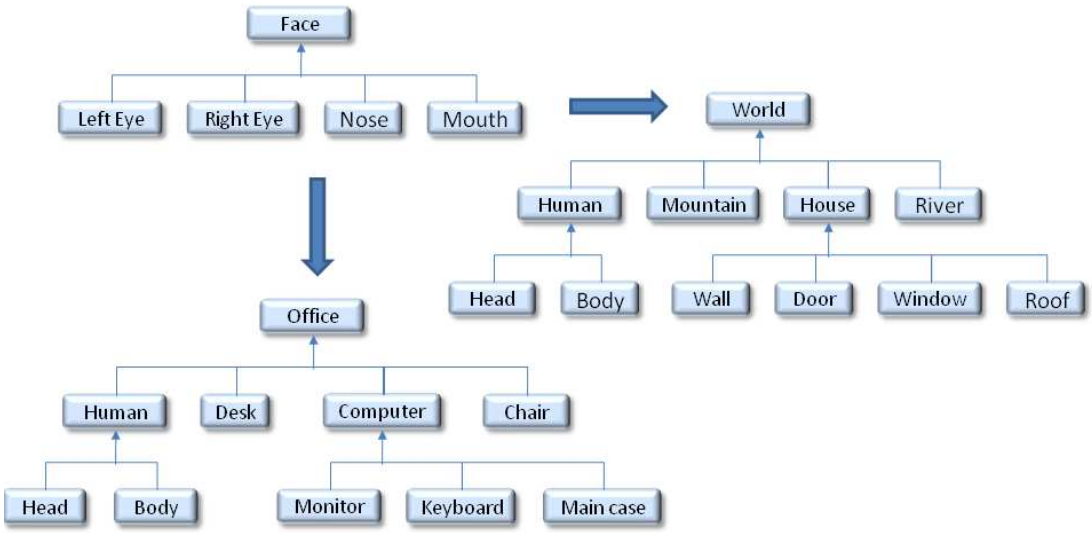


Figure 2.1: *Hierarchical structures of objects and scenes*

The human vision system resolves scenes into components and learns the relations among these components to form knowledge of a scene [12]. In the same way, an object can be resolved into individual key components or features and knowledge will be formed for this hierarchy. This knowledge helps the human vision system to understand an object in two ways, bottom-up and top-down [34, 76, 4]. In a bottom-up approach, the individual features found in an object are first detected. These features are then grouped together to form larger objects, which then in turn are grouped, sometimes in many levels, until a complete hierarchy is formed. In a top-down approach, an overview of the hierarchy is formed first. This overview predicts which features form objects and how these objects compose the scene. The object recognition system searches targets in different levels until it reaches the basic features.

The visual process of the human vision system must attend to pieces of the scene in turn, and the way in which each fixation point is selected is based on a combination of information from different sources, including lower-level responses

to visual stimuli [106, 112], and higher-level expectations of what objects may be found in the scene [11]. Although recent computer-vision systems have incorporated component-based recognition processes, there are still few which integrate the learning of compound objects with the attention mechanism which guides the system to areas likely to contain relevant features.

The main benefit of adopting the hierarchical approach will be smaller area to be processed by the system, which will speed up the system. This speedup is achieved with the help of the attention mechanism. The attention mechanism helps the system concentrate its calculation power only on areas of interests without paying attention to other irrelevant areas. With the same computation power, a system with the attention mechanisms can process the same input image faster or process larger input with the same time. The hierarchical representation is an effective way of describing an object. It reduces information to be processed by the system by keeping key features of object. It also reduces the chance of introducing noise because less areas are processed. Therefore the hierarchical representation is a good choice for this research.

2.3 Attention Mechanism

The study of how machines may handle visual information has had a long history, and combines information from several disciplines. Within psychology, vision typically is seen as a way in which high-level knowledge is acquired about the world around us. For example, Marr [73] proposed a model of visual perception in which visual stimuli are processed in stages, leading from the raw visual input to a final, high-level representation. However, it has long been recognised that locating information in a visual scene requires a more active seeking of patterns based on familiar patterns, or schemata [40, 66, 67].

Human perception relies considerably upon expectations of what may be found in a scene. For example, in character recognition, an expectation that letters are from a standard alphabet, or form words, can enable correct identification of otherwise ambiguous data [81, 91]. Biederman [11] has also shown that

objects are recognised more quickly and more accurately when they fall within a familiar schema. Various cognitive models have been developed of these phenomena [67, 91], but none has reached the level of performance required to perform meaningful identification of objects in a complex application.

2.3.1 Associative Memory

When a human sees a part of a familiar object, a whole image of this object can be formed in the human's eye. A piece of component of this object helps the human brain remember the object. Humans can also categorize the world without explicit teaching. This mechanism can not only help humans rapidly detect or recognise objects from a complex background but also give indication of relative features of the detected feature. In computer vision, an autoassociator has the similar mechanism. An autoassociator reproduces at output the same pattern that was presented at input. It can store independent memories on the same set of connections and it has the desirable property of *'cleaning up'* incomplete or noisy inputs [74].

2.3.2 Attention Mechanism

Attention is the cognitive process of selectively concentrating on one aspect of the environment while ignoring other things [cited from en.wikipedia.org]. This mechanism allows humans to concentrate on certain tasks without being disturbed by irrelevant information [59, 87, 71, 58, 21, 37, 83]. Attention is one of the most intensely studied topics by psychologists and neuroscientists, because it is closely tied to perception. Attention mechanism has attracted much attention from computer vision scientists. They hope it can improve efficiency of scene understanding model.

Some of nowadays computer vision systems do not discriminate importance of individual parts of image to the understanding of whole image. They pay the same attention to each part and deal with them using the same methods and procedures. This situation spends limited computation resource on some useless

calculations. It wastes computation time and reduces the computer vision system efficiency. The application of an attention mechanism can change this situation clearly. It makes the computer vision system ignore irrelevant information and concentrate on useful clues. This can save much time and computation resource.

Lane *et al* described the attention mechanisms in CHREST, a computational architecture of human visual expertise [40, 66]. They summarised the role of attention within CHREST as follows. First, an image is perceived, and, as the eye has a limited field of view, a portion of the image has its features extracted. These features are then used to sort through long-term memory, seeking a familiar pattern. Any retrieved pattern is placed into short-term memory. The contents of short-term memory, some high-level domain-specific knowledge, and any items on the periphery of the field-of-view will all combine to guide the model's eye to locate a new point of the image to focus on. This process continues, and the model will attempt to build up, in short-term memory, a set of pointers to familiar patterns in long-term memory which cover the image. When the attention mechanisms are applied to an object recognition system, it can provide the system useful information to search within areas with higher likelihood containing a target object.

2.4 Machine Learning Algorithms

Machine learning is the research and application of algorithms and techniques that allow computers to acquire information from data automatically or under the supervision of a human being [77, 6, 78]. We can describe an object in two ways: program a computer by explicitly telling it how an object looks or present a computer with sufficient samples to teach it about the object. The first approach needs a manual description which may not be feasible to cover all characteristics or appearance of an object. For example, a human face can have a variety of appearances. We do not have a formal description covering all possible faces. But we do know there are some common properties existing which discriminate faces from other objects. Therefore, the second approach presents samples of

object to a computer and teaches the computer to find the common properties. This approach avoids the tedious manual description of an object and has much better generality [77, 6].

Usually, machine learning methods fall into two categories: supervised learning and unsupervised learning [48, 77, 6]. Supervised learning creates a learning model from training data and uses this model to predict unseen data. There are two parts in the training data, one is the input part, another is the output part accompanied with the input. The input is often a vector and its output can be a continuous value (regression), or a class label of the input object (classification). Unsupervised learning generates a density model fit for the input training data. It is distinguished from supervised learning in that there is no output accompanied with the input data.

Machine learning has wide applications in modern science and technology. Recently, it has been applied to computer vision successfully. It helps a computer vision system achieve notable performance in object recognition, scene understanding and so on. Typical algorithms include Support Vector Machines, Neural Networks, Haar like feature classifier, k-Nearest-Neighbour Classifier, K-means Clustering and so on [26, 50, 108, 70, 109, 110, 48, 77]. There are three main machine learning algorithms applied in developing my object recognition system, *support vector machine*, *feedforward neural network*, and *Haar like feature classifier*. The support vector machine is used as the main classification algorithm for its high classification performance. In this thesis, it is used to classify image patches to find individual features. It will also be used to classify the structure of the detected image patch to decide if it belongs to a target object. A feedforward neural network will act as an autoassociator to store an associative memory of the object structure and its components [27]. The attention mechanisms will use it to focus the attention of the proposed object recognition system. A Haar like feature classifier is also applied in the experiments as a comparison. I will describe these algorithms in turn.

2.4.1 Support Vector Machines

Support vector machines (SVMs) are a relatively recent class of learning system for data classification and regression which received significant development after the 1990s [26, 50]. Much attention has been paid to the theoretical understanding of SVMs and their implementation, with many SVM applications becoming used to solve practical problems [32, 44, 50].

SVMs, based on the principle of structural risk minimization now form a well established approach in the application of machine learning algorithms and are proving to be particularly promising when used to construct accurate models based on large feature spaces [78, 97, 50, 26]. Particularly, SVMs deliver state-of-the-art performance in real-world applications [26]. They have some superiorities over other approaches, especially: a) global minimum solution, and b) learning and generalization in huge dimensional input spaces [26, 50]. Essentially they use a hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. The aim is to find a hyperplane which can classify two classes of data correctly, by maximizing the distance between the two classes of data and the hyperplane, in a space of higher dimension (see Figure 2.2).

SVMs [20] perform pattern classification by determining the separating hyperplane at a maximum distance to the closest points in the training set. These points are called support vectors. The decision function of the SVM has the form:

$$f(x) = sign[\sum_{i=1}^{N} \alpha_i y_i K(x, x_i) + b], \tag{2.1}$$

Where x is the data point to be classified, x_i are support vectors, K is the kernel function mapping inputs to a higher dimensional space, N is the number of support vectors, b is a constant decided from training and $y_i \in \{-1, 1\}$ is the class label of the support vector x_i. The coefficients α_i are the solution of a quadratic programming problem. The margin, which is the distance of the support vectors

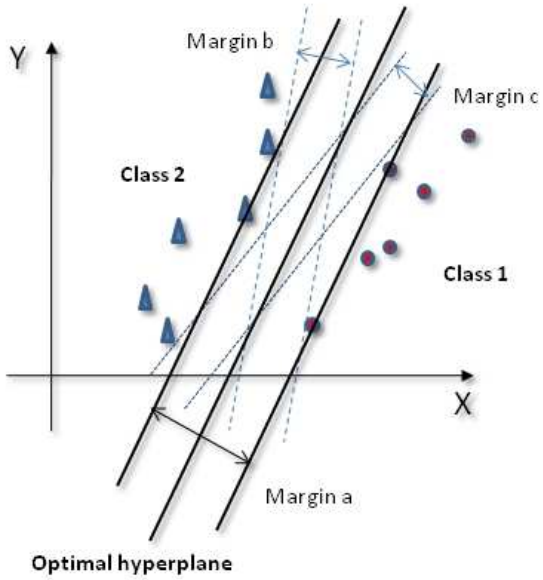


Figure 2.2: *Class 1 and class 2 can be separated by many hyperplanes. Only the optimal hyperplane separates two classes with maximum margin. Margin b and c are shorter than margin a. Those points lying on the margin generated by the optimal hyperplane are support vectors.*

Kernel Type	Kernel
linear	$x_i^T x$
polynomial	$(x_i^T x + \tau)^d$
radial basis function	$exp(-\frac{\|\|x-x_i\|\|^2}{2\sigma^2})$
sigmoid	$tanh(\kappa x_i^T x + \theta)$

Table 2.1: *The most popular kernel types.*

to the hyperplane, is thus given by:

$$M = \frac{1}{\sqrt{\sum_i^N \alpha_i}} \tag{2.2}$$

The margin is an indicator of the separability of the data within the dimensionality of the hyperplane (Figure 2.2).

The kernel function plays a key role for SVMs in solving real-world problems because many such applications are not linearly separable in their original dimensional space (i.e. that of the input). By applying a kernel transform K, the input data vectors are mapped into a higher-dimensional space. In this space, the mapped data vectors could be linearly separable or have improved separability [26, 97]. There are several popular kernel transforms shown in Table 2.1. The Radial Basis Function (RBF) kernel is commonly considered as the most powerful [97, 57] but linear kernels are best understood and simplest to apply [26]. Figure 2.3 depicts an exemplar classification problem in 2D together with the application of linear and RBF kernels. The selection of a suitable kernel type for a SVM classifier applied to a given problem will be discussed in Section 3.4, Chapter 3.

There are many SVM packages available, such as $mySVM$, $SVMdark$, $winSVM$, $LibSVM$, SVM^{struct} and so on. $LibSVM$, developed by Chang and Lin [24], is selected as the SVM package for face classification experiment. SVM implementation within $OpenCV$ which is an open source computer vision library is chosen for the person detection experiment. I used the approach of Chang and Lin [24], to perform the grid search by providing two parameters to optimize a SVM classifier: the cost C and a kernel parameter γ [24, 54]. The cost C is the

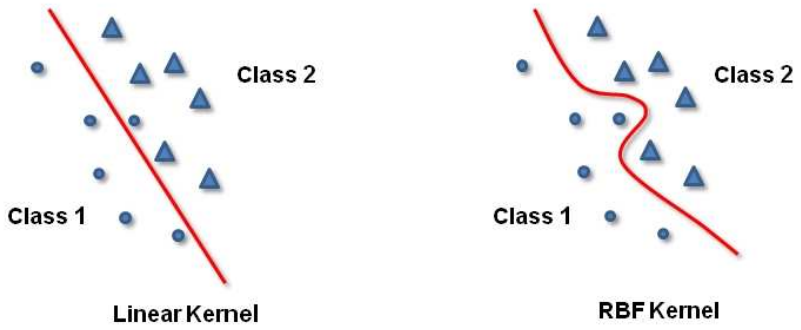


Figure 2.3: *Classification using linear kernel and RBF (radial basis function) kernels. The linear kernel is suitable for solving linear separable problems. If target classes are not linear separable, they have to be projected to a higher dimension space where they are linear separable or easier to be separated. This process is done using non-linear kernels, such as the RBF kernel. In this example, a linear kernel can not separate two classes without misclassification; a class 1 point is misclassified as class 2. By contrast a RBF kernel separates them correctly.*

penalty parameter on the error [97, 94, 20, 26]. A bigger Cost C means giving a heavier penalty on errors. γ is a parameter used to configure kernels; for different kernel, γ has different meanings. For example, in the RBF kernel, $exp(-\frac{||x-x_i||^2}{2\sigma^2})$, γ is $1/2\sigma^2$. There is no γ for a linear kernel. A smaller value of γ will produce a more general classification boundary [26].

2.4.2 Feedforward Neural Network

The artificial neural network is an algorithm inspired by biological neural networks [16, 86, 25, 77]. It has certain similarities to a biological neural network in that it calculates in a parallel and collective pattern. The basic calculation unit is the neuron which performs the function of a threshold. These units are connected together according to certain structures, for example separated by different layers, then they can achieve different purposes. Figure 2.4 is an example feedforward neural network.

The feedforward neural network is one of the simplest and most popular net-

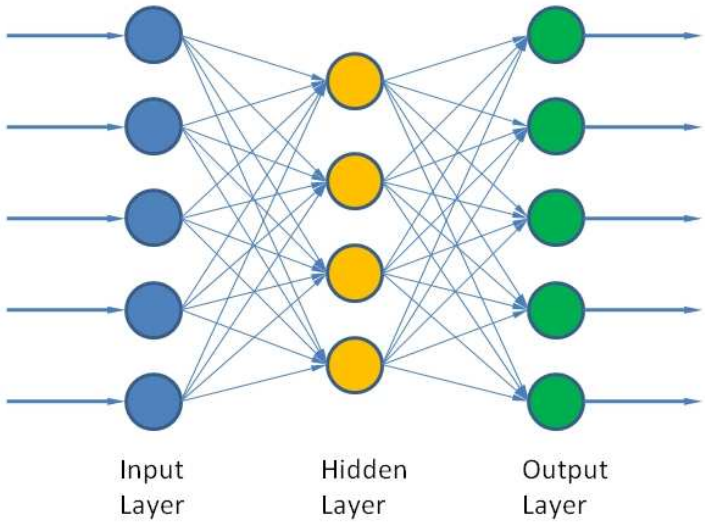


Figure 2.4: *A fully connected feedforward neural network with one hidden layer. The output node number is equal to the input node number. This kind of feedforward neural network is used to store associative memory in this research.*

works. There can be several layers within a network. The first layer is input layer. Following the first layer are several, mostly only one, hidden layers. The last layer is the output layer. Each neuron is connected to the previous layer by connections. These connections have different weights. Data are fed into the input layer, then multiply weights and go through hidden layers until reach output layer. There is no *feedback* connections. Hence this kind of network is called *feedforward* neural network [86, 77].

Weights are calculated by using a supervised learning method. Backpropagation is one of the most popular training methods for feedforward neural network [96, 30]. It is an abbreviation for "backwards propagation of errors". During training, this technique propagates errors from output layer to the input layer. It calculates the differences between the outputs of each node with expected outputs, then uses these differences to adjust weights until the output error reaches a chosen threshold or the training has reached a preset number of cycles. Here is the procedure of backpropagation training method: 1. Input a training pair, which normally includes an input vector and an output vector, to the neural net-

work. 2. Compare the difference between the desired output, which is the output vector, and the network's output. Calculate the error in each output neuron. 3. Calculate the local error for each neuron. 4. Adjust the weights of each neuron to reduce the local error. 5. Backpropagate errors through the network to previous layer. Adjust the weights of each neuron. 6. Repeat the steps until the input layer is reached. 7. If the output error can not reach the desired value, repeat these steps or stop according to criteria set by trainer.

The activation function used in this thesis is the sigmoid function which has the form:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

The error is calculated as:

$$E = \frac{1}{2} \sum_{i=1}^{P} e_i \tag{2.4}$$

Where:

$$e_i = (d_i - y_i)^2 \tag{2.5}$$

P is the total number of the output nodes of the neural network. Before feeding training samples into the neural network, the value of each pixel should be rescaled to between 0 to 1.

A feedforward neural network can act as an auto associator [82, 49, 72, 60]. For the simplicity of construction and training, it is chosen as an auto associator. When training an auto-associative feedforward neural network, both input and output are identical vectors. The trained feedforward neural network will learn the input vectors and can restore it even when the input is a partial vector.

2.4.3 Haar Like Feature Classifier

The Haar cascade classifier was firstly proposed by Viola [108] and later improved by Lienhart [70] with a primary application domain for the detection of

faces within the visual surveillance domain. The concept is to use a conjunctive set of weak classifiers to form a strong classifier - in this instance, a cascade of boosted classifiers applying Haar-like features (Figure 2.5). These Haar features are essentially drawn from the response of Haar basis functions to a given type of feature at a given orientation within the image (Figure 2.5). Individually, they are weak discriminative classifiers but when combined as a conjunctive cascade a powerful discriminative classifier can be constructed capable of recognising common structure over varying illumination, colour base and scale [108].

Figure 2.5: *Haar-like features. A. Edge features. B. Line features. C. Centre-surround features.*

This cascaded Haar classifier is trained using a set of multiple positive object (i.e. cell, face, car ...) and negative object images (i.e. no cell, face, car etc.). The use of boosting techniques then facilitates classifier training to select a maximally discriminant subset of Haar features to act as a multi-stage cascade. In this way, the final cascaded Haar classifier consists of several simpler (weak) classifiers that all form a stage in the resultant complex (strong) classifier. These simpler classifiers are essentially decision-tree classifiers (with at least 2 leaves) that take the Haar-like feature responses as input to the weak classifiers and return a boolean 0,1 pass/fail response. A given region within the image must then achieve a pass response from all of the weak classifiers in the cascade to be successfully classified as an instance of the object the overall strong classifier has been trained upon. Figure 2.5 shows the Haar-like features used as the basis for the weak classifiers in this approach.

The Haar cascade classifier thus combines successively more complex classifiers in a cascade structure which eliminates negative regions as early as possible during detection but focuses attention on promising regions of the image. This detection strategy dramatically increases the speed of the detector [108, 70]. It

has been successfully applied to solve some challenging computer vision tasks, such as real-time face detection [109] and pedestrian detection [110].

2.4.4 Cross Validation and Grid Search

A classifier has to be initially trained before use for classification. During training, samples of two classes, positive and negative, are presented to the classifier. After analysing these training data, the classifier learns how to classify these two classes. For example, a SVM classifier finds the hyperplane which separates two classes with maximum margin in a given dimensional space using a given kernel function, K. The aim of training the SVM classifier is to find a suitable kernel, K, and its associated parameters.

A commonly used parameter evaluation method is cross-validation [62]. This is a statistical method of testing hypotheses that keeps a subset of data as a test set and the remaining data as a training set. There are several commonplace cross-validation methods with *k-fold* cross-validation being the most popular [77]. This procedure divides original data into k subsets and keeps one of them as a test set while the remaining *k-1* subsets are used to train the classifier. This process is repeated until every subset has been used as test set and the mean correct rate is taken as the single estimation of the classifier. Cross-validation helps to eliminate the unilateral testing hypotheses suggested by the data [79] (i.e. over-fitting), so that it can give a comprehensive estimation of parameter configuration.

In order to utilize the cross-validation method to find the best configuration of a classifier, a grid search method is used [24, 54]. The grid search method performs cross-validation on training sets using all possible parameter configurations within certain ranges. This method searches a parameter space based on a "*try-all*" method, i.e. searching all the possible combinations and all ranges of different parameters specified by user to find the best configuration. For a SVM classifier, the configuration includes the kernel, K, and its parameters. The parameters of a kernel to be tuned are cost C and a parameter γ. This way, the kernel and its parameter configuration with best performance (normally highest detection rate

on training sets) is decided. In addition, the grid search method can also prevent over-fitting by finding the model with the best generalization [54]. Lin *et al* [54] find that trying exponentially growing sequences of C and γ is a practical method to identify good parameters (for example, $C = 2^{-5}$, 2^{-3}, ..., 2^{15}, $\gamma = 2^{-15}$, 2^{-13}, ..., 2^{3}). After this coarse search, a better search space will be decided and a fine search can be conducted to find the best configuration of parameters. I use their methodology to find the best kernels and their parameter configurations for SVM classifiers.

2.5 Related Work

A popular object recognition approach is to train a classifier using whole target images and recognise the target from background. Nattkemper *et al.* [80] designed a neural cell detection system (NCDS) for the automatic quantization of fluorescent lymphocytes in tissue sections. This system acquired visual knowledge from a set of training cell-image patches selected by a user. The trained system evaluated an image by calculating the number, the positions and the phenotypes of the fluorescent cells. The NCDS detected a minimum of 95% of the cells.

Using single-layer SVMs to detect objects has achieved remarkable success. Osuna *et al* [50] applied SVMs to face detection, later extending this application to a real-time system. El-Naqa *et al.* [32] applied SVMs to detect microcalcifications in mammogram images, which outperformed all the other methods considered within the study. Single micro-calcification is the input to the SVM classifier. A sensitivity as high as 94% was achieved by the SVM method at an error rate of one false-positive cluster per image. Breckon *et al* applied trained single classifiers to detect a person from thermal imagery and a vehicle from optical imagery [18]. These classifiers were trained to detect whole body of a person and whole front and back patches of a vehicle. They also developed a generic and robust approach for the detection of road vehicles from an Unmanned Aerial Vehicle (UAV) which is within the framework of fully autonomous UAV deployment for aerial reconnaissance and surveillance. Their approach facilitates the

real time detection of both static and moving vehicles invariant to orientation, colour, type and configuration by using multiple trained cascaded Haar classifiers (a disjunctive set of cascades). This approach successfully detected differing vehicle types under varying conditions in both isolated rural and cluttered urban environments with minimal false positive detection [17].

Eichner *et al* proposed a machine learning based system which offers continuous speed restriction monitoring for driving. It integrates both numerical speed limit sign recognition, robust cancellation sign detection together with GPS navigation for operation in all possible road scenarios (i.e. city, country, highways). It ensures real-time processing speed in all weather conditions with good performance [31].

Han *et al* [43] applied a machine learning based approach (cascaded Haar classification [108]) to classify three types of cystic lesions of the jaws: solitary odontogenic keratocysts, basal cell naevus syndrome associated odontogenic keratocysts, and radicular cysts [99]. In this application, the Haar classifier was trained using whole cyst images and a total correct classification rate of 86% was achieved. It showed successful detection of individual cell nuclei within the pathological slides in addition to promising classification rates on the cyst subtypes [43, 65]. It can detect whole cysts successfully but with a limited success rate in detecting occluded cysts. Han also applied SVM classifier to detect cells from complex background. The cell images were in different colour, different lighting, and cells were in different average sizes. The SVM classifier was trained with only limited number of samples but can detect cells of different sizes under varying lighting conditions. The average detection rate was above 90%. The above examples of single layer classifier based classification applications have achieved high performance on whole object detection but there are still some limitations on detecting incomplete or occluded objects.

Recently, multi-layer classifiers, most of them adopting an hierarchical structure, are becoming popular. Dillon *et al* [28] developed *Cite*, a scene understanding and object recognition system, which can generate hierarchical descriptions of visually sensed scenes based on an incrementally learned hierarchical knowledge

base. In this system, traditional feed-forward segmentation is augmented with knowledge-driven re-segmentation that closes the control loop on low-level vision processes to match the human annotation process as close as possible. *Cite* extends the conventional knowledge bases (including relational database models) to a fully hierarchical one with, in principal, no depth limitation. Multiple hypotheses are generated for each scene element, and these are resolved using a hierarchical extension to relaxation labeling. Behnke *et al* [10] proposed a hierarchical neural architecture for image interpretation, which was based on image pyramids and cellular neural networks inspired by the principles of information processing found in the visual cortex. One of the main features of the architecture is the transformation of the given image to a sequence of representations with increasing level of abstraction and decreasing level of detail. The binarization of handwriting has been implemented and has been shown to be perform well.

Especially, Heisele *et al* [53, 55] have presented a dual-layer SVM algorithm learning discriminative components (features) of objects similar to the system which will be introduced in this thesis. Figure 2.6 shows the system overview. In their system, component-based face classifiers were combined in second stage to yield an hierarchical SVM classifier. On the first layer, the component classifiers independently detected components of the face. On the second layer, the combination classifier performed the detection of the face based on the output of the component classifiers. There is a collection of 14 features from the face on the first layer. These features were chosen automatically from seed areas of a set of synthetic face images [52]. When their system identifies faces, features are located on an image by component classifiers within rectangular search regions around the expected positions of the components. The expected component positions were decided when extracting component samples from training images. Once identified, the features' maximum outputs within a search region and their locations are passed to the second layer in the following format: $(O_1, X_1, Y_1, ..., O_{14}, X_{14}, Y_{14})$. The second layer then decides if this component configuration belongs to a face. They also compared the performance between component-based (dual-layer) and global (single-layer) approaches [51]. Their experiments showed the potential of

dual-layer SVMs on pose and illumination invariance. My system differs from these other approaches in several main aspects. Firstly, my system uses a *feature map* rather than their special format vector as an intermediate representation, in order to retain detailed relations between features. Secondly, I employ an active attention mechanism [44, 45, 46]. The system training strategy is also different in my system. Their system trains each layer separately. My system joins two layers together to get the best training results from limited training samples. Chapter 3 will describe my system in detail.

In a real-world scene, an object recognition system should be able to find useful features of a target then, according to these features, decide if it should trigger an alert that it has found a meaningful target. Occlusion of a target can have negative influence on whole-image classifiers, as compared with component-based one. A simple single-layer approach has good performance of detecting targets when there is no occlusion existing. But in a real environment, especially in a battle field, it is not always possible to get whole target images. Therefore it is useful that the system has the ability to deduce the existence of an object by detection of its features.

Apart from the feature-based detection, an ideal system also should be able to find other related features in likely areas when a feature has been detected. This mechanism will take advantage of learned knowledge about the target and help to improve the detection speed and confidence. Many efforts have been done to handle the influence of occlusion occurring in detection which is now summarized. Rensink *et al* [90] have found that early vision can use monocular cues to rapidly complete partially-occluded objects. Visual search for easily-detected fragments becomes difficult when the completed shape is similar to others in the display; conversely, search for fragments that are difficult to detect becomes easy when the completed shape is distinctive. They concluded that it is only the completed structures–and not the fragments themselves–that serve as the basis for rapid recognition.

Williams *et al* [111] investigated the reconstruction of 3D surfaces when occlusion exists. They generalized the conventional visual reconstruction to include

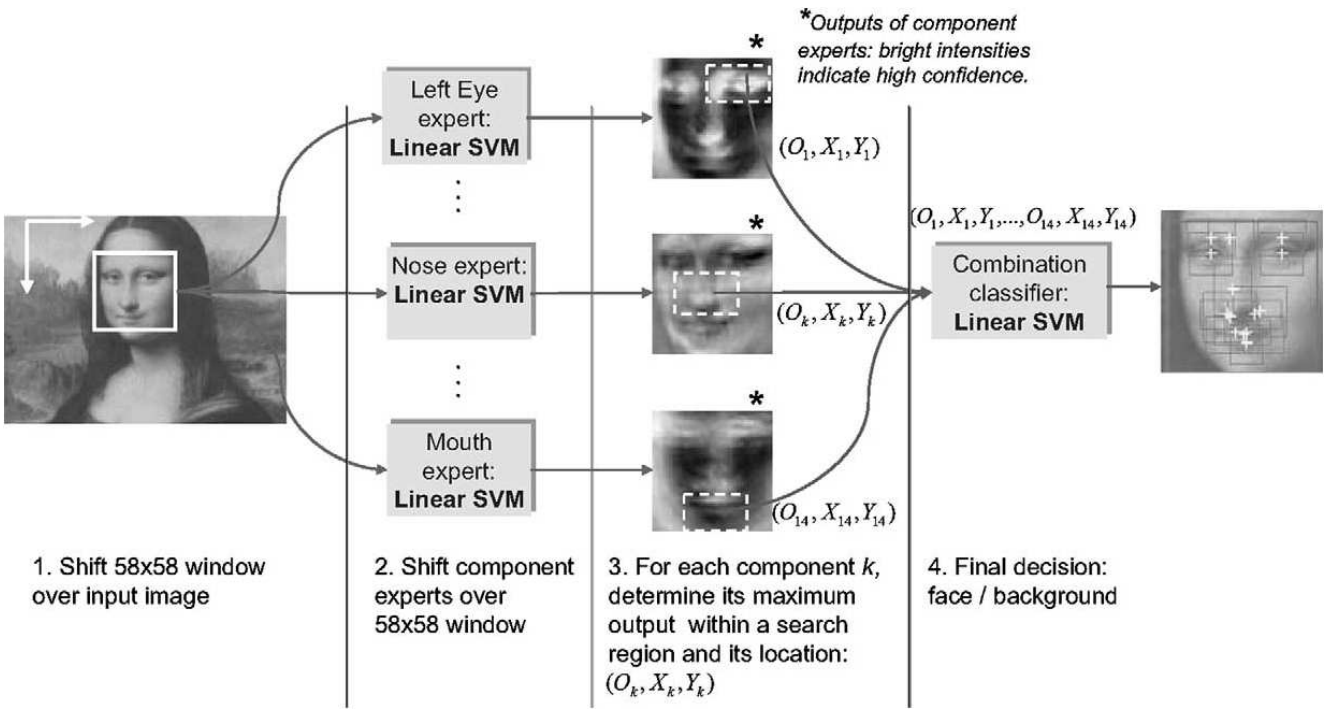


Figure 2.6: *System overview of the component-based face detector using fourteen components. Adapted from [53].*

both visible and occluded forward facing surfaces and found that the topology of three dimensional scene structure can no longer be taken for granted, but must be inferred from evidence provided by image contours.

Elgammal *et al* [33] presented a framework that uses maximum likelihood estimation to estimate the best arrangement for people in terms of 2D translation that yields a segmentation for the foreground region. They conducted occlusion reasoning to recover relative depth information and presented a method to automatically initialize a person model which is based on segmenting the body into regions in order to spatially localize the colour features corresponding to the way people are dressed and learn them before the occlusion. Tan *et al* [103] addresses the influences on face recognition caused by partially occluded face images by extending a previous local probabilistic approach presented by Martinez, using the self-organizing map (SOM) instead of a mixture of Gaussians to learn the subspace that represented each individual. Their experiments show that the proposed method exhibits high robust performance against the partial occlusions and variant expressions.

The hierarchical component-based object recognition system provides a suitable platform to recognise objects from complex real-world scenes. It represents objects with their components. By using this representation, the detection should be robust even when there is only partial image of an object available. This representation also provides a better chance to apply an attention mechanism to speed up the recognition of an object, by focusing on the important parts of an image.

Chapter 3

A Multi-Level System for Object Recognition

This chapter describes the object recognition system developed for this thesis. This system is a hierarchical component-based computer vision system and has a name '*Cengji*'. I want to use this word to emphasize the important characteristic of this system, hierarchical, and distinguish this system from other existing systems. This chapter begins with an overview of the *Cengji* system, and then describes it in detail through Section 3.2 to 3.4.

3.1 Overview of the *Cengji* System

Figure 3.1 provides an overview of this system. It is an object recognition system, taking an image as its input and returning a classification of that image as its output. Internally, there are two layers. The first consists of a number of feature detectors, or classifiers, whose job is to recognise specific features within the image; each classifier is tailored to recognise one kind of feature, such as an eye within a face. The features identified by all of the classifiers are placed together onto a feature map, as shown in Figure 3.2. The feature map, which will be introduced in detail in Section 3.2, is a grid, of the same size as the input image. The elements of the grid encode the label for the feature which overlaps that

position.

The dynamic behaviour of the complete system captured in Figure 3.1 is as follows. First, the feature detectors are scanned over the image in turn, until one of them identifies their feature; for example, the eye classifier will report when it locates an eye in the image. Once the first feature has been located, the feature is placed onto the feature map, and passed to the autoassociator. If successful and the attention mechanism is applied, the autoassociator returns a prediction for the complete feature map, encoding locations for the remaining features. These predicted locations are then passed back to the feature detectors, which attempt to identify the predicted features. If the attention mechanism is not applied, other feature classifiers will be scanned over the image just like the first classifier does. When all features have been located, the complete feature map is passed to the second layer, which classifies the feature map. This process stops when all the feature classifiers have performed a complete scan of the input image, or located a feature.

I have implemented the feature classifiers and final classifier using Support Vector Machines. The autoassociator is a neural network, trained using back-propagation.

3.2 Feature Map

A significant characteristic of *Cengji* is the introduction of a *feature map*. The phrase *feature map* has been seen in previous applications [112]. Here, I use it to name the intermediate representation which conveys key information of individual features and describes how they compose a higher level object. 'Feature' in this *feature map* refers to components forming higher level objects.

The feature map is an abstract image of an object, keeping only the essential information. When constructing a feature map of an object, each feature in the input image forming this object is given a label (a number). Each pixel belonging to the image patch containing this feature is replaced with this feature's label. Pixels not belonging to any feature are set to 0. So a feature map keeps the

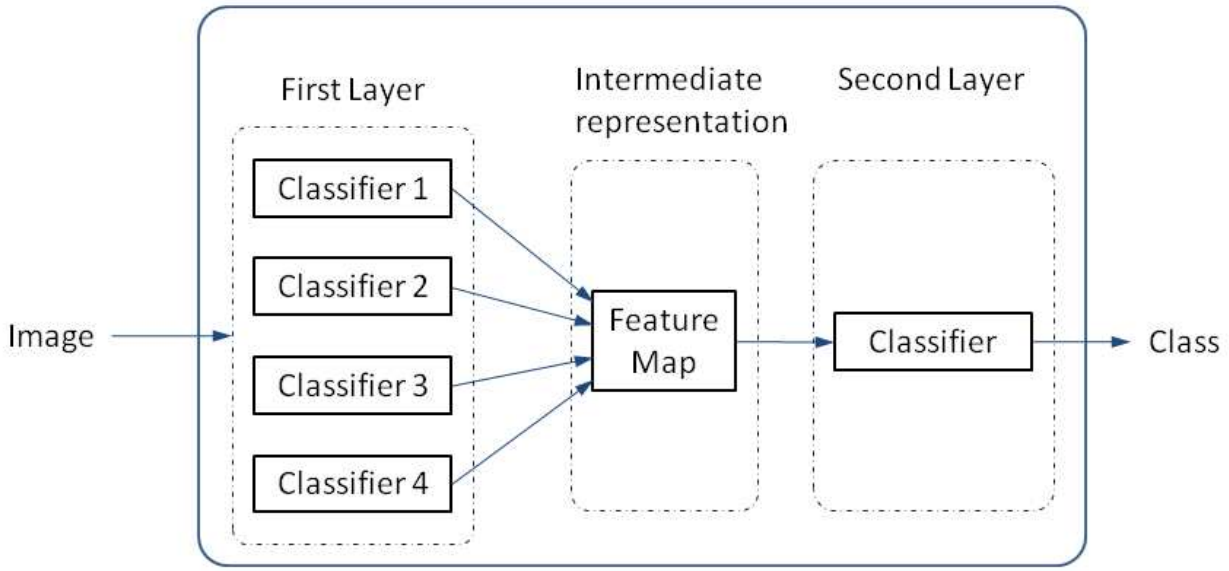


Figure 3.1: *Hierarchical structure of* Cengji. *In the first level are classifiers for basic components. The intermediate representation holds position information for these components, and this representation is converted into a class in the second layer.*

geometric information of each feature and the spatial relations among features. Potentially the feature map can also group objects to form a higher-level object, in a hierarchical manner.

For example, Figure 3.2 is a feature map of a face. Each feature in the face map is given a label: left eye is 4, right eye 1, nose 3 and mouth 2. The background is 0. Each number in the map corresponds to a pixel in the face image. So the map matrix has the same length and width as the face image, and the matrix of a feature also contains the size information of this feature. There is overlapping between two different features. The system will not be confused by this because I can arbitrarily design the feature detection process proceeding always in the same sequence, for example, for a face: right eye $\rightarrow$ mouth $\rightarrow$ nose $\rightarrow$ left eye. The overlapping sequence will be embodied by the feature map. This ensures that the overlapping between two features is always consistent.

3.2.1 Feature Detection

In computer vision and image processing, feature detection refers to methods that find useful information from images and decide which image pixel is a part of the target patch carrying that information [39]. These features will be subclasses or a kind of abstraction of the image to be detected. They might be edges, corners, interesting points, blobs, regions and so on. In *Cengji*, a feature is defined as a component of a meaningful object. It refers to a patch of image pixels. For example, we can say an eye is a feature belonging to a face. So one of the feature detectors might be detecting an eye from an image. Of course, this does not mean this system ignores the detection of features like edges, corners and so on. These kinds of features can also help the system to recognise objects in complex images, and could be included.

Feature detection is a relatively low-level image processing operation. Object recognition starts from the detection of individual features. Therefore, feature detection performance has a fundamental influence on the whole system.

Feature detection identifies a feature from a patch of an image. A computer has to know what a feature looks like. In general, there are two main methods.

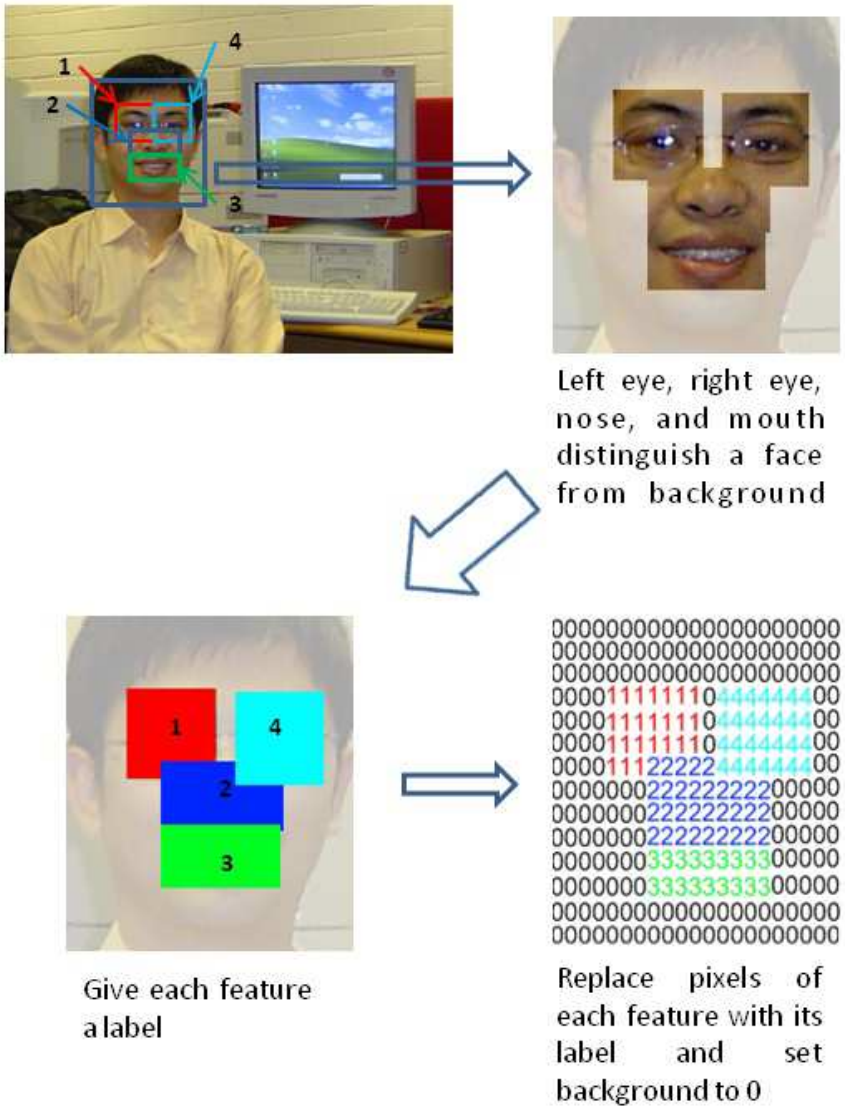


Figure 3.2: *A feature map of a face. A face is distinguished from a complex background by several features: two eyes, nose, and mouth. These features are composed together to form a face. These features and the relation of locations and sizes are unique to a face. A feature map conveys this information.*

The first is to describe the feature manually. It needs an expert using a special programming language to code the feature, line detection is a good example. The second is to teach the detector by presenting it training samples, using a machine learning approach. By using machine learning algorithms, a computer program automatically learns features from those training samples. These two methods both have their merits. Recently, much emphasis has been paid on the latter because it is often difficult to describe a mass of objects manually. There have been many feature detection algorithms following the second approach. In this thesis, the second approach is taken, using machine learning algorithms to perform object classification.

An important task *Cengji* must fulfill is to scan the input image and identify individual features in it. When an input image is presented to this system, feature classifiers will scan the image to find features. During this process, classifiers classify each scanned sub-image patch, and the system will give each sub-image a label. Detected features are then passed to the next stage.

The scanning can be simple, pixel by pixel, or smart, guided by higher-level knowledge. A commonly used method is to use grids or certain templates to scan across the image pixel by pixel to extract data to the classifiers [108, 50, 43]. When one classifier finishes its searching over the image, another classifier then repeats the same scanning for the next feature.

The crude pixel by pixel scanning for features can be computationally expensive and time-consuming. Some knowledge about the target object may be used to guide the feature detection stage to reduce this time. When the system has knowledge of the spatial relations among features and some features have been detected, the expected relation between features mean only certain parts of the image need to be searched for the remaining features. The human visual system makes extensive use of contextual information for facilitating object search in natural scenes [105]. This suggests that the movement of eyes is driven by target locations which are deduced by applying learned knowledge about the scene. Using or imitating these processes can help the hierarchical system effectively search for features. The autoassociation and attention mechanism, which will be

introduced in section 3.2.2, provide such a process. When the system finds one of the features, an expectation of other relative features is generated and tells the system possible locations and sizes of those features. At the same time, the system can ignore less relevant information and concentrate on searching important parts of the image for relevant features. This strategy can improve system scan performance.

The process of feature detection, especially of scanning, can use either a general approach or a domain-specific one. For example, the pixel by pixel sequential scanning is an often used general approach of scanning. It can work for many scanning tasks but may not be the most effective. When adapting this system to a domain, the scanning strategy can be adjusted to have a better performance, i.e. speed and precision. The adjustment can include domain-specific scanning order and locating important image areas. The adaption of the feature detection process to be domain-specific will be illustrated in Chapter 4 and Chapter 5.

3.2.2 Using Attention Mechanism

Cengji is designed to take advantage of an attention mechanism such as used by the human vision system to optimize the search for features. So far, humans have not had a thorough understanding of their own vision system, but found some important mechanisms, which can improve computer vision systems' performance.

Associative Memory

An autoassociative memory is a structure storing learned patterns, which can map an input pattern to a most similar stored pattern [74, 66, 67, 40]. When this system is used to help feature detection, it needs information about the features to be searched. This information includes what these features are and where they can be located. A feature map comprises all this information. So for *Cengji*, the feature map is the pattern that an autoassociative memory learns and stores. A well trained autoassociative memory can retrieve a whole feature map stored

when the input is only a partial feature map which only contains feature having been found [74].

Attention Mechanism

An attention mechanism is a way to help the object recognition system focus on features relevant to the target object without spending time on irrelevant information. The core of the attention mechanism is the autoassociative memory. The autoassociative memory forms a whole feature map using only those features found earlier. This generated artificial feature map will tell the system what other possible features may exist and where they might be. This process is a mapping from a partial feature map (containing only the found feature) to a learned whole feature map stored in the autoassociative memory, which has the smallest difference from the partial feature map.

The attention mechanism helps *Cengji* search for features in a 'smarter' way. For example, when the system is scanning for a face and an eye is found, it will automatically select feature classifiers and go to the possible positions of other features, for example, nose and mouth, under the instruction of the associative memory. These feature locations are indicated by their top left corner coordinates. The system will conduct a local search at those proposed locations. The search area size can be decided arbitrarily or experimentally to reach the best balance between speed and precision. Currently, the search areas are decided by the user. For example, when the system is led to the proposed location of a mouth, the search area will be a 20×20-pixel area (over which the top left corner of the feature can move) whose geometric centre is the proposed location.

There are some machine learning algorithms that can be used to implement the attention mechanism. For example, a feed-forward neural network trained with feature maps both as input and output or a KNN algorithm can act as an associative memory. Figure 3.3 is a sample. Each box in the picture represents a feature holding the pixels of some label value. The box in the left side represents a right eye. When it is input to the autoassociator, the autoassociator generates a partial feature map of the face. Then, according to the feature maps stored in

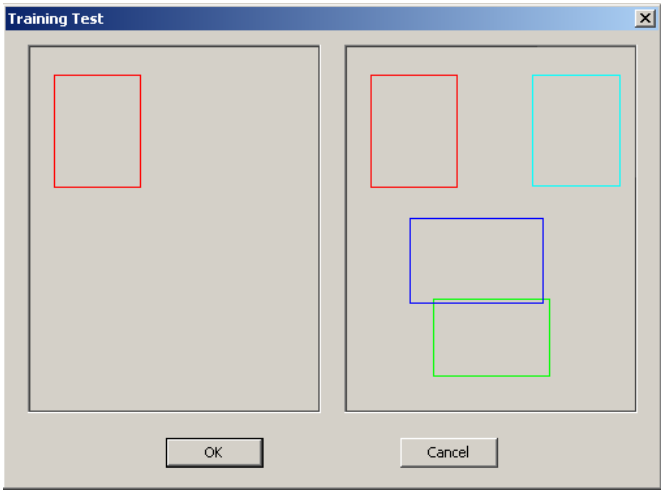


Figure 3.3: *An autoassociator generates a complete face feature map according to a partial face feature map.*

the associative memory, the autoassociator automatically completes this partial map and retrieves a whole artificial feature map of a face.

Work Flow of *Cengji* Using Attention Mechanism

When applying the attention mechanism, this system uses knowledge of the target object to focus only on certain features and possible regions which may contain these features, without paying attention to every part of the image. This knowledge is stored in the autoassociative memory and includes location relations among features and possible locations of each feature in the target object. This information is conveyed by the feature map. A feed-forward neural network, trained using feature maps, can learn and memorize knowledge about the target objects.

Figure 3.4 is a flow chart showing how a dual-layer *Cengji* uses its attention mechanism. The first layer of this system starts from searching for the first feature. This feature can be any of those forming the target object and can be found by basic pixel by pixel search method. When it is found, the system will use it to form a partial feature map. Then the attention mechanism uses its autoassociative memory and this partial feature map to produce an artificial

whole feature map. This artificial feature map will tell the system what other relevant features may be present and where these features are most likely located. According to this information, the system selects corresponding feature classifiers and goes to possible locations to search for these suggested features. If a feature classifier finds an expected feature in the possible region, then it will report the feature label and its coordinates to the system. Otherwise, this expert reports nothing to the system, and the next expert will perform search. This process will stop when every classifier completes its search. *Cengji* then forms a final complete feature map according to search results reported by the feature classifiers and sends it to the second layer. The second layer judges whether this feature map has the structure that a target object has and gives the final output. The attention mechanism speeds up feature search and improves feature location precision by directing classifiers to locations with a higher likelihood of containing features. There are two layers in this flow chart.

The autoassociative memory is stored in a feed-forward neural network. Its input is a partial feature map which only contains one or a few features. Each pixel within a feature map has the label value of the feature which it belongs to. These values have to be scaled to between 0 and 1 before the partial feature map is fed into the autoassociator. Different inputs, i.e. the locations of the right eye vary, will produce different outputs which means the locations of other features might be different.

Locating proposed features may be difficult due to the performance limitation of the associative feed-forward neural network. In particularly separating each feature type from its real number. Some features can be directly isolated, but the identification of some features will be difficult because the associator can not discriminate them well. Therefore it needs some generic method to discriminate those overlapping features. For example, in the face feature map, it is difficult to separate the mouth and nose. In order to locate these features, the output raw feature map is thresholded to a binary image. I find the smallest rectangle containing the lowest image patch and use its coordinate to decide the locations of proposed mouth and nose. The method of feature separation is domain dependent

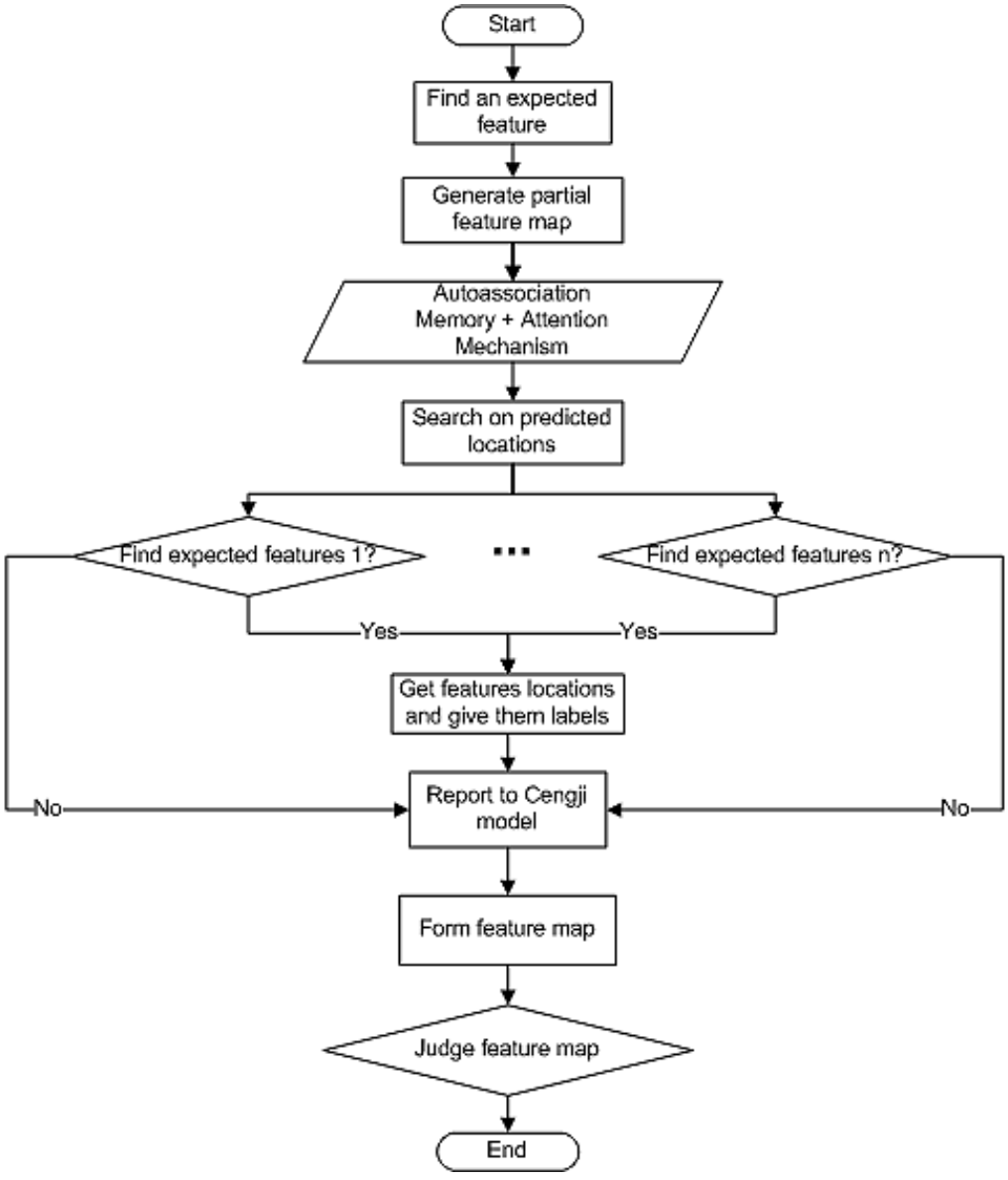


Figure 3.4: *Flow chart of a dual-layer* Cengji *with attention mechanism.*

and must be adapted for each target. The following is the procedure of applying the attention mechanisms, taking face classification as an example.

A. Find the right eye and generate a partial feature map;

B. Feed this partial feature map into the associative feed-forward neural network and produce a raw feature map, see Figure 3.5;

Figure 3.5: *Step 1: associator produces a raw feature map using a partial feature map.*

C. Apply image processing methods (erosion, dilation etc.) to generate binary image patches, see Figure 3.6;

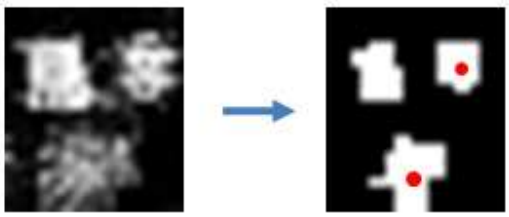


Figure 3.6: *Step 2: process the raw feature map to be binary.*

D. Find the smallest rectangles containing each patch and decide patch label (i.e. feature label according to the major values within the raw feature map). The location of each proposed feature is geometrical centre of the rectangle. To separate the mouth and nose, I use rectangle corners to determine the locations of proposed mouth and nose, see Figure 3.7;

E. A feature map is proposed. See Figure 3.8.

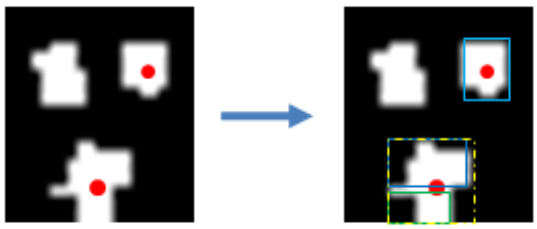


Figure 3.7: *Step 3: find locations of features.*

Figure 3.8: *Step 4: produce a full feature map.*

3.3 Algorithms Used and Data Format

Cengji is designed to be an open system. Every part of it is a module whose algorithm can be selected according to the algorithm's performance and system requirements. In this thesis, I used SVM as the main classification algorithm and feed-forward neural network as the associator, as described here.

3.3.1 Support Vector Machines

In this experiment, the system takes advantage of the high classification performance of Support Vector Machines (SVMs) to perform feature detection and object classification. There are several SVM packages available, such as $mySVM$, $SVMdark$, $winSVM$, $LibSVM$, SVM^{struct} and so on. $LibSVM$, developed by Chang and Lin [24], was selected as the SVM package for face classification experiment. An *OpenCV* SVM implementation was selected for thermal person detection. Both are widely used SVM packages and are easy to integrate into a system.

3.3.2 Feedforward Neural Network

A feed-forward neural network can act as an autoassociator when the input vectors and output vectors for training are exactly the same [63, 72]. For *Cengji*, a two layer feed-forward neural network, which comprises one hidden layer, is used and trained using back propagation. When it is trained, each feature map is both the input and the output. This feed-forward neural network remembers these feature maps after training. So that when a partial feature map is presented, this autoassociator can produce a completer feature map containing other expected features.

3.3.3 Data Format

This face classification experiment uses 8 bit gray scale bitmap (bmp) as input. The bmp format is one of the simplest image formats. It is widely used in research. There are 3 or 4 parts in a bmp file. The first part is bmp file header. The second part is bmp information header. An optional palette follows. It is optional because there is no palette unless we are dealing with an indexed colour image. The last part is the image data. The number of bytes in each line must be divided exactly by 4. If it can not, 0x00 must be appended to this line until it can be divided exactly by 4.

Bmp image files have to be converted to the format that the *LibSVM* and *OpenCV* SVM can read. Following lines are data samples:

1 1:0.423529 2:0.505882 3:0.592157 ... 10:0.674510 11:0.674510 ... 725:0.227451

2 1:0.568627 2:0.568627 3:0.596078 ... 10:0.654902 11:0.670588 ... 725:0.615686

The first number is the sample class, which is the target value. In classification mode, the target value denotes the class of the example. It is an integer to indicate which class this sample belongs to. Following it are sample element indices and element values. Element indices start from 1 and increase. Element values can be integer or decimal. Normally these values are scaled to -1˜1 or 0˜1 to avoid

calculation problems [54]. Value pairs are separated by a space. Feature/value pairs are ordered by increasing feature number. Features with value zero can be skipped. A program was created to convert image files to the specific format using Visual C++.

3.3.4 Software Used

In this thesis, the SVM package for the face recognition experiments is $Libsvm-2.84$. $OpenCV-1.0$ provides the SVM implementation for the extended thermal person recognition. *Visual C++ 6.0* is used for the face classification experiments. *Visual Studio 2005* is used for the person detection experiments.

3.4 Training of *Cengji*

To create *Cengji*, one of the most important procedures is the training of the system. A classifier has to be trained before use. In this procedure, training sets for each individual classifier are created. The quality of the training sets determines the performance of a machine learning system. A well-constructed training set can demonstrate the best performance of an algorithm, on the other hand, a bad one can limit the algorithm's potential and lead research in a wrong direction by discouraging use of an algorithm. So it is important to have a proper training methodology.

There are more than one layer and more than one classifier existing within this system. It requires different training datasets for each layer and classifiers within these layers. In the lowest level, classifiers are trained to detect features which are the basic components forming higher level objects. Their training sets are image patches containing a feature. In the higher level, there are feature map classifiers. Their training sets are feature maps generated by the trained lower level. These training sets are generated automatically.

I did not use the training set creation method proposed by Huang *et al* [55], which considered the creation of training sets for two layers as two independent processes. They created training sets for each layer separately. There was no

direct connection between two layers when creating training sets. I consider the dual-layer system as an integrated system. The creation of feature training sets is tightly connected with the creation of the feature map training set. Therefore the two layers have mutual influence on each other.

The training sets for the higher level are determined by the lower level. An ideally trained first layer can precisely locate basic features and reject non-features in an image. At the same time, a well trained higher level can also help to improve the training sets for the lower level in return by giving the correct locations of targets in the lower level.

Considering the complexity and enormous diversity of the negative samples, it is difficult to create perfect training datasets, especially a negative training set. When training a machine learning system, several challenges emerge. First, what is the best performance that a learning method can reach using the available data? There are some documents describing how to decide the training standard for a learning task [102]. This limiting performance can be a trade off between the false positive detection rate and the true positive detection rate. This trade off can be found by using the ROC curve [36, 68] of a learning system. It is located at the turning point at which the increasing speed of correct detection rate starts to become slow with respect to the increase of false positive detections. Second, how to select training samples from image database? Sung [102] introduced *active learning*, which uses information derived from its current state and prior knowledge to gather useful examples, to optimize the example selection. *Cengji* is trained hierarchically from the lowest level to the highest level. I use trained lower level to form training datasets for higher level, then use the higher level to improve the lower level training database in return. This is a closed loop and is repeated until the system becomes stable.

Bigger training datasets usually give more comprehensive sampling on the domain in which a learning system is to learn and generalize, and have higher likelihood to include key samples. But it is impossible to include all possible samples into training sets in some real-world applications. For example, if I exhaustively sample a 84×96 non-face image into 25×29 non-eye sub-images,

1. *Initialize training sets for each feature classifier.* Extract lowest level features from positive training images to form the initial version positive training datasets for the first level. Randomly extract non-features from negative training images and non-feature areas of positive training images to form the initial version of negative training datasets.
2. *Create first version of feature classifiers.* Train classifiers using the initial version training datasets.
3. *Improve feature classifier training sets further.* Scan trained classifiers across both positive and negative training images to generate training datasets for feature map classifier in the second layer. Train the feature map classifier and join the first layer and the second layer together to scan both positive and negative images. Check misclassified images. Add non-detected features to positive feature training datasets and false positive detections to negative feature training datasets.
4. Optimize each classifier using new training sets and train feature classifiers on new training sets.
5. *Get the final version of training sets for all classifiers.* Repeat step 3 and 4 until the correct detection rate starts to become stable. The creation of training sets is finished.
6. *Use the final system to detect test sets.*

Table 3.1: *The procedure to create training database for a dual-layer* Cengji.

there would be 3963 sub-images. Therefore, for a 350 images training dataset, there will be 138,7050 sub-images if all these 350 images are sampled. This large sample number would make training and classification infeasible. Some method must be used to find key samples to limit the size of the training sets. Table 3.1 shows the procedure of creating the database and verifying the training result for a dual-layer *Cengji*.

An important procedure of training a SVM classifier is to find the proper kernel and its optimized parameters. During training, samples of two classes are presented to the classifier. After analysing these training data, the classifier finds the hyperplane which separates two classes with maximum margin in a

given dimensional space using a given kernel function, K. The aim of training the SVM classifier is to find a suitable kernel, K, and its associated parameters. Cross-validation [62, 77] is used for parameter evaluation.

3.5 Summary

Cengji is a component-based computer vision system. It represents an object using its constituent features. A feature map is a medium to convey the object's structure directly and preserve all relative positions among features explicitly. This representation is inspired by the way that the human vision system stores knowledge of objects. The benefit of adopting the hierarchical structure is clear that it can effectively speed up the system by searching a smaller area. It also provides a chance to introduce the attention mechanism into the system. By applying these mechanisms, this system can focus its computation only on those salient features forming an object, therefore a higher speed can be achieved. Also less noise will be introduced as the attention of the system is focused on a limited area.

The next two chapters apply this system to two different domains in image classification and object detection.

Chapter 4

Application to Face Detection

In this chapter, I will apply *Cengji* to classify faces. Face detection is one of the important and challenging areas of object recognition. Much effort has been done to improve the performance of face detection systems [113, 115, 109, 103, 44]. Face detection involves identifying image patches containing a face, which might be in different attitudes, illumination and expression. There are broad applications of face detection in security surveillance, vision-based human computer interaction, ID identification, and so on. Face classification is the first step of face detection, which is to classify if an image patch is a face. The variation of face pose, presence or absence of structural components, facial expression, occlusion by other objects, image orientation, and imaging condition present challenges for face detection [113]. There are many face databases, and much work has been done in this area. So face detection is a good choice to verify the framework of this hierarchical system and test its performance.

4.1 Applying *Cengji* to Face Classification

A human face comprises several prominent features, especially two eyes, a mouth, and a nose. These features form a face according to certain geometric and spatial relations. These relations include distances between different features, their relative locations, the proportion of feature sizes to higher level objects, and so on.

They can be represented using a two-level hierarchy. The bottom layer of this hierarchy is composed of individual features. The top layer is a face classifier. Figure 4.1 is the basic structure used for face classification. In this figure, the attention mechanism is not shown.

SVM classifiers are used to detect features and relations among them because of the high classification performance of the SVM [26, 50]. There are four individual SVM feature detection classifiers, two eyes, nose, and mouth, composing the first layer. This layer searches for features in the input image. By detecting these features, the system finds their locations in the image and labels each feature. Using the features' geometry and position information, the system generates a feature map, which comprises the structure information among features. The second layer is used to identify the structure of the target object formed by those detected features. Its input is the feature map created by the first layer. The output of this system is a class label identifying if the input image is a face.

When this system classifies faces, feature classifiers in the first layer scan the input image according to a certain strategy. The search for a feature is achieved by moving a grid over the target image to extract image patches for classification. The grid can have different sizes from the feature size for training. The system just needs to resize the image patch extracted by the searching grid to the training size before classification. In the face classification experiment, which is mainly to verify the framework of *Cengji*, the feature sizes and face sizes are fixed as for training, so the grid sizes keep fixed as well. When the feature grid moves to a pixel, it extracts all pixels in an area which starts from that pixel and has the same width and length as the feature. These pixel values are then sent to the feature classifier to calculate whether these pixels form a feature.

Search of a feature stops when the first sample of this feature is detected. When a classifier finds the first feature sample, the movement of the grid stops. This feature's coordinate and label are used to form a feature map. The classifier can also continue to search the whole potential area and find all possible features, then the system takes the features with the highest probability to form a feature map. In this thesis, *Cengji* takes the first detected feature to form the feature

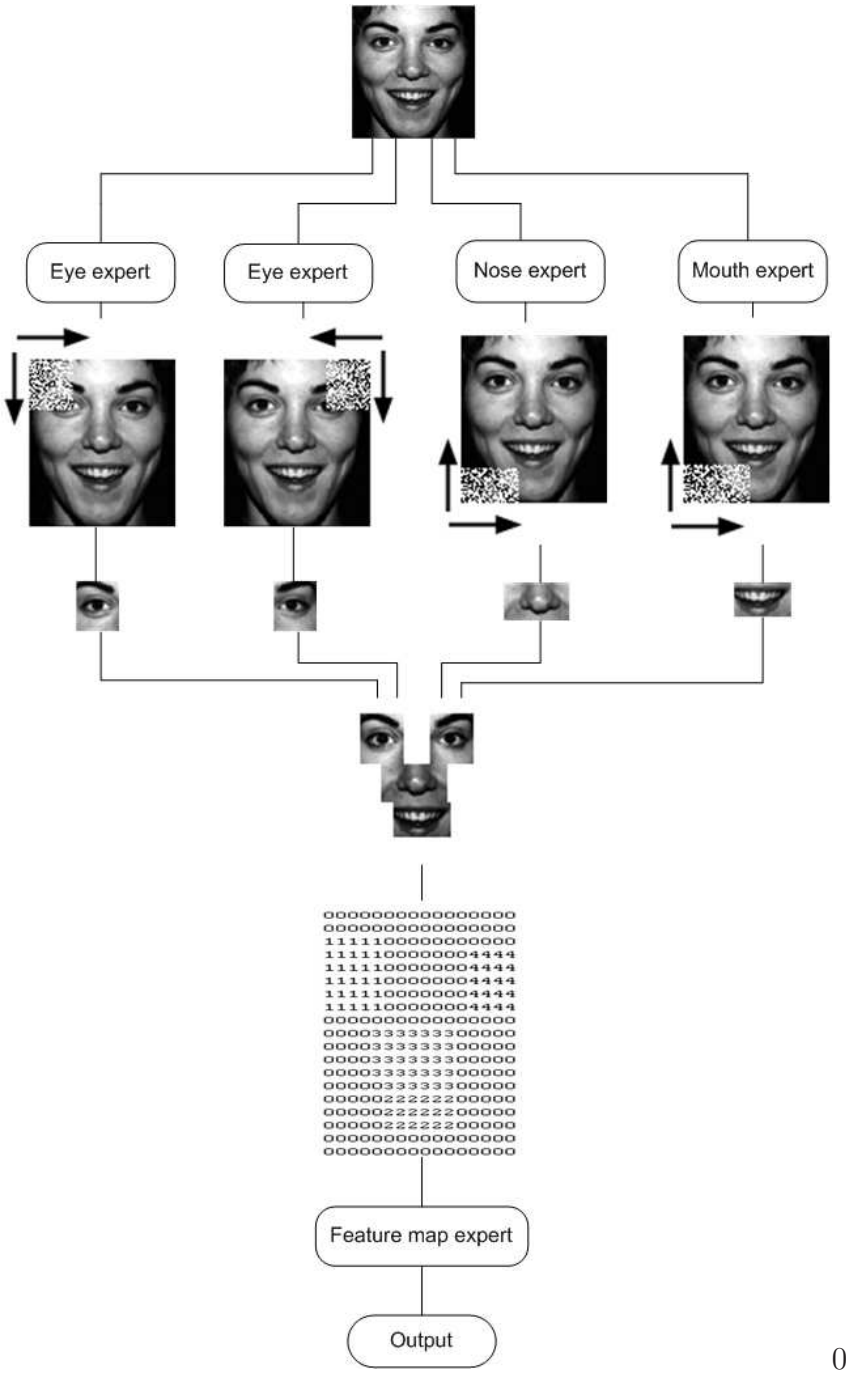


Figure 4.1: *Structure of dual-layer* Cengji *without attention mechanism used for face detection.*

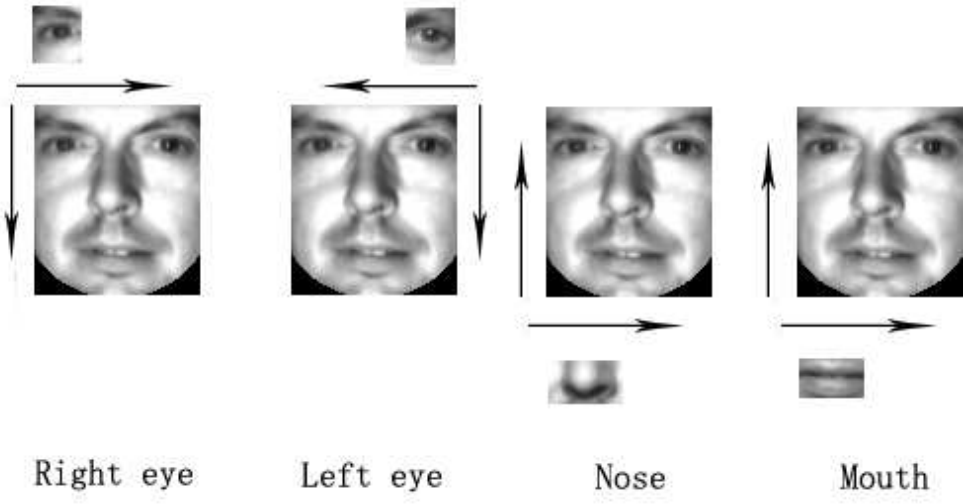


Figure 4.2: *Feature classifier movement routes for* Cengji *on face classification. To detect a left eye, the eye classifier moves from right to left and top to bottom. For the right eye, the classifier moves from left to right. To detect mouth and nose, two classifiers move from bottom to top and left to right.*

map.

For face classification, I designed a feature scan strategy for this system. Figure 4.2 shows the movements for each feature classifier. Two eye classifiers were trained with the same training sets. This is because the difference between the left and right eyes is small. They are identified by different search routes in the potential target area. During the scan, the grid can 'jump', which means after the grid extracts data and the feature classifier does not detect a feature at one position, the grid can ignore neighboring pixels and jump to another position to start new data extraction and detection. This can save scan time for the system without reducing the detection rate significantly.

When applying the attention mechanism, this system uses learned knowledge stored in the associative memory to speed up search and improve feature location precision. For a face, the stored knowledge can be that the nose is above the mouth but below the two eyes, two eyes are almost at the same altitude, and so on. All this information is conveyed by the feature map. A feed-forward neural

network, trained using feature maps, learns the face feature maps. Figure 4.3 shows how this system detects features using the attention mechanism.

The attention mechanism helps this system focus attention on areas with more likelihood containing other features when one feature is detected. The first feature to be searched can be any of those features within the first layer. The search for the first feature is the same as that in Figure 4.1. Once the first sample of this feature is found, the feature classifier will report its location and label to the system to form a partial feature map only containing this feature. The system then uses this partial feature map and the autoassociative memory to produce an artificial feature map containing other features. This generated feature map will tell the system other related face features and their possible locations. Then the proper classifiers can be selected and searching areas can be decided. The search within these proposed areas is the same as that of the first feature search. Once all the classifiers finish their feature classification, their search result will be composed together to form a final feature map. The second layer of this system will decide if it is a face's feature map. The procedure of applying the attention mechanism has been introduced in Figure 4.15, Chapter 3.

4.2 Source Database

The main purpose of the experiments on face classification is to understand the performance of the dual-layer *Cengji* when the attention mechanism is absent and present. This will display the impact of the attention mechanism. As a single-layer structure is popular and easy to implement, it is also useful to compare the difference between it and the dual-layer system to see how a dual-layer system performs. A face database, which comprises a face training set and a non-face training set, is constructed. This database is used to train the single-layer SVM classifier directly and to create training sets for feature and feature map classifiers in the dual-layer system.

In order to focus on the understanding of systems' performance of different classification systems and simplify the training process, faces in this database are

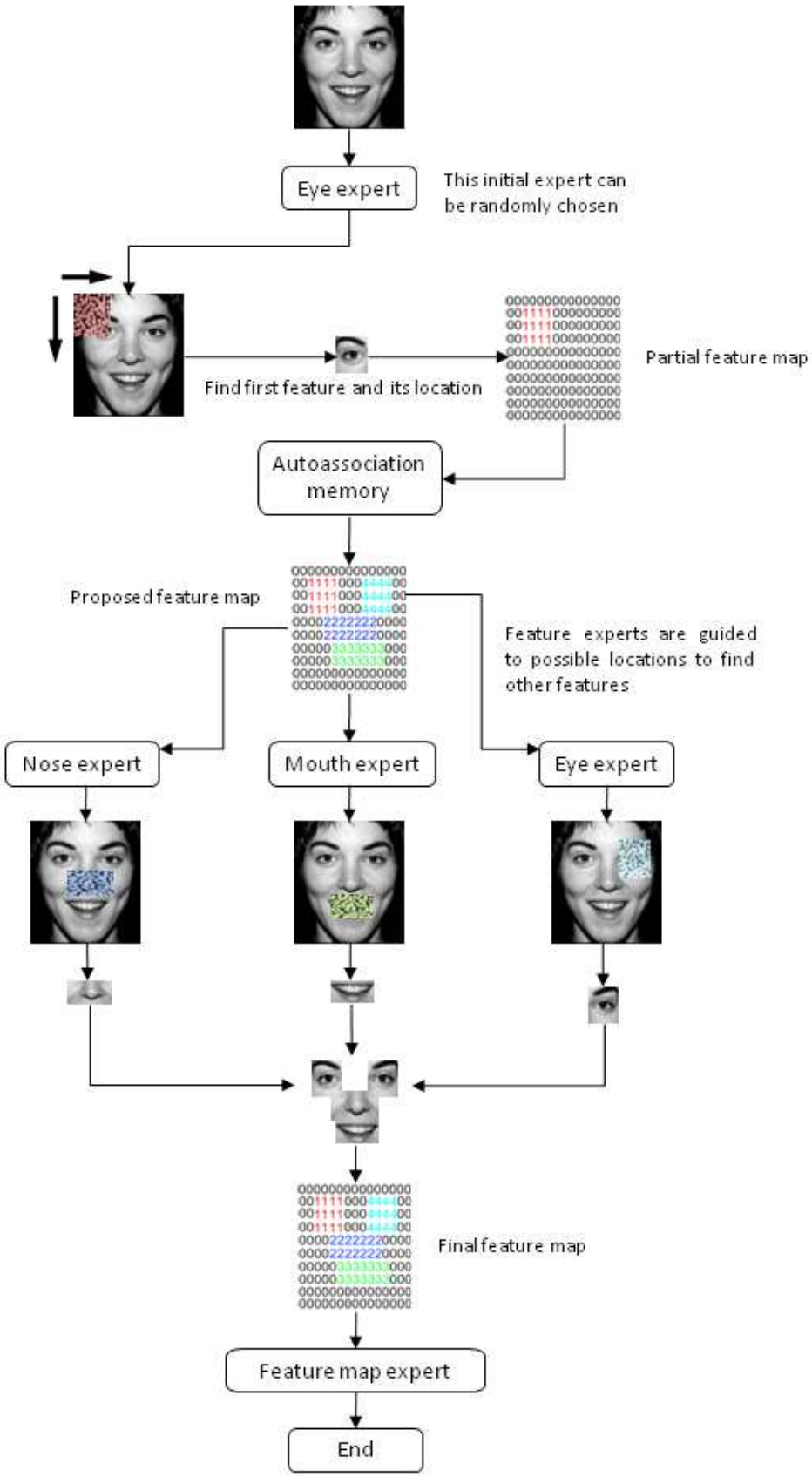


Figure 4.3: *Structure of the dual-layer* Cengji *with attention mechanism used for face classification*

mainly frontal, that is the face attitude extremity in all directions is no more than $\pm 30^o$.

All face and non-face examples are available in public databases. I took the following source databases to construct the face database: The Database of Faces from AT&T Laboratories, Cambridge, The Japanese Female Facial Expression Database, The CalTech Database, The PIE Database, The Psychological Image Collection at Stirling. BEV1 Dataset and Caltech Database were used for creating negative samples.

The Harvard Face Database is another database used, which is not used to create feature databases to train classifiers. It is used to test the generalization performance of different classifiers.

Each face database has different image sizes and different background against the faces. Faces and negative samples were manually extracted from these databases and adjusted to the size of 84×96 in order to coincide with the Harvard face dataset. For the experiments, 350 face images were randomly selected for training from above datasets and 150 were chosen for test. 350 non-face images were chosen for training and 150 for test. The testing samples have the same distribution of source databases, which means if 10% training samples are from database A, then 10% testing samples are from database A as well. For details of databases used, please see Appendix A.

4.3 Constructed Databases

To train this dual-layer system requires several separate training datasets for each layer. In the first layer, four feature classifiers need to be trained. There are three databases, each respectively for eyes, mouth, and nose. They were created manually, as described in Section 4.3.1. In the second layer, there is a feature map classifier. Its training set is generated by the trained first layer. It is mainly created automatically, but needs certain manual control and adjustment. There are no feature training sets directly available. They are derived from the face databases mentioned in Section A.1.

The training set for the second layer is determined by the first layer. An ideally trained first layer can precisely locate features and reject non-features in a face image. So when creating the training set for the second layer, the ideal first layer can create correct feature map for faces and non-faces. A well trained second layer can also help to improve the performance of the first layer in return by giving the correct feature locations. But in fact, considering the complexity and enormous diversity of the non-face background, it is difficult to create a perfect first layer.

Training a dual-layer system is a process of improving the training database gradually to reach a training limit. Table 3.1 in Chapter 3 has shown the procedure of generating the database and verifying the training result for this system.

4.3.1 Creating Feature Database

The initial training dataset has 350 faces and 350 non-face examples. I extracted face feature training datasets from these faces and non-faces. The feature sizes are decided by their geometry characteristics. For the eye, each eyebrow and eyeball in all images should be contained entirely within the feature area and the pupil should be at the centre of the feature image patch. 25×29 is the size which contains the biggest eyebrow and eyeball. The size of nose, 38×22, is decided by the nose width and the distance from bridge of a nose to the very bottom of nose. The width and the height of mouth decide the size, 33×20, of mouth samples.

Positive feature examples in the initial training sets had to be extracted manually image by image. The negative feature examples are extracted from non-face images randomly. Due to the large number of potential non-feature examples, the creation of negative feature training sets is more difficult. Table 3.1 describes the basic procedure of creating feature and feature map training sets.

When creating initial negative datasets, I randomly extracted non-feature examples from non-face images. I also extracted some non-feature examples from face images. At the same time, one kind of feature samples can act as another feature's negative samples, for example, noses and mouths as negative examples of eyes. Each feature classifier has its own negative training set. This is the step

1 in Table 3.1. Before the feature classifiers are trained using these sets, the grid search method is used to choose kernels and optimize kernel parameters.

The most important step is step 3. Those trained feature classifiers are used to search across face and non-face images in the training face database. The training face database is randomly split into 5 independent subsets. One is held out for test. Feature classifiers scan over those remaining four subsets and generate feature maps. Features used to generate the feature map could be the first detected during the scan or the one with the highest probability to be a feature. In the experiment of this thesis, I chose the first detected features to form the feature maps. The generated feature maps are used to train a feature map classifier in the second layer. This classifier's kernel and kernel parameters are chosen by grid search method. Connecting those feature classifiers and this feature map classifier together, a dual-layer system is constructed. Then the held out subset is presented to the system and classified. This system might mis-classify some samples in the held out subset and all these misclassified images are checked. Those non-detected features will be extracted to positive feature training sets, and false positive detections will be taken to negative feature training sets. This process will be repeated 5 times until all subsets are tested. At the same time, the average classification correct rate, $R_{correct}$, of all tests is recorded. In step 4, the updated training sets are used to retrain the feature classifiers. Again, grid search is used to optimize the classifiers' configuration.

Step 3 and step 4 will be iterated several times until $R_{correct}$ reaches maximum. Normally $R_{correct}$ will have a maximum. The configurations of classifiers in this iteration will be chosen as the system configuration. The training samples will also be fixed. This maximum may not be the global maximum but as long as the $R_{correct}$ is good enough for the experiment, it will be taken. How to find the global maximum is not a topic of this thesis. Now the system is converged and all individual classifier training sets are formed. Final version feature classifiers are trained using these datasets. The final feature map training set is generated by scanning feature classifiers on all examples in the training face database using the final version feature classifiers. By training on this final feature map training

Feature Name	Number of Images		Image size
	Positive	Negative	
Eye	700	700	25×29
Nose	350	372	38×22
Mouth	350	372	33×20

Table 4.1: *Feature Datasets details*

set, the final version feature map classifier is created.

Table 4.1 shows the details of final version feature training sets.

Figure 4.4 shows some positive feature examples.

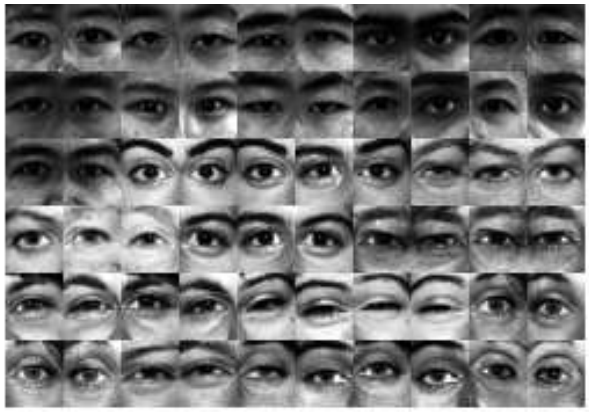
The generation of negative samples is the core of creating good feature classifier training sets. The challenge is how to effectively find those key negative examples, which are close to positive images and can be support vectors. The method proposed above can effectively solve this problem. One of the benefits of using the whole system to create and optimize feature training sets is that the user does not have to manually create a feature map training set. This eases the creation of a feature map training set. This method can offset the imperfection of feature classifier training sets. It counteracts the influence of imperfect feature training sets by decomposing feature training errors into feature map training sets. This makes the dual-layer system tolerate certain misclassification in the first layer so that the correct classification rate can be achieved overall.

Figures 4.5 displays some samples in the negative feature training sets.

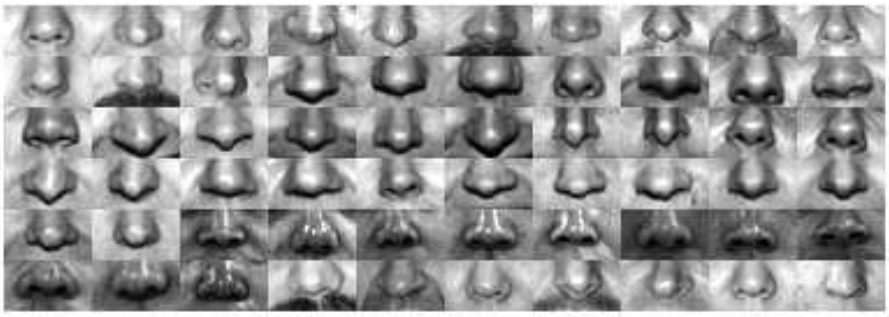
4.3.2 Training the Feature Map Classifier

In the second layer of the dual-layer system, more interest is paid to the structure of the face. There is a feature map classifier, which is trained on an automatically generated training set to learn the face structure.

I considered two ways to create the training set for the feature map classifier. One is to manually create feature maps by giving random locations to each feature. In order to create a positive dataset, different tolerances are given to



(A) Examples of eye



(B) Examples of nose

(C) Examples of mouth

Figure 4.4: *Some positive examples in feature datasets. These samples are from different source databases. Their illumination are various.*

(A) Negative eyes

(B) Negative noses

(C) Negative mouths

Figure 4.5: *Some negative examples in feature datasets. There are some patches containing features but the centres of these features are too far away from the patch centres. Also a feature can act as another feature's negative sample.*

each feature which allows each feature to change its location in a certain area. The feature map is treated as a negative sample if these features are outside of their positive areas. It is convenient to control the composition and distribution of samples in the training set by using this method. But it puts high demand on the training of feature classifiers. These feature classifiers must be trained to find features precisely. Otherwise, some false positive detections may be produced. If these false positive detections locate where true features should be, the feature map will be wrong and classification will fail.

The problem with the manual training set creating method is that it does not cooperate with the training status of the first layer very well. Instead of using a manually created training set, both positive and negative samples in feature map training sets were automatically generated by the trained first layer. This technique has been introduced in 4.3.1.

4.4 Experiments

4.4.1 Single-Layer SVM Classifier

The structure of a single-layer SVM classifier is relatively simple. It directly takes images as inputs, and outputs the classification results. The single-layer SVM classifier is widely used due to the simple structure and ease of training. I trained the SVM to perform classification of faces as follows.

Method. 350 face images and 350 non-face images in the training face dataset were used to train the single-layer SVM classifier. The remaining 150 face and 150 non-face images were used in testing. I compared 4 types of kernels' performance using grid search method. A linear kernel had the highest correct rate, 98.33%, converged over 5-fold cross validation. The linear kernel was chosen for the single-layer SVM classifier. The grid search method helped to find the best value of the kernel parameters. The Linear kernel has the best performance when cost is 3.

In the grid search, the RBF kernel reached its highest correct rate 97.50% when Cost=8 and γ=0.000124; the linear kernel reached the highest correct rate

1. Use grid search method to search proper kernel and optimize kernel parameters on training dataset.
2. Use selected kernel and its optimized parameters to train classifier.
3. Use trained classifier to classify test dataset.

Table 4.2: *A procedure to train and test single-layer SVM classifier.*

Set Name	Classification result			Correct(%)
	Correct	Incorrect	Total	
Faces	149	1	150	99.33
Non faces	150	0	150	100
Total	299	1	300	99.67

Table 4.3: *Results of experiment using single-layer SVM classifier.*

98.33% when Cost=3; the polynomial kernel reached its highest correct rate 97.50%, when Cost=2 and γ=7; the sigmoid kernel only reached the highest correct rate 50%. For all kernels, the search range of Cost is between 0 and 10 which was decided after a coarse search from 2^{-5} to 2^{15} where 2^3 gave the highest correct rate, search step is 1; the search scope of γ is between $1/k$ and 10 which was decided after a coarse search from 2^{-15} to 2^5 where 2^3 gave the highest correct rate. k is the number of attributes in the input data. γ has different meaning for different kernel. See Chapter 2.

Once the grid search method found the proper kernel and its optimized parameter configuration, a SVM classifier was trained on the training face dataset using the optimized kernel. The trained classifier was tested on the test face dataset. Table 4.2 shows the procedure of testing a single-layer SVM classifier.

Results. The single-layer SVM classifier displayed excellent performance. For the 150 test faces, 149 were successfully detected and only 1 was missed. The success rate was 99.33%. For the 150 non-faces, 150 were successfully detected as non-faces and 0 were misclassified as faces. The success rate was 100%. The total correct rate for the 300 test images is 99.67%. See Table 4.3.

Figure 4.6 is the only wrongly detected face. Compared with other faces, this

face has bigger attitude extremity and relatively different feature configuration. Its eyes are located lower and hair occupies more area than other face.

Figure 4.6: *This face is recognised as a non-face by the single-layer SVM classifier.*

Discussion. The single-layer SVM classifier displays a good performance on detecting faces. On the one hand, the high correct rate is due to the high performance of *Support Vector Machines*. On the other hand this might be because of the non-face examples. The negative examples might be relatively easy to be discriminated from faces. The SVM classifier can easily learn the separation boundary. The performance can be further tested using more difficult examples in future.

4.4.2 *Cengji* without Attention Mechanism

Cengji works in a different way compared with the single-layer SVM classifier. There are two layers each for detecting features and judging feature maps. In this experiment, I test the system without attention mechanism on the face dataset.

Methodology

The procedure of creating training datasets for the dual-layer system has been introduced in Section 4.3.1. For the dual-layer system, feature kernel types and their kernel parameters have been decided when creating training sets.

When detecting a feature, a grid with the same size as the training feature images scans the whole input image and outputs image patches to the classifier. Once the classifier finds a feature, it gives the feature a label unique to it and records its location. The grid moves according to certain strategies which are task specific. For example, the grid scans from right to left and top to bottom for the right eye, etc. See Figure 4.2.

The emphasis of *Cengji*'s training is put on the first layer. I tested performance on the first layer then I did the joint test of the whole system.

Performance of the First Layer on Feature Detection

The first layer detects independent features from faces. There are 3 classifiers to be trained separately. The performance of the first layer is decided by individual feature classifiers.

Method The performance of an individual classifier is determined by the training dataset and training method. The creation of the training set is the most important procedure of creating a classifier. This procedure has been introduced in 4.3.

When a classifier finds a feature, it gives the feature a label. The grid moves according to rules, as in Figure 4.2 and previous description. When a search finds a feature or finishes, other begin searches for the remaining features in turn.

150 face images and 150 non-face images were presented to the first layer for the test. For each of these images, 4 feature classifiers scanned it in turn. In the experiments, I used the first detected features to build feature maps.

Results. I used a 5-fold grid search introduced in Section 2.4.4 of Chapter 2 to find the best kernels and optimize their parameters. The RBF kernel performed the best for each feature classifier. During 5-fold cross validation grid search, the highest average accuracy on the eye dataset was 96.77% when cost C was 11 and γ was 0.001379, nose dataset 95.57% when cost C was 6 and γ was 0.001196, and mouth dataset 87.26% when cost C was 10 and γ was 0.001515. These kernel parameters were then used to train the feature classifiers.

For the test face images, the feature classifiers work well but there were false

positive detections in some faces. For non-faces, the second layer can reject most of these, but there were also some false positive detections existing in the first layer. Figure 4.7 is a sample of detecting different features from the face. Note that each feature may be recognized several times. The bottom right image is a correctly classified face in which only the first detected features were taken to form the face feature map.

Discussion. The first layer of this system plays an important role for the dual-layer SVM system. The first layer feature detectors can detect most features in images but there were false positive feature detections existing. The false positive detections increase when the light and attitude extremity are increased. This situation can be improved by improving the datasets, such as adding more key negative samples and adjusting the positive samples. (The performance of the feature classifiers can also be improved by adjusting cost C of each class when the imbalance of training example numbers of two classes is big. This method, also known as weighting, adjusts the separation boundary between two classes to optimize the classification [94, 107, 26].)

Joint Performance of Dual-Layer *Cengji*

Method. In this experiment, the first layer and the second layer were joined together to detect faces from images. The first layer is the same as mentioned in 4.4.2. The feature map classifier of the second layer is a SVM classifier as well. The creation of the training set has been introduced in 4.3. The kernel type and kernel parameters of the feature map classifier were selected by 5-fold grid search method.

Figure 4.8 shows the relation between the correct rate and the number of training cycles. There are four iterations. During the second training iteration, $R_{correct}$ reaches maximum. The configuration of the classifiers in this iteration was chosen as the system configuration. The training samples were also fixed.

When this system classifies faces, an input image is presented to the first layer. Four feature classifiers in the first layer scan this input image according to predetermined sequences, right eye→mouth→nose→left eye. When the first

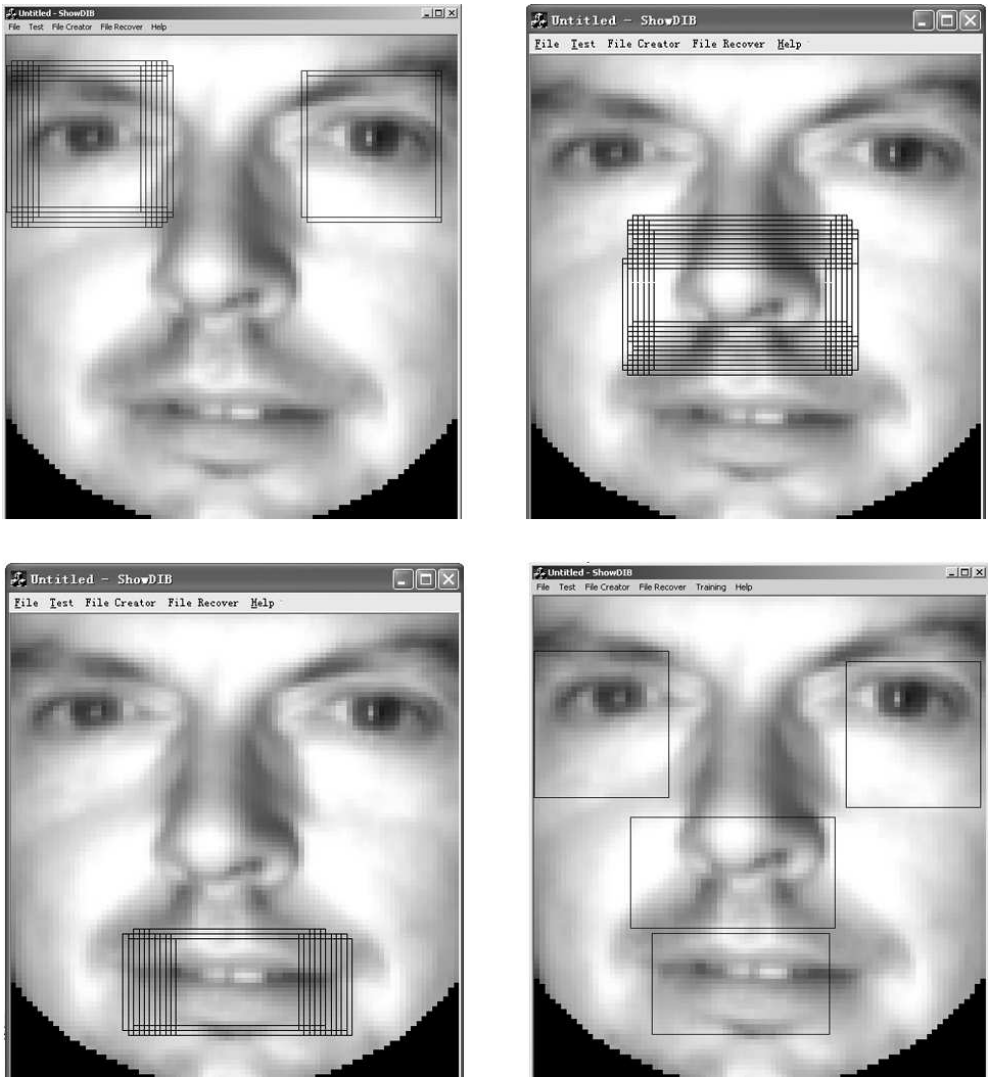


Figure 4.7: *Detection of features. Each feature can have more than one detected samples, only the first one will be passed to* Cengji *to form the final feature map.*

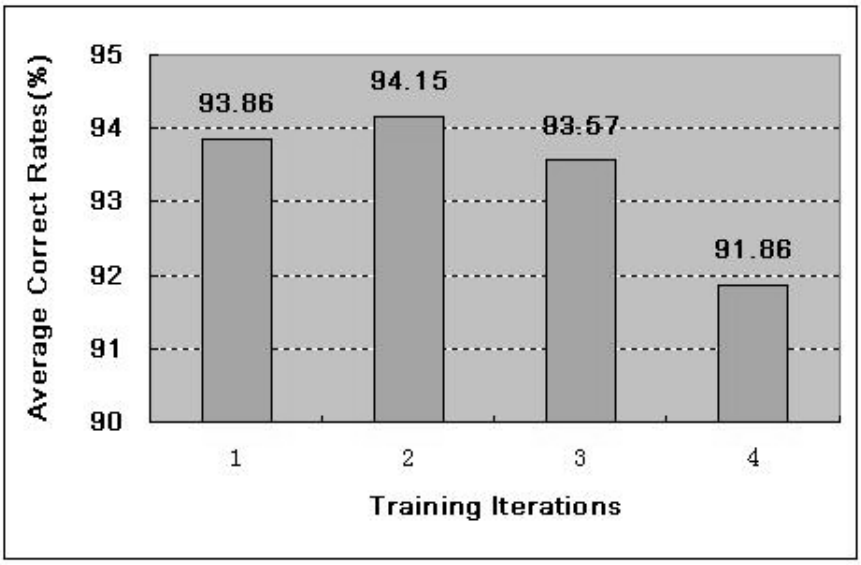


Figure 4.8: *Relation between training iterations and average correct rates. During the second iteration, the average correct rate reaches maximum: %94.15. The configuration of classifiers and training sets were fixed as they were in this iteration.*

classifier finds the first feature it is looking for, it stops the search and reports the feature label and coordinates. Then the next classifier starts to search. This search process will be over once all classifiers complete their search. After the feature search is complete, feature classifier outputs are composed to form a feature map, which is the input to the second layer. The second layer judges whether the feature map fed to it has the structure that a face has. Its output is 1 for face, 2 for non-face.

Results. Feature classifiers have exactly the same configurations as introduced in 4.4.2. Parameters of the feature map classifier kernel were decided by grid search. The linear kernel had better performance than the other kernels. The highest correct rate was 94.17% when cost C was 7. The dual-layer system gave a good performance: 286 images were correctly detected from 300 test images, 150 faces and 150 non-faces. The overall correct rate was 95.33%. See Table 4.4. Figure 4.9 and Figure 4.10 show some samples of successful classifications. Figure 4.11 and Figure 4.12 show some samples of failed classifications.

Set Name	Classification result			Correct(%)
	Correct	Incorrect	Total	
Faces	138	12	150	92
Non faces	148	2	150	98.67
Total	286	14	300	95.33

Table 4.4: *Face classification result of the dual-layer* Cengji.

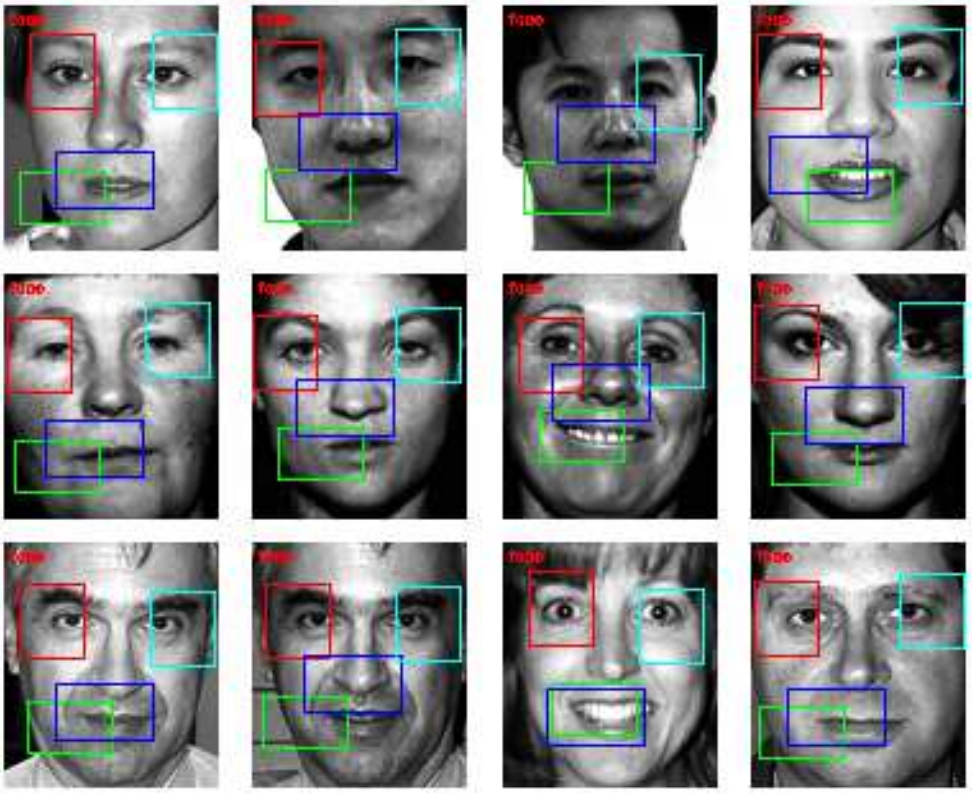


Figure 4.9: *Images correctly classified as faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth.*

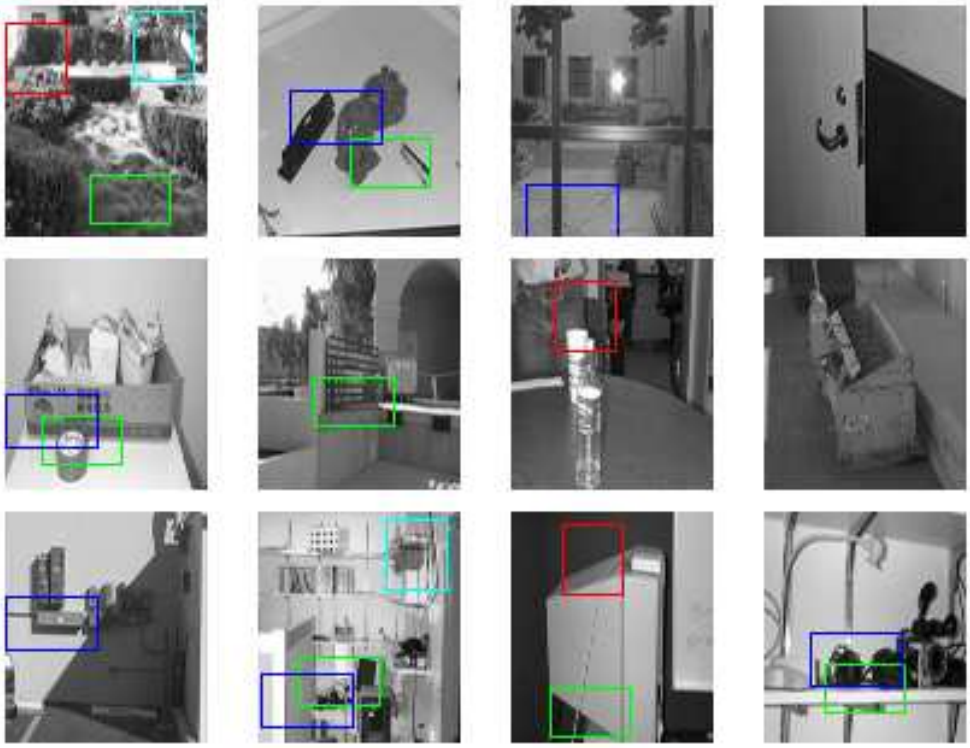


Figure 4.10: *Images correctly classified as non-faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth.*

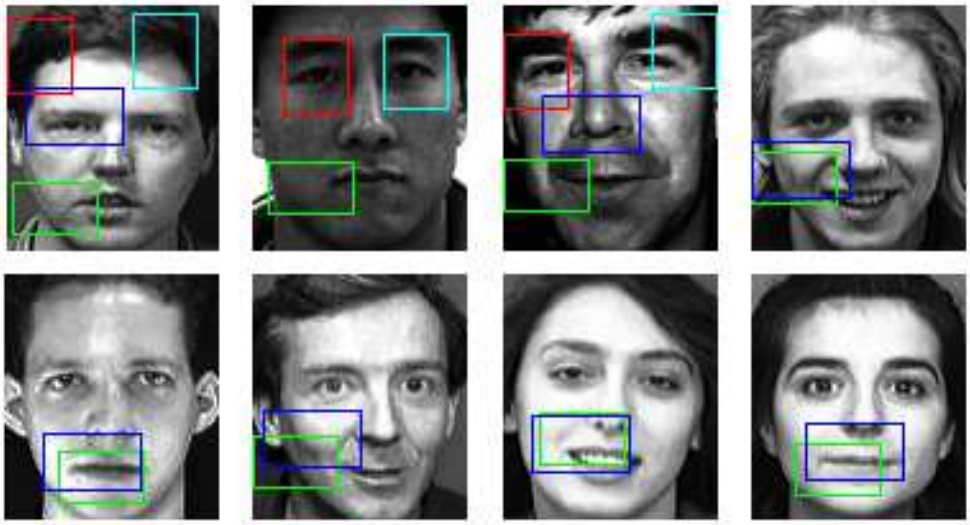


Figure 4.11: *Images incorrectly classified as non-faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth. These images are misclassified because: 1. Features are not detected; 2. Features locate in positions forming a negative feature map.*

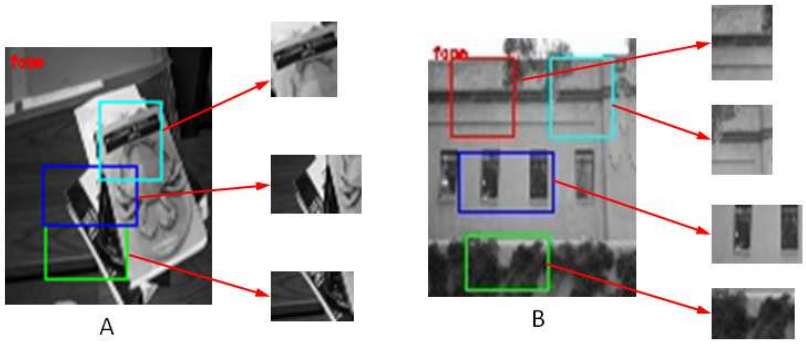


Figure 4.12: *Images incorrectly classified as faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth. The misclassification is caused by those false positive feature detections at the right side of each image. These false positive features form positive feature maps, therefore these non-faces are classified as faces.*

Discussion. There were more than one example of each feature detected in the first layer. Only the first detected samples were taken, and their locations have a decisive influence on the feature maps. See Figure 4.7. The second layer classification depends strongly on the first layer. Comparing the results in Table 4.3 and Table 4.4 suggests that the performance of the dual-layer system is not as good as the single-layer SVM classifier. What makes the dual-layer system have lower correct classification rate than the single-layer SVM classifier is false feature detections, especially the false positive detections, where an eye, for example, is found in an incorrect location. Many non-features are detected as features. When one of these false positive feature detections was at the first position to be detected, the feature map can be incorrect. The occurrence of the false positive feature detections reflects the imperfection of feature classifier training.

Figure 4.13 shows some correctly classified faces. This figure displays the compensation effect of the second layer on the training imperfection of the first layer. In the faces shown in this figure, some features detections are not positioned well, but the feature map classifier learns these imperfections when being trained. It can ignore or tolerate them and give the right classification. The training datasets and procedure have counteracted the mistakes or imperfections generated by individual feature classifiers.

4.4.3 Dual-Layer *Cengji* with Attention Mechanism

Cengji with attention mechanism (for simplicity, I call *Cengji* with attention mechanism *Cengji AM* and *Cengji* without attention mechanism *Cengji WA*) focuses on the expected positions of certain features relative to the target object. This section evaluates the complete system, as described in Chapter 3. This experiment compares the system's performance with and without the attention mechanism, to see the benefits of the attention mechanism.

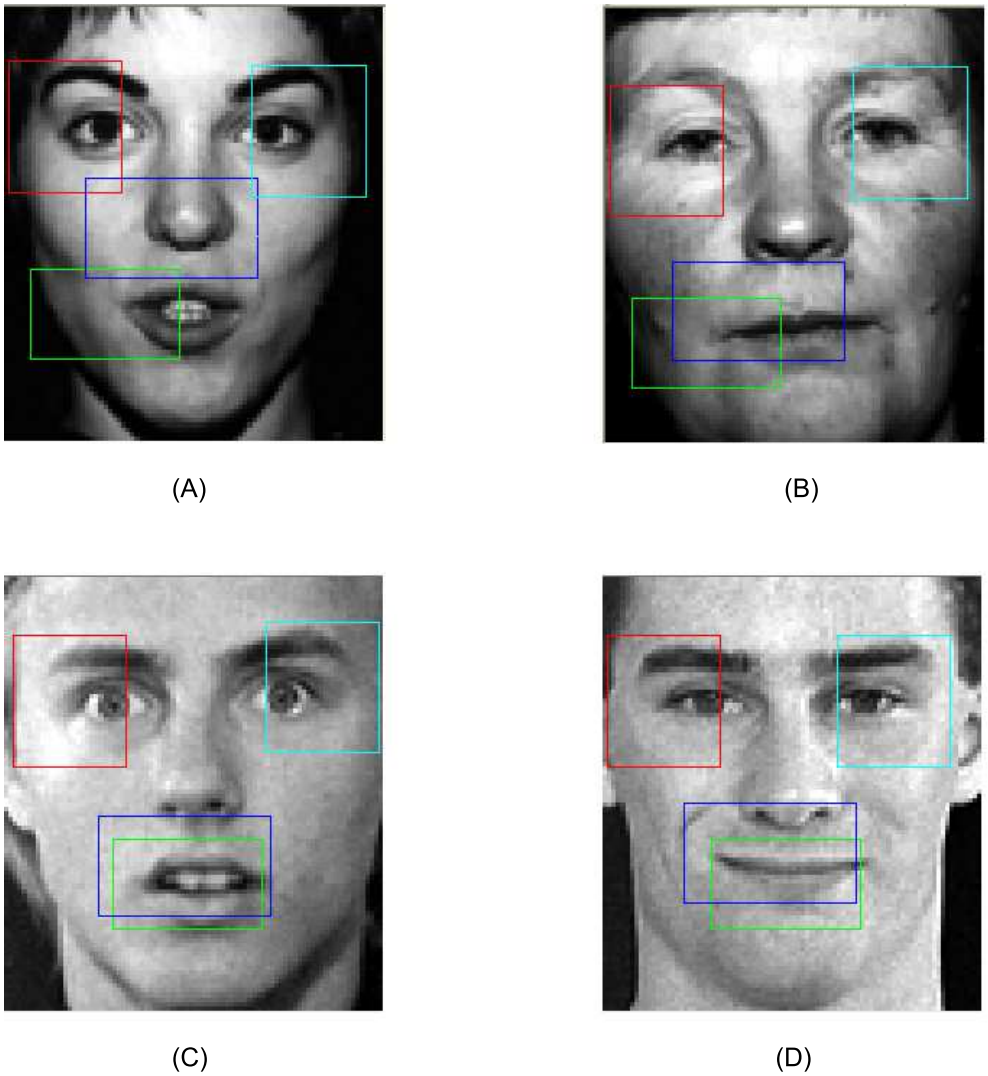


Figure 4.13: *Compensation on imperfect detection of individual features. (A). The mouth is not located well; (B). The mouth is not located well and the nose is at the wrong place; (C). The nose is at the wrong place; (D). The nose is not located well. But all these faces are correctly classified by* Cengji.

Methodology

Cengji AM uses the same first and second layer of the dual-layer *Cengji* introduced in Section 4.4.2. Each classifier uses the same kernel, same parameters, and classification model trained on the same dataset. This means the trained classifiers of *Cengji WA* are directly used for *Cengji AM*. The sole difference is the introduction of the attention mechanism.

Without the attention mechanism, the system searches for features in the input image according to certain routes. This is a simple method when there is not enough knowledge available about the target object. It is straightforward but time consuming. If the performance of an individual feature classifier is not good, this method can bring many false detections. The introduction of the attention mechanism changes this situation. It gives the system clues about the locations of features when relative information is available. This makes the system 'smarter'.

A feed-forward neural network is used as an autoassociator. It is trained using feature map training set by BackPropagation algorithm [77], which is introduced in Chapter 2. When it is trained to be an autoassociator, both the input and output vectors in a training pair are the same feature map. The purpose of training is to reproduce the input fed into this network at the output layer. After training, it learns and memorizes the structure of feature maps of faces. Even when the input is a partial feature map, it can still recover the whole feature map. The challenges of training are to decide the hidden layer size and when to stop training, and at the same time, to prevent over-fitting. I chose the hidden layer size by trying as many as possible. The stopping criterion is the training error.

The training of the autoassociator is relatively independent of the training of those two layers. Due to the limited training samples and in order to speed up the training, I resized feature maps from 84×96 to 21×24. The autoassociator is a 3 layer feed-forward neural network with 1 input layer, 1 output layer and 1 hidden layer. There are 504 nodes in the input and output layers respectively. The hidden layer has 500 nodes which is decided by experiment. I set a very small stop error, 0.00001, then trained the neural network with different hidden

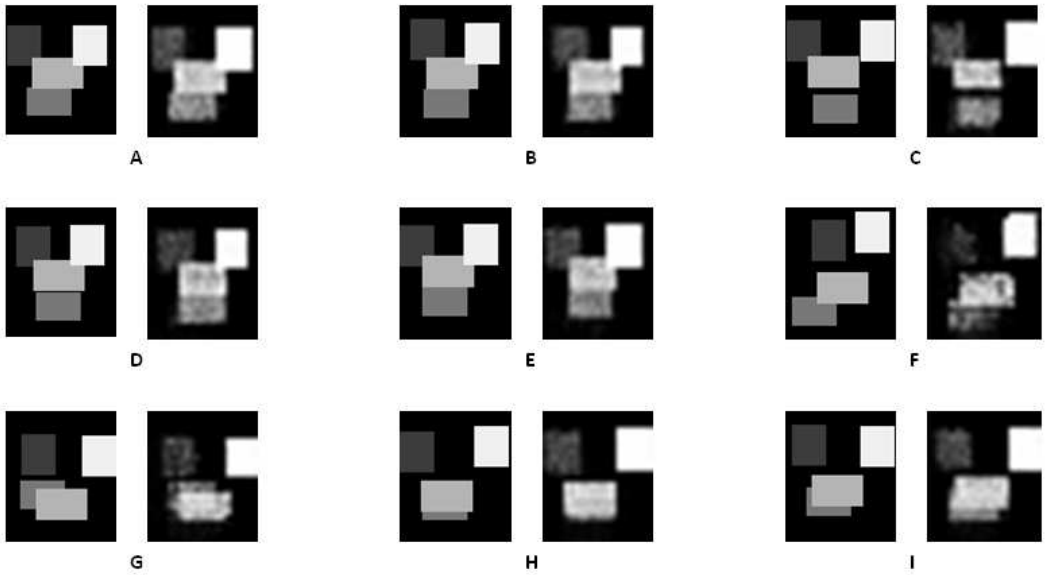


Figure 4.14: *Training test on face feature map associator. The left side of each small image is the input feature map. Right side is the associator output.*

layer numbers. Actually, the training did not reach this stop error. But after certain training period, the training error was stable. 100 node hidden layer configuration gave stable error at around 1.1. 200 gave stable error at around 0.95. 300 is stable at around 0.81. 400 is stable at around 0.32. 450 is stable at around 0.09. 500 node hidden layer gave the lowest error 0.06. The stop criterion is set as the average output error of each sample being smaller than 0.06. 330 face feature maps were used to train the associative feed-forward neural network. Figure 4.14 shows some test samples of the face feature map associator. The inputs are face feature maps generated by the first layer but not used to train the associator. All together 20 face feature maps were used to test the performance of the associator. An average error of 0.08 was achieved.

A trained autoassociative feed-forward neural network can recall a full feature map when a partial feature map is presented. Then this recalled feature map can help the system go to possible places to search for features. When the system goes to the proposed location, it searches within a certain area, which takes the proposed coordinates as its geometrical centre. Figure 4.15 shows some samples

Set Name	Classification result			Correct(%)
	Correct	Incorrect	Total	
Faces	137	13	150	91.33
Non faces	146	4	150	97.33
Total	283	17	300	94.33

Table 4.5: *Face classification result of the dual-layer* Cengji AM.

of applying the attention mechanisms to produce proposed artificial feature maps with partial inputs.

The same test data for *Cengji WA* is used, for comparison.

Results

Table 4.5 shows the results of face classification using *Cengji AM*. This system gave a good performance: 283 images were correctly detected from 300 test images, 150 faces and 150 non-faces. The overall correct rate was 94.33%. The clear benefit from this system is that it can save classification time. For classifying test faces, it saved 25.22% time of a *Cengji WA*. For non-faces, it saved 16.46% time. Classifying faces saved more time because for a non-face, a classifier usually finds no feature after searching the proposed areas.

Table 4.6 is the confusion matrix between *Cengji AM* and *Cengji WA*. It suggests *Cengji AM* gave a slightly inferior performance: 94.33% correct compared with 95.33%. But it is noticed that the performance of *Cengji AM* on detecting positive faces is almost equal to that of *Cengji WA*: 91.33% vs 92%. What makes the performance of *Cengji AM* worse is mainly the recognition of non-faces. This will be discussed later.

Figure 4.16 and Figure 4.17 show some samples of successful classifications. Figure 4.18 and Figure 4.19 show some samples of failed classifications.

Figure 4.20 and Figure 4.21 compare the number of calls to feature classifiers within *Cengji WA* and *Cengji AM*. The calls to eye classifier to detect right eyes are fixed call numbers both for *Cengji AM* and *Cengji WA* because the search for this feature is required by both. Apart from this, the remaining calls are

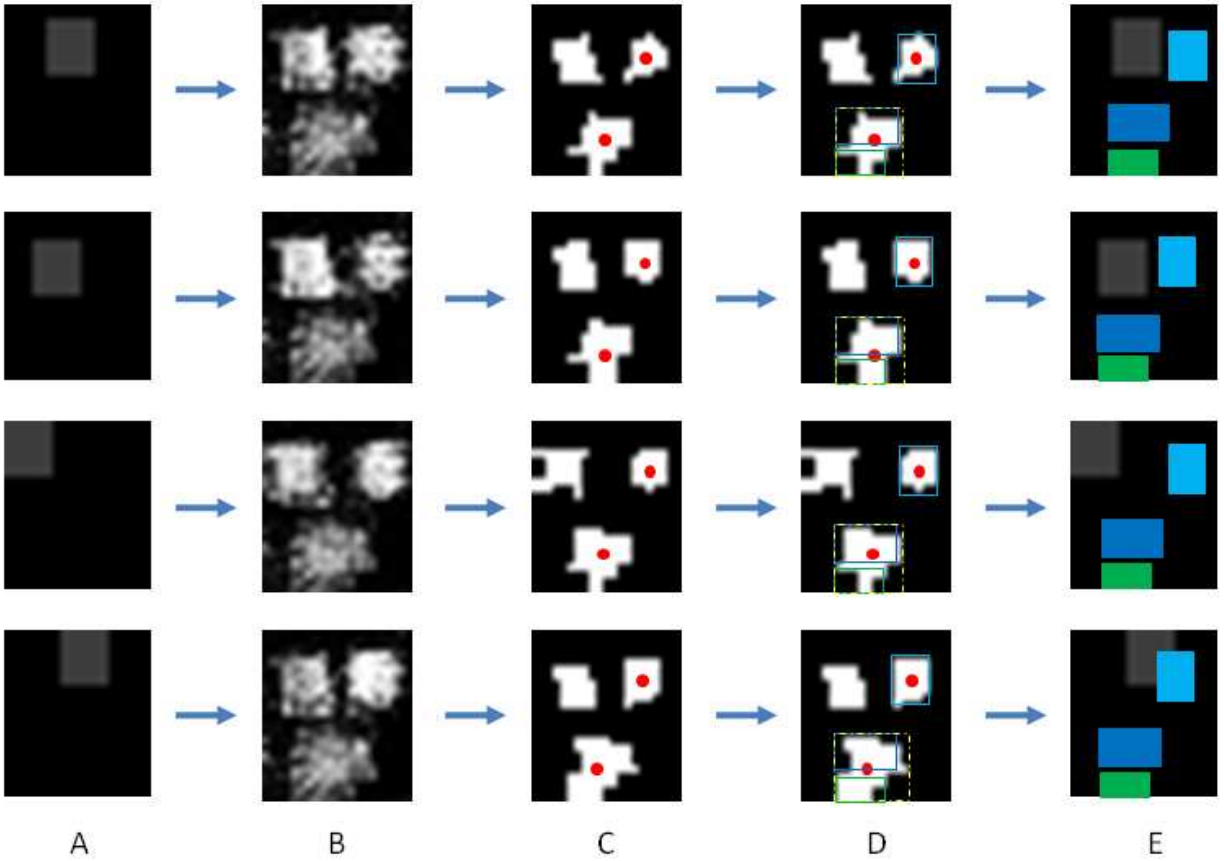


Figure 4.15: *Samples of proposing feature map by applying the attention mechanisms. Different inputs will produce different outputs.*

	With AM		Without AM	
	Face	Non-face	Face	Non-face
Face	137	13	138	12
Non-face	4	146	2	148

Table 4.6: *Confusion matrix of* Cengji*, with and without attention mechanism. The row gives the correct class, and the column gives the class produced by the system.*

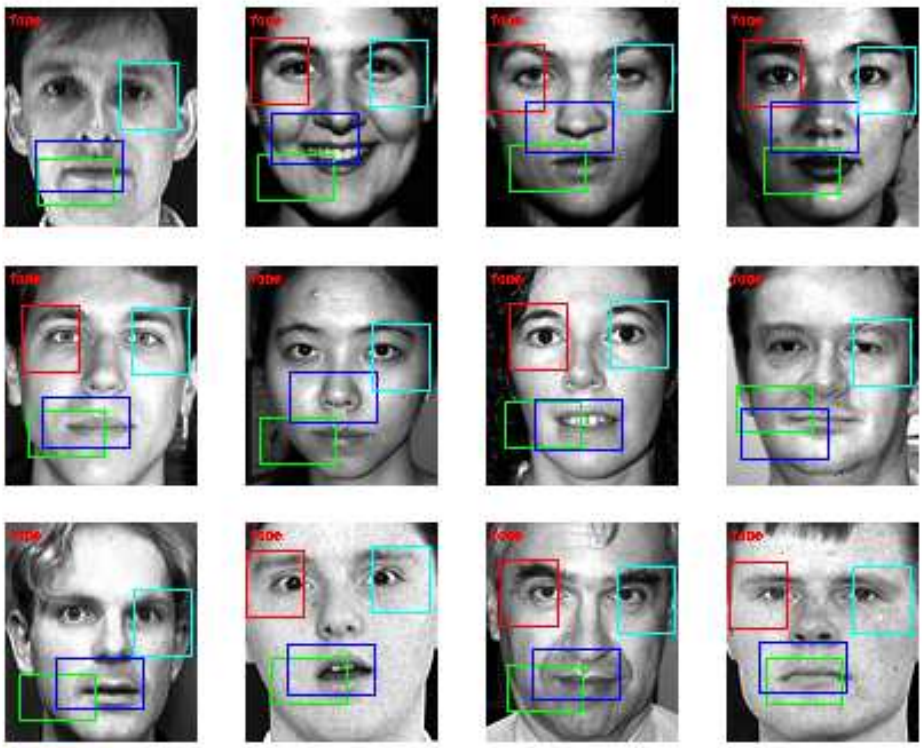


Figure 4.16: *Images correctly classified as faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth.*

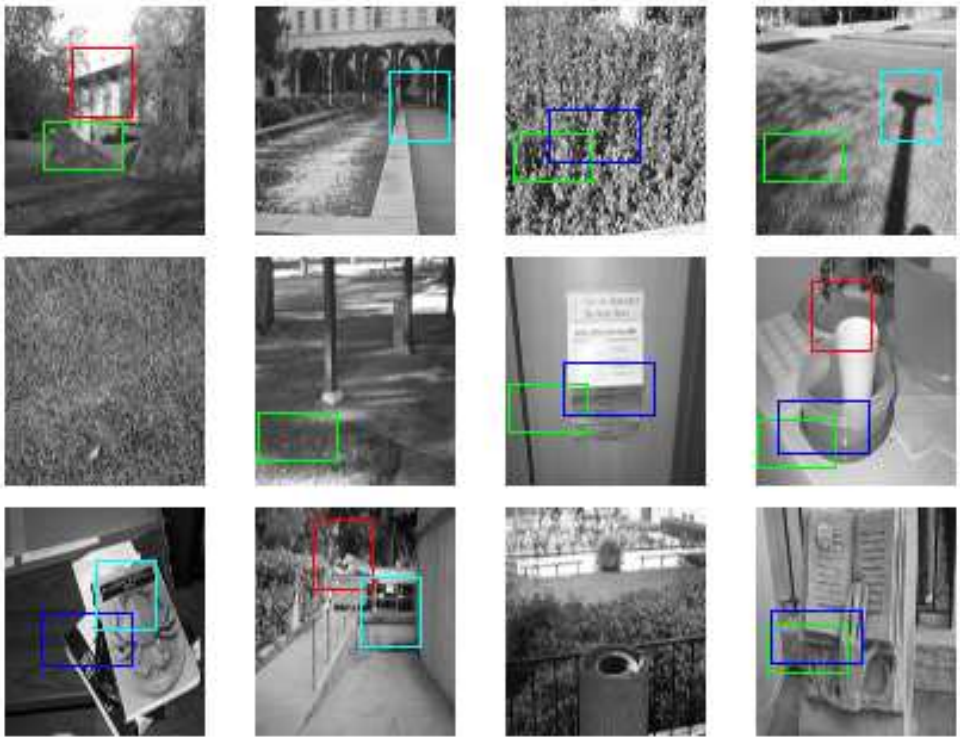


Figure 4.17: *Images correctly classified as non-faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth.*

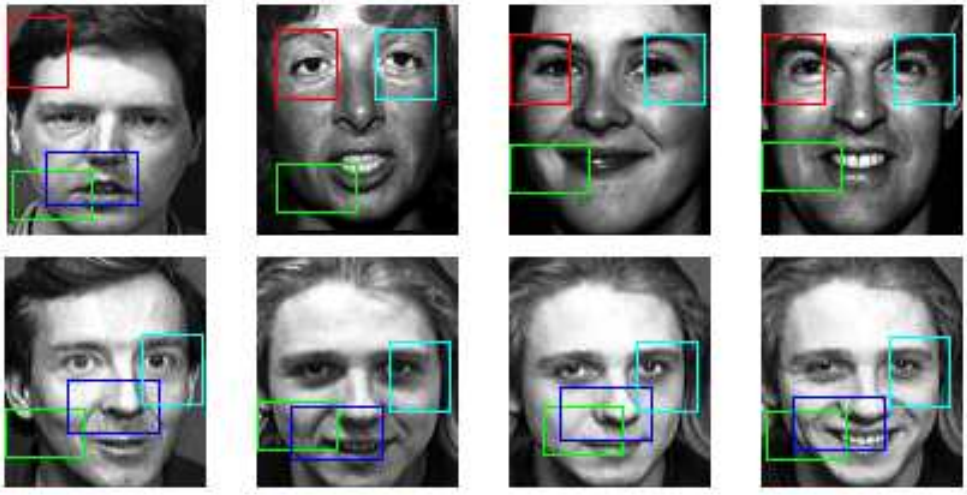


Figure 4.18: *Images incorrectly classified as non-faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth.*

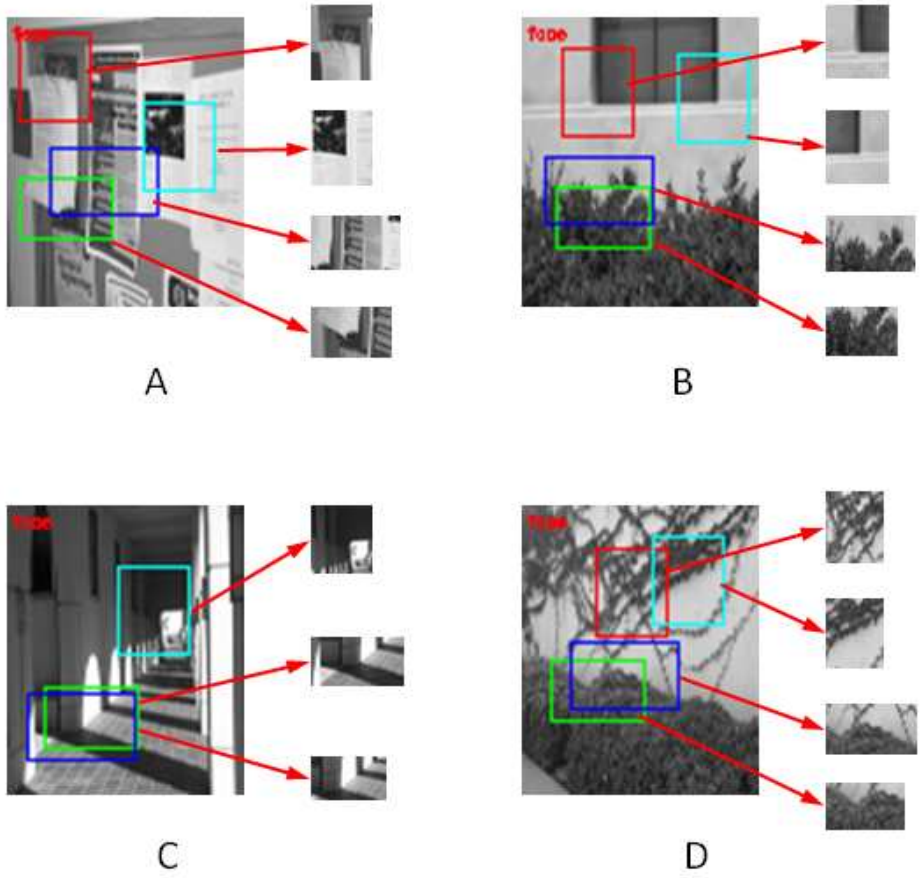


Figure 4.19: *Images incorrectly classified as faces. Red: right eye. Light blue: left eye. Blue: nose. Green: mouth. The misclassification is caused by those false positive feature detections at the right side of each image. These false positive features form positive feature maps, therefore these non-faces are classified as faces.*

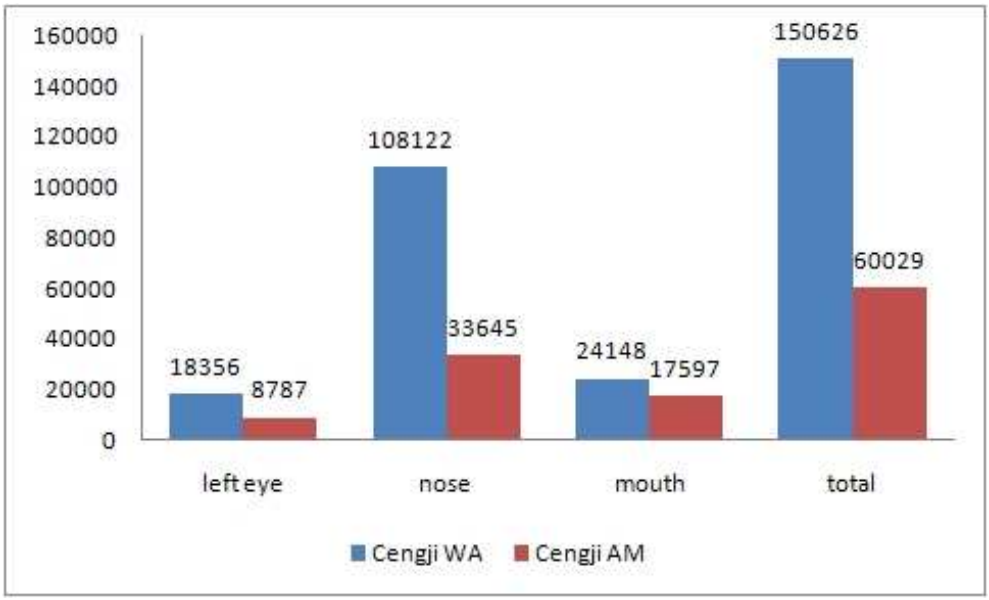


Figure 4.20: *Calls to SVM feature classifiers by* Cengji *with and without attention mechanism on test faces. This table shows calls on SVM feature classifiers when applying* Cengji *on 150 test faces. Blue bars are of* Cengji WA *and red bars are of* Cengji AM*. The search of the right eye is not influenced by the attention mechanism. It is not included in this figure. 1. Calls to eye classifier to detect left eyes, 18356: 8787 times; 2. calls to nose classifier, 108122: 33645 times; 3. calls to mouth classifier, 24148: 17597 times; 4. total calls, 150626: 60029 times. Attention mechanisms save up to 60.15% SVM calls.*

influenced by the attention mechanism. These figures show that the application of the attention mechanisms saves SVM calls up to 59.11%, 6678 with attention mechanism compared with 16332 without it.

Discussion

An attention mechanism is the cognitive process of selectively concentrating on one aspect of the environment while ignoring other things [cited from en.wikipedia.org]. This mechanism allows a human to concentrate on certain tasks without being disturbed by useless information. This process also plays a role in machine learning systems, to avoid confusion from irrelevant features [14].

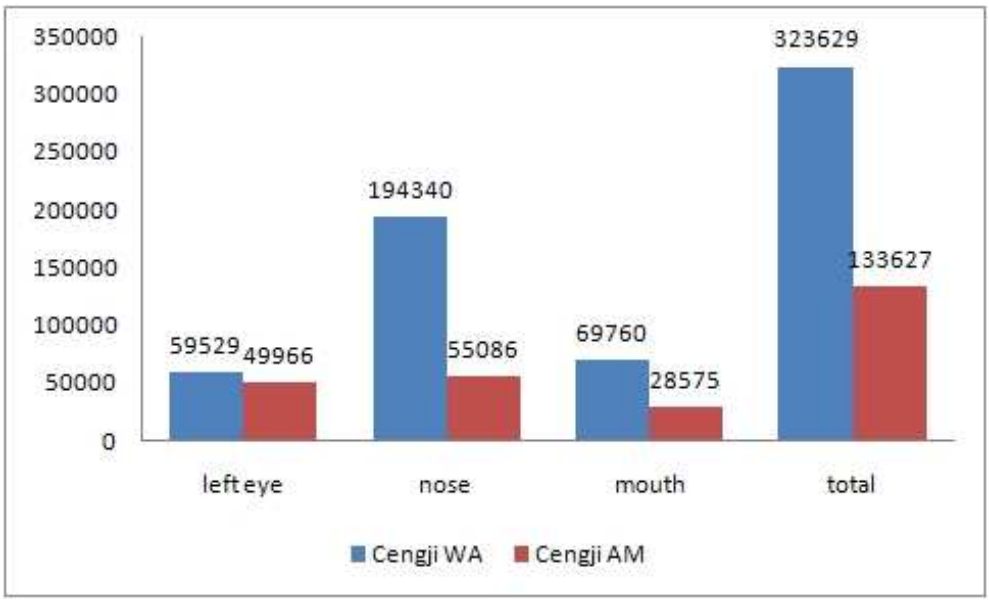


Figure 4.21: *Calls to SVM feature classifiers by* Cengji *with and without attention mechanism on test non-faces. This table shows calls on SVM feature classifiers when applying* Cengji *on 150 test non-faces. Blue bars are for* Cengji WA *and red bars are for* Cengji AM. *The search of the right eye is not influenced by the attention mechanism. It is not included in this figure. 1. Calls to eye classifier to detect left eyes, 59529: 49966 times; 2. calls to nose classifier, 194340: 55086 times; 3. calls to mouth classifier, 69760: 28575 times; 4. total calls, 323629: 133627 times. Attention mechanisms save up to 58.71% SVM calls.*

The results here indicate that the attention mechanism can help the system save classification time up to 20.84% on average; 25.22% and 16.46% for faces and non-faces respectively. Especially the calls to relative SVM classifiers have been saved up to 59.43% on average; 60.15% and 58.71% for faces and non-faces respectively. The time and classifier call saving are achieved by leading the system to the most likely areas containing features. Hence, the searched areas are reduced. Figure 4.22 shows the reduced search areas. Another effect of applying the attention mechanism is to help the system locate features with higher precision. In other words, the attention mechanism enables *Cengji AM* to avoid misclassifying certain face features. Figure 4.23 gives an example of the face correctly classified with the attention mechanism.

The attention mechanism has two influences on the performance of this system. On one hand, the attention mechanism reduces the detection of, for example, non-noses as noses. This means that the feature map is more likely to have features within the correct locations, which the second layer can correctly identify as a face. When the feature classifiers are trained well, they are more likely to identify non-features in a non-face image within areas suggested by the attention mechanism, where a positive feature might be. But on the other hand, if there is an imperfection when training the feature classifiers, the attention mechanism can cause this system to make some mistakes. When a feature classifier is led to the possible area suggested by the attention mechanism, if it is not well trained, it may take a non-feature as a feature. This false positive feature is in the position that can form a positive feature map. This increases the chance that a non-face image is taken as a face. The performance of this system is decided by the interaction of these two kinds of influences introduced by the attention mechanism. The better the performance of the feature classifiers is, the higher the performance of *Cengji* will be.

4.4.4 Performance Test Using Different Database

In the previous experiments, both training datasets and test datasets were from the same source database. So samples in the test datasets have some common

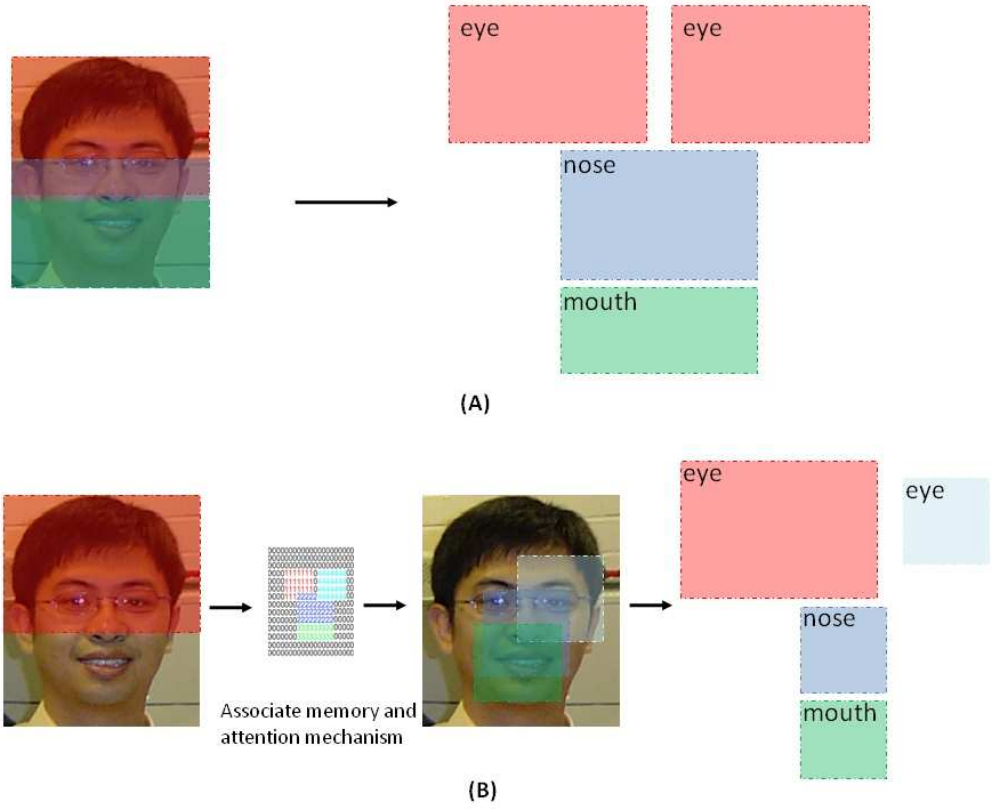


Figure 4.22: *Reduced search area. (A) shows the search areas of a* Cengji WA*. This system searches every feature in broad areas. (B) shows the effect of the attention mechanism. Once the first feature is detected, the associative memory generates an expected feature map, which gives the likely area for each feature. This reduces the search area, hence saves search time by up to 20.84% on average and reduces calls to feature classifiers up to 59.11% where the attention mechanism is applied.*

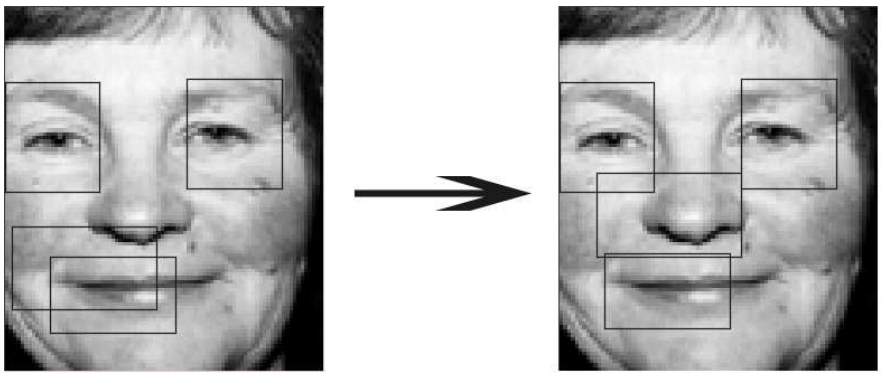


Figure 4.23: *Comparison. Left is a result produced by* Cengji WA. *The nose location is wrong. Right,* Cengji AM *correctly located the nose and detected it as a face.*

Figure 4.24: *Some samples in Set 5 of Harvard face database. They have more illumination extremity and some features (eyes, noses, mouths) are even dark, virtually occluded.*

ground as samples in the training datasets. In order to compare the generality of different algorithms, especially the single-layer classifier and the dual-layer classifiers, I used two untouched databases, the Harvard face database and a negative sample database provided by DIP Lab at Cranfield University.

There are 5 datasets in the Harvard face database. Set 1 has the smallest illumination extremity. And the extremity increases from set 2 to set 5. In set 5, some features are even dark. Figure 4.24 shows some samples in Set 5. This kind of images is not in the training sets used for the previous experiments. Table 4.7 shows the experiment results.

Set Name	Correct Rate (%)		
	Single-Layer SVM classifier	*Cengji WA*	*Cengji AM*
Set1	70	100	100
Set2	52.22	81.11	80.56
Set3	20	56.92	55.27
Set4	8.24	58.24	53.66
Set5	1	47.69	46.83

Table 4.7: *Experiments on Harvard database. The face attitude and illumination extremity increase from set 1 to set 5.*

Classifier	Classification result			Correct(%)
	Correct	Incorrect	Total	
Cengji WA	141	13	154	91.6
Cengji AM	143	11	154	92.9
Single SVM	139	15	154	90.3

Table 4.8: *Classification results on Cranfield non-face database.*

There are 154 non-face images in the Cranfield University negative sample database. They are outdoor scenes, cars, indoor scenes and persons. Figure 4.25 shows some samples in this dataset. This kind of images are not in the training sets used for previous experiments. Table 4.8 shows the experiment results. Interestingly, *Cengji AM* has the best performance while the single SVM has the worst performance on this non-face database.

In this experiment, the dual-layer *Cengji AM* and the *Cengji WA* clearly gave better performance than the single-layer SVM classifier on a different database, both on faces and non-faces. Especially on the most difficult set 5 of the Harvard dataset, the single-layer only got a correct rate of 1%. It is much lower than *Cengji*. Even for the easier sets, *Cengji* outperformed the single-layer classifier. What caused this is mainly the difference between the two different databases. In different training databases, the face structure may have some differences even when the image sizes are the same. For example, the distance between two eyes may be longer than in another database. The single-layer SVM only learns

Figure 4.25: *Some samples in non-face database provided by DIP Lab at Cranfield University. They are from difference sources.*

one kind of eye distance within one training database, so it can not cope with the variety of distance in a different training database. Another factor is the illumination. In the training sets used for previous experiments, faces do not have illumination extremities like faces in Harvard set 5. Therefore there is no chance for the single-layer classifier to learn the dark or even occluded faces. But for the dual-layer system, its first layer detects features with the feature classifiers then forms a feature map, which is the input to the second layer. During training, the second layer classifier will learn the configurations of feature maps. Because only the first detected features are taken to form the feature map, there is the possibility the detected locations of features may vary from the locations of features in the images. Therefore the learned feature maps may cover the feature configurations within another database.

From this experiment, *Cengji* displayed better generalisation performance than the single-layer classifier. It has the ability to cope with the variance of target illumination, appearance and occlusion. This superiority is an important reason to build a hierarchical and trainable object recognition system.

4.5 Conclusion

In this chapter, I did face classification experiments using the single-layer SVM classifier, dual-layer *Cengji AM*, and dual-layer *Cengji WA*. Then I compared their generality performance using the Harvard face database and a negative image database provided by Cranfield University.

In the experiments, the single-layer SVM classifier displayed good performance. It got a high classification rate of 99.67%. This has been described in 4.4.1. The high performance of the single-layer SVM classifier is why it is appropriate for individual classifiers in the dual-layer system.

The experimental results demonstrate that the developed system achieves a good result in face classification. The feature map concept has been shown to work. The dual-layer system with and without attention mechanism displayed good performance. For 300 test images, the correct rate of *Cengji WA* and *Cengji*

AM are 95.33% and 94.33% respectively. This is the accuracy of discriminating face from non-face images in a held-out test dataset of 300 images. Further, the results show that the attention mechanism saves the total time to locate features by an average of 21%. The calls to relative SVM classifiers have been saved up to 60% on average. Also the accuracy of located features is improved. I had expected a greater increase in time, because the scanning algorithm requires many comparisons before it reaches the point where a feature may occur. I am currently investigating alternative local search routines to try to improve the search time. Although potentially the training procedure described in Table 3.1 in Chapter 3 could involve a large number of iterations, the system achieved its best performance on the training set after just two cycles, so the overhead is small. However, the improvement in performance was not so large, indicating that perhaps the initial training sets for the individual feature classifiers were already comprehensive enough.

The attention mechanism makes this system search intelligently; that is to say it can directly search in areas with a higher likelihood of containing features and ignore irrelevant information. This is smarter than exhaustively searching everywhere without discrimination when prior knowledge about target objects is available.

The experiments showed the positive effect of the attention mechanism that it can reduce the classifier calls by leading this system to areas that may contain features. It can also bring a negative influence into the system when the individual feature classifiers perform poorly. This influence can be offset to a certain extent by the joint training dataset creation method introduced in 4.3. The whole performance of the system is a trade-off between the positive and the negative effects. The experiments suggest that the hierarchical structure of *Cengji* with and without attention mechanism is successful.

Another notable priority of the hierarchical system is its better generality. The dual-layer system performs better than the single-layer classifier on different databases. It displayed better invariance to the illumination, appearance and occlusion variance. This is an importance characteristic to find targets, which

may be different from its training samples in terms of illumination, appearance and occlusion, from complex real-world backgrounds.

From the experiments, the feasibility of the hierarchical multi-layer system has been verified. The application of an attention mechanism has been successful. It saves feature detection time significantly. Because this system is a machine learning based multi-classifier system, the training of these individual classifiers is decisive to the system. The strategy of training this multi-layer system is effective. Though the single-layer SVM classifier displayed a higher performance on face classification, the target of this thesis is to create an object recognition system for a general and robust scene understanding system. In the next chapter, I extend and further test *Cengji* in a new domain.

Chapter 5

Application to Detecting People from Thermal Imagery

In this chapter, I look at the problem of detecting people from complex backgrounds in infrared images. Infrared imaging has extensive application in both military and civilian domains [15, 5]. Military purposes of infrared imaging application include target and threat detection, night vision, target guidance, surveillance, target tracking, and so on. Civilian applications include remote temperature measurement, thermal efficiency investigation, wireless communication, medical diagnosis assistance, fire rescue, and so on. It also has broad uses in astronomy to investigate the features of planets and other projects from complex space. *Cengji* will process more general scenes in this chapter. The images used were taken from natural environments using a thermal camera. The targets are people in the scenes. Compared with the face recognition experiments in Chapter 4, target objects have more variety and the background in the image is more complex. And it is also inevitable the target objects can be blurry or occluded by other objects. In this kind of situation, a single classifier detecting the whole target will find it difficult to successfully seperate targets from background. The hierarchical system is applied to solve these problems. The benefit of applying an auto-associative memory and attention mechanism will be explored. In addition, I will also test the performance of a single-layer SVM classifier and a Haar like

feature classifier, which allows me to compare standard whole-image classifiers vs my component-based classifier.

5.1 Detecting People from Thermal Images Using *Cengji*

In this experiment, the focus is to detect thermally-imaged people from images. These thermal images of people may be in different appearances compared with the original training samples or some parts may be occluded. I will apply *Cengji AM* to explore the ability of dealing with these complexities and compare the performance with *Cengji WA*. The purposes of this experiment are to explore the generality of this system on different and more difficult inputs and explore the effects of the attention mechanism on the whole system .

Compared with the experiments of face classification, images with more complex scenes will be used in this experiment. There will be a more complex background to be processed. The target to be searched is also more complex than a face especially in terms of pose and illumination. Therefore I used more features to represent a person. These features include head, waist, left upper body, right upper body, left lower body and right lower body.

Figure 5.1 shows the structure of the system which is used to detect people from infrared images, which is similar to that in Figure 4.3.

A difference of the feature search in this experiment is its clustering of detected features. Detected image patches with similar dimensions within a certain area will be clustered together to form a new single patch which will be treated as a single feature. Those clusters with less members than the threshold number, which is set by the user from an experiment, will be discarded. For example, all recorded heads will be clustered to filter out repeated detections and noise. Figure 5.2 shows the clustering procedure. This procedure is intended to reduce false positive feature detections. The size of the cluster is the feature size but its location is the mass centre of those clustered detections.

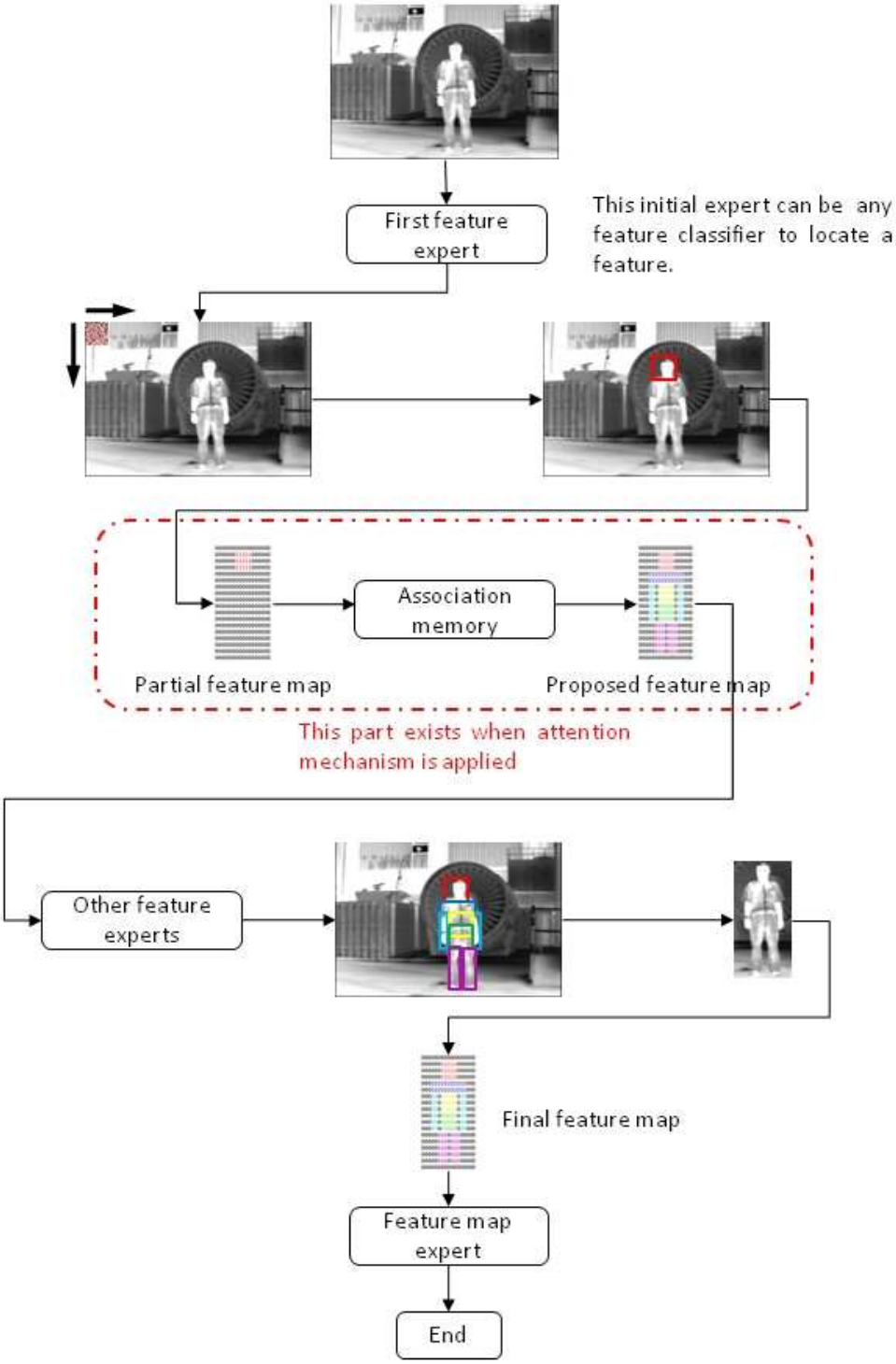


Figure 5.1: *Structure of the dual-layer* Cengji *for the person detection experiment.*

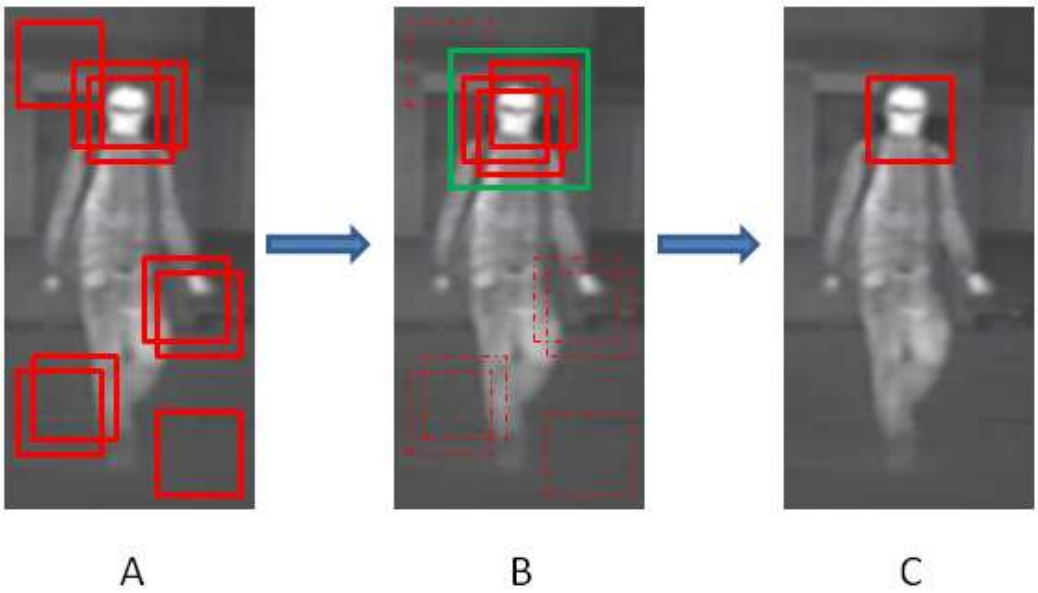


Figure 5.2: *Cluster multi-detections to a final detection. This is a sample of clustering multi-detections of head for illustration only. The clustering threshold is 3 detections within certain area which is decided by experiment. A. There are more than one head detections. Some of them are false positive detections. B. Those detections with less than 3 neighbours will be removed. Detections with no less than 3 neighbours (within the green box) will be clustered together to form a united detection. C. Final detection. Its centre is the average centre of those three detections.*

When the first seed feature, a head feature currently, is found, its coordinate and size will be recorded and the system will extract a sub-image from the input image. The size of this sub-image is decided by the head patch's size. The rule is that the head centre line should be identical with the centre line of the sub-image in width direction and its top should be 1/6 length to the top of the sub-image. After this sub-image is created, the system will search for other features within this sub-image. The second feature to be searched can be chosen randomly. For simplicity in the experiment, I chose waist as the second feature to be searched. The search strategy is same as that of the head.

When the attention mechanisms are not engaged, this system will scan over this sub-image for all remaining features using the same procedure used by the seed feature classifiers. There may be more than one sample (cluster of image patches) of a feature. I took the first one which passes the clustering threshold as the detection of these features to compose the thermally-imaged person feature map. This feature map will be the input to the second layer.

When the attention mechanism is engaged, this system will use the detected head and waist features to compose an expected feature map using the associative memory. This memory will indicate other features' possible locations and dimensions. The system will go to these suggested locations to search for possible features in relatively smaller areas compared with areas without applying the attention mechanism. This procedure will confine the search in areas with a higher likelihood of containing a feature. The search strategy within these areas is the same as that of the seed feature search. Once all feature searches suggested by the proposed feature map are complete, the detected features will be composed to form a final feature map for the second layer classification. Figure 5.3 shows the procedure of generating a proposed body feature map. This proposed body feature map will provide the search areas of the left and right upper bodies, left and right lower bodies. The output from the associative neural network does not discriminate different features clearly but gives the block area containing these features. Therefore in Step D, I find the smallest rectangle containing the image patch and use its coordinates to decide the locations of proposed upper and lower

bodies. Its four corners will decide the search area centres. The search area sizes are determined experimentally to achieve the best balanced performance in terms of search time and accuracy. This figure also shows the different responses from the associator when the inputs are different.

An important issue with machine learning based object recognition systems is the selection of inputs used for detection. We have to answer a question when we develop an object recognition system. That is what kind of inputs distinguish targets from the background best. It is straightforward to take raw pixels as input to a classifier directly. But this kind of input may not be the best choice. Some key information which separates different classes of pixels can be hidden within raw pixels. They have to be discovered or amplified by certain operations.

For this experiment, I mainly tested three types of values for an image patch: raw pixel values, Laplacian values [95], and the combination of these two kinds of values. Other information considered were image moments and histograms [95], but these showed no clear improvement on classification performance. During training, I rescaled raw pixel values and Laplacian values into a range between 0 and 1 (as required by SVMs). The combination of them is simply the sum of two types of values and was also rescaled into between 0 and 1. The Laplacian value is calculated according to Equation 5.1, Where src is the source image. The kernel of applying Laplacian kernel to a digital image is shown in Equation 5.2. The reason for using edge information as an input component is that in a positive sub-image, the target edges form relatively stable and unique contours, the shapes of head, waists, arms and legs. But in a negative image patch, the edges may be distributed in a random shape. Another phenomenon worth noticing is that the edge strength of the person is often stronger than other parts of the image. This is because of the relatively higher temperature of a person compared with the background, though there is a chance of some hot spots existing. In order to find which kind of input was the best choice, I used a grid search method to find the input with the highest classification performance. In the grid search, I tested different kernel types, their parameter configurations, and parameters of input values in certain ranges. The combination of raw pixel value and Laplacian

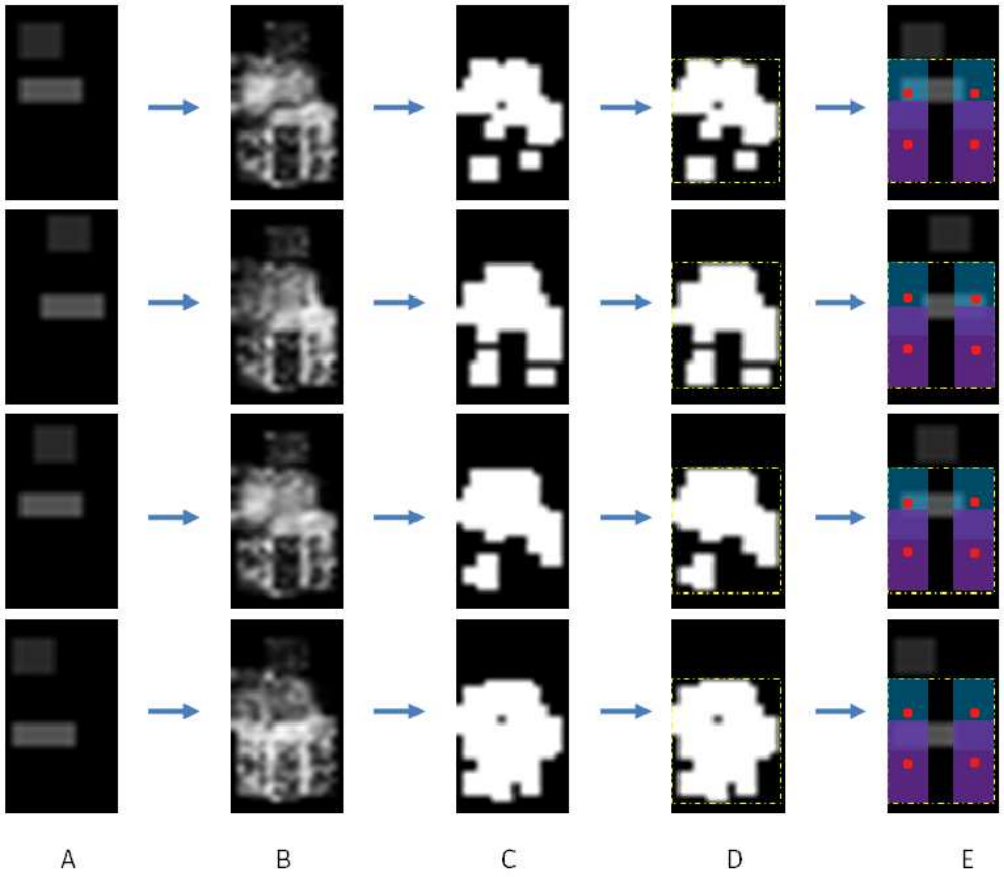


Figure 5.3: *Procedure of generating artificial feature maps. A. Find the head and waist then generate a partial feature map; B. Feed this partial feature map into the associator and produce a raw feature map; C. Applying image processing methods (erosion, dilation etc.) to remove the head patch and generate binary image patches; D. To separate arms and legs, I find the smallest rectangle containing these image patches and use its coordinate to decide the locations of proposed arms and legs; E. A feature map is proposed.*

value gave a better classification rate and was chosen for the experiment.

$$L(x, y) = \frac{d^2 src}{dx^2} + \frac{d^2 src}{dy^2} \tag{5.1}$$

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{5.2}$$

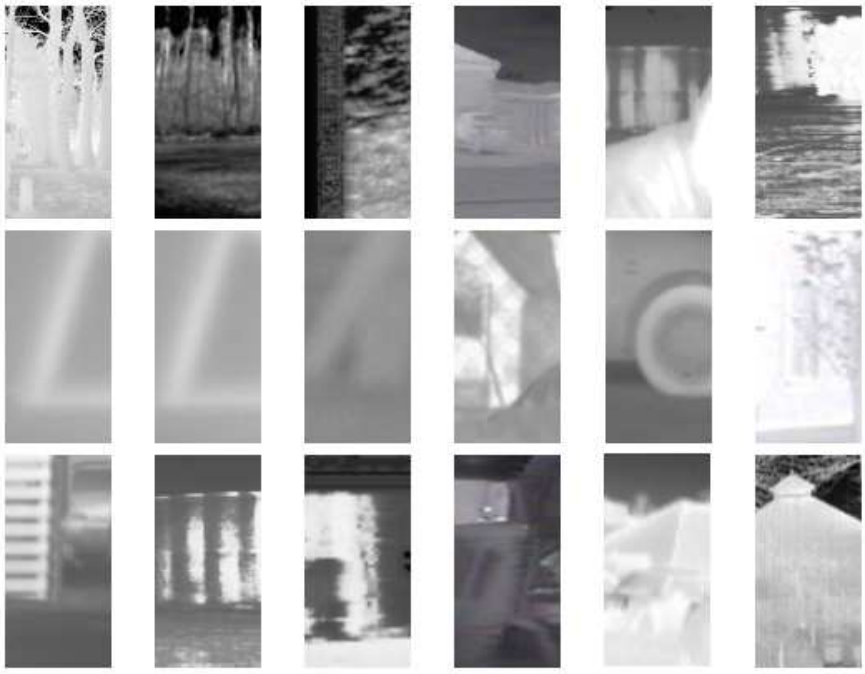
5.2 Database Used

The source database used is provided by Digital Image Processing Laboratory at Cranfield University. All images in the source database were manually extracted from videos shot in a natural environment, mainly at Cranfield University. The image size is 64×128. Person bodies within these images cover around 60% of the image height. Spreading limbs cover 30-60% of the image width. The body central line is as close to the image central line as possible. The ambient temperature was varied when shooting images, so the contrast between people and background is also varied. The people are also in different poses and at varying distance.

There are 1216 positive images in the database. These images are front or back of person bodies. 1000 of them are taken for training and the remaining 216 images are for test, selected by random sampling. Also there are 1240 negative images in total. 1000 of them are taken for training and the remaining 240 images are for test, selected by random sampling. Figure 5.4 shows some positive and negative samples of the training sets. Apart from the front and back person images, there are also 1133 side person images. Some 210 of these images, which are unseen by the system when training, are randomly chosen to test the generality of the system.

When training the first layer of this system, the people are extracted from those 1000 training images. Then key components are extracted from these thermal images. Originally, these key components include head, shoulder, waist, arms, legs, upper up body, and lower up body. But I found the relatively small components could not cope with some difficult person gestures and gave many

A. Samples of positive thermally-imaged person. They were taken at different time. The background and temperature are various.



B. Samples of negative thermally-imaged person. They were taken at different time. The background and temperature are various.

Figure 5.4: *Some person samples used for training* Cengji *to detect person from complex thermal background.*

false positive detections. These gestures are mainly with stretched arms and legs. These stretched arms and legs occupy bigger areas than naturally pendulous ones. If I use small rectangles to extract both vertical and stretched limbs, some limbs will be cut out therefore some of the gestures can not be covered. The classifier will be confused by these partial gestures. A relatively bigger image patch size will ease this confusion to the classifier. So I re-chose *head*, *waist*, *Left-upper body*, *Right-upper body*, *Left-lower body* and *Right-lower body*. They are extracted from the training images manually. Because the person bodies, especially limbs, have complex gestures, it is difficult to set strict rules of extracting component images. But I tried to follow a consistent process. For the head feature, I tried to keep the neck central line to be overlapping with the image central line and keep a certain part of the shoulder (top of the shoulder is beyond 1/3 but below 1/2 of the image patch height). When extracting the waist feature, its centre is overlapping with the image central line and the vertical centre of it is often the separation line between the clothes and the trousers. For the remaining features, I tried to cover as many as possible gestures within image patches. Hands and feet were included within these image patches. All these extracted components have a unique label from 1-6 in turn. The background has a label of 0. All the negative training data for these features were randomly extracted from those images not containing these features. Table 5.1 shows the details of the final version feature training sets. Figure 5.5 shows some samples of body features in the positive training sets. More training samples are shown in Section A.2, Appendix A.

For the second layer, the input is a body feature map which is composed of feature labels. It comprises the information of feature sizes and the geometrical relations among them. Positive and negative targets will be discriminated by their feature maps. Both the positive and negative training sets for the second layer are generated by the first layer. The training sets creation procedure has been introduced in Table 3.1, Chapter 3. The training samples for the autoassociator are the body feature maps generated by the first layer. When training the associator, these feature maps are presented to both the input layer and the out-

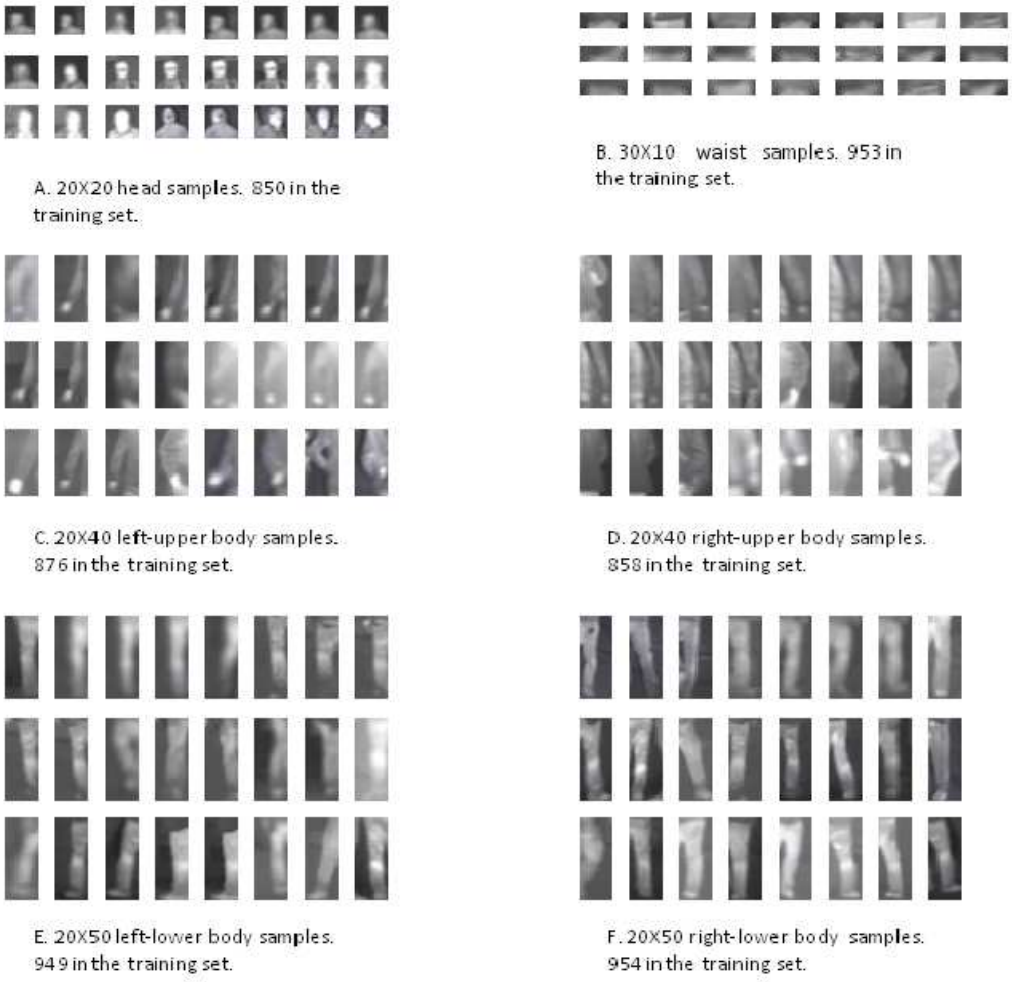


Figure 5.5: *Some body feature samples used for training* Cengji *to detect person from a complex thermal background.*

Feature Name	Number of Images		Image size
	Positive	Negative	
Head	850	900	20×20
Waist	953	970	30×10
Left-upper body	876	912	20×40
Right-upper body	858	900	20×40
Left-lower body	949	977	20×50
Right-lower body	954	970	20×50

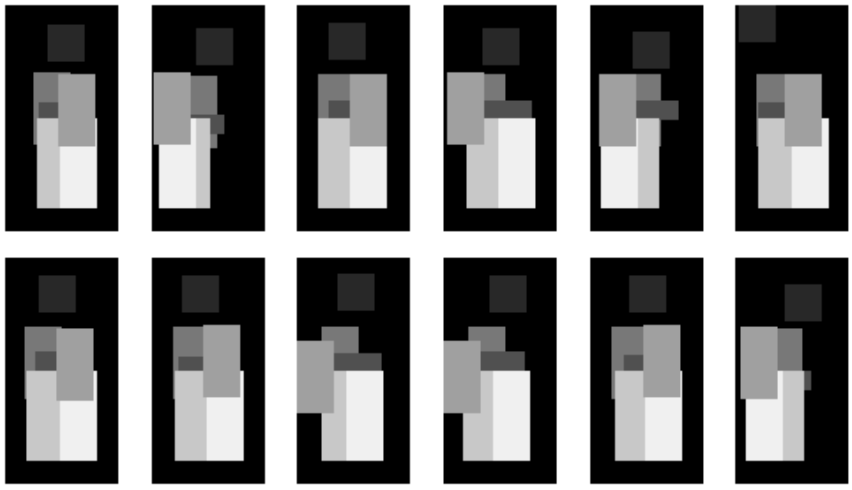
Table 5.1: *Feature Datasets details.*

put layer. Figure 5.6 shows some samples of positive and negative body feature maps.

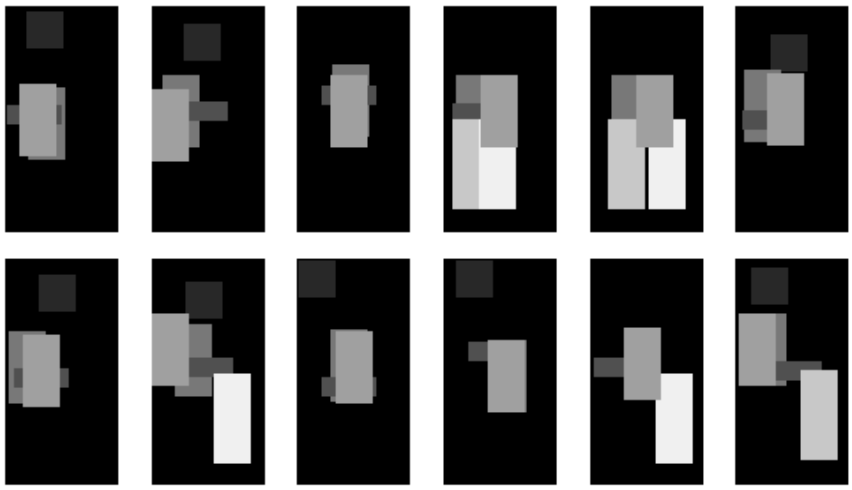
5.3 Training Procedure

The main training difference between this experiment and the previous face classification experiment is the feature clustering. In the face classification experiment, the first detected features are directly used for generating feature maps. But in this experiment, detected features will be clustered to decide a location to reduce false positive detections due to the much more complex scenes. The threshold for the feature clustering has to be decided during training. There are 6 features, therefore 6 clustering thresholds need to be decided. The feature classifier kernels and their parameters are decided by grid search. When deciding the threshold, I applied each classifier to find features in whole person images in the feature training sets. For each feature classifier, I calculated the average distance from the first clustered feature to the real feature in each image. The distance is from the clustered image patch centre to the actual feature patch centre.

Once the thresholds of clustering feature patches have been decided, feature classifiers will be used to generate feature maps. A feed-forward neural network is used to store the association memory. It is trained using these generated feature maps. Apart from the feature clustering threshold, the other training



A. Samples of positive body feature maps.



B. Samples of negative body feature maps.

Figure 5.6: *Some body feature map samples used for training* Cengji *to detect person from a complex thermal background.*

procedure is similar to that used in the face classification experiments. It follows the same training strategy described in Section 3.4, Chapter 3. The kernels and kernel parameters of each classifier are decided by using a grid search. During the training iterations, every time the classifier changes, the threshold should be recalculated. I used a 5-fold grid search to find the best kernels and optimize their parameters. The RBF kernel performed the best for each feature classifier. Table 5.2 shows the parameter configurations of each feature classifier and their performance during the 5-fold cross validation.

Classifier Name	Cost	γ	5 fold correct rate (%)	Clustering threshold
Head	5	0.05	98.8877	2
Waist	4	0.07	98.443	2
Left-upper body	9	0.04	97.4399	3
Right-upper body	7	0.03	97.2046	3
Left-lower body	9	0.01	97.9366	3
Right-lower body	10	0.04	97.6406	3

Table 5.2: *Feature classifier parameters and their performance.*

The training of the autoassociator is relatively independent of the training of those two layers. It is a feed-forward neural network trained using back-propagation [16, 86, 25, 77]. The training samples are those body feature maps generated by the first layer. They are presented to both the input and output layers. In order to speed up the training, I resized these images from 64×128 to 16×32. The autoassociator is a 3 layer feed-forward neural network with 1 input layer, 1 output layer and 1 hidden layer. There are 512 nodes in the input and output layers respectively. The hidden layer has 500 nodes which is decided by experiment. I set a very small stop error, 0.00001, then trained the neural network with different hidden layer numbers. The training can not reach this stop error. But after certain training period, the training error will be stable. 100 node hidden layer configuration gave stable error at around 0.9. 200 gave stable error at around 0.75. 300 is stable at around 0.72. 400 is stable at around 0.12. 450 is stable at around 0.07. 500 node hidden layer gave the lowest error 0.05. 510

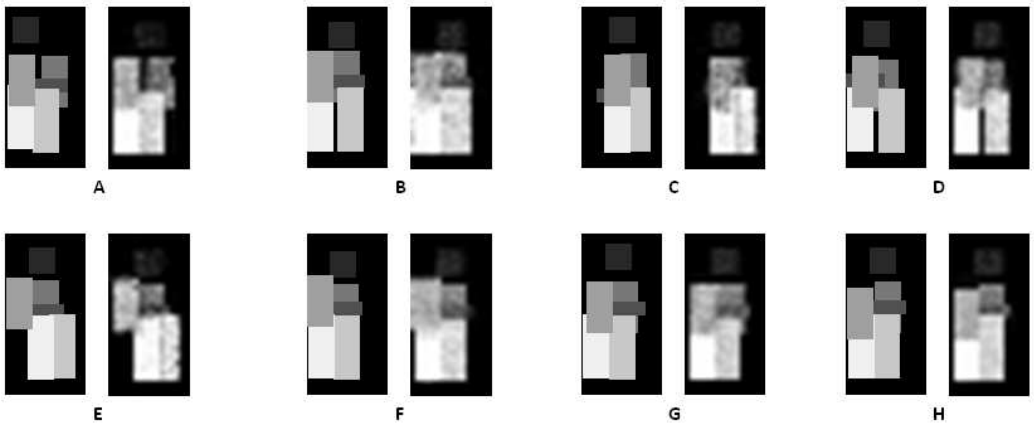


Figure 5.7: *Training test on person feature map associator.*

gave almost the same error. The stop criteria is set as the average output error of each sample is smaller than 0.05. 900 person feature maps were used to train the associative feed-forward neural network. Figure 5.7 shows some test samples of the body feature map associator. The inputs are body feature maps generated by the first layer but not used to train the associator. All together 100 person feature maps were used to test the performance of the associator. An average error of 0.07 was achieved. These test images were all resized from 64×128 to 16×32.

For the additional single-layer SVM classifier, the input is also the combination of raw pixels and Laplacian values. The calculation of this input is same as those person body features. I applied a 5-fold grid search to find the best kernel and its parameters. During this search, the RBF kernel performed best with cost C of 10 and γ of 0.004. The correct rate with this kernel and its parameter configuration is 99.5595%. The Haar like feature classifier also takes the same input.

5.4 Experiment

5.4.1 Method

In this experiment, the performance of the dual-layer system with and without attention mechanisms is tested. Additionally, as in Chapter 4, I trained and applied a single-layer SVM classifier and a Haar like feature classifier to the test set. The single-layer SVM classifier and the Haar like feature classifier were trained on those person and non-person images which were used to create feature training sets.

When applying *Cengji WA*, all classifiers scan over the whole input images from the top left corner to the bottom right corner. These classifiers report all detected feature image patches' sizes and locations to the system. The system then post-processes these patches by clustering them together to form new detections. Those clusters containing image patches less than the thresholds will be discarded. The first post-processed detections which satisfy the thresholds will be taken as the final detections for each feature. These final detections are used to form a feature map which is the input to the second layer.

When applying *Cengji AM*, head and waist classifiers scan over the whole input images. Their outputs will be post-processed to form clustered detections. The system will use the first clustered detections to form a feature map. This proposed feature map will lead other classifiers, two upper body and two lower body classifiers, to search in the areas expected to contain features. In these areas, the feature classifiers will search for each feature and report any found to the system. The system will then cluster them together to form final detections. These final detections and the previous detected head and waist will then be composed to form a final feature map for the second layer to decide if this feature map belongs to a person.

Set Name	Classification result			Correct(%)
	Correct	Incorrect	Total	
People	201	15	216	93.06
Non-people	192	48	240	80.0
Total	393	63	456	86.8

Table 5.3: *Person classification results of the* Cengji AM.

Set Name	Classification result			Correct(%)
	Correct	Incorrect	Total	
People	203	13	216	93.98
Non-people	194	46	240	80.83
Total	397	59	456	87.06

Table 5.4: *Person classification results of the* Cengji WA.

5.4.2 Results

Firstly, the thermal target detection was performed on the 456 test thermal images, 216 person and 240 non-person images.

Experiments on Cengji. With the attention mechanism, 201 people were successfully detected. 15 were misclassified. Among non-people, 192 of them were correctly detected as non-people and 48 were misclassified. The total classification rate is 86.8%. Without the attention mechanism, 203 people were successfully classified. 13 were misclassified. Among non-people, 194 of them were correctly classified as non-people and 46 were misclassified. The total detection rate is 87.06%. There were more non-people misclassified as people. This is because this system emphasizes on detecting as many targets as possible. Therefore the feature map tends to give a loosen restriction on the non-person targets. It can be adjusted by changing the classification boundary. The performance of the *Cengji AM* is equivalent with that of the *Cengji WA*. See Table 5.3 and 5.4.

Figure 5.8 shows some successful samples.

Figure 5.9 shows some failed samples.

Figure 5.10 and Figure 5.11 compare the number of calls to feature classifiers

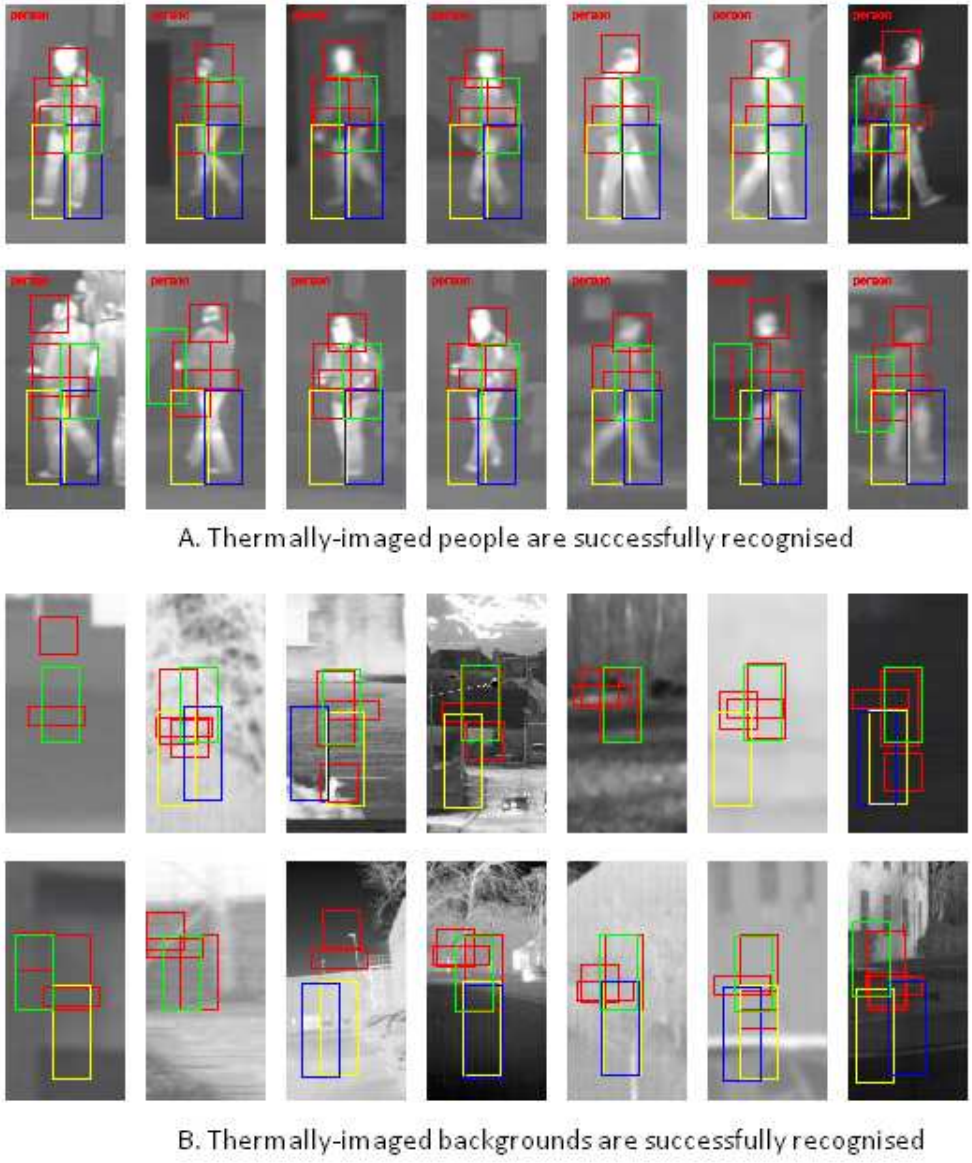


Figure 5.8: *Successful person classification. A. People are successfully recognized in thermal images. B. These photos are successfully recognized not containing any person images. Though there are some person features are found, but they do not compose a complete person. The structure configuration among them does not belong to a person.*

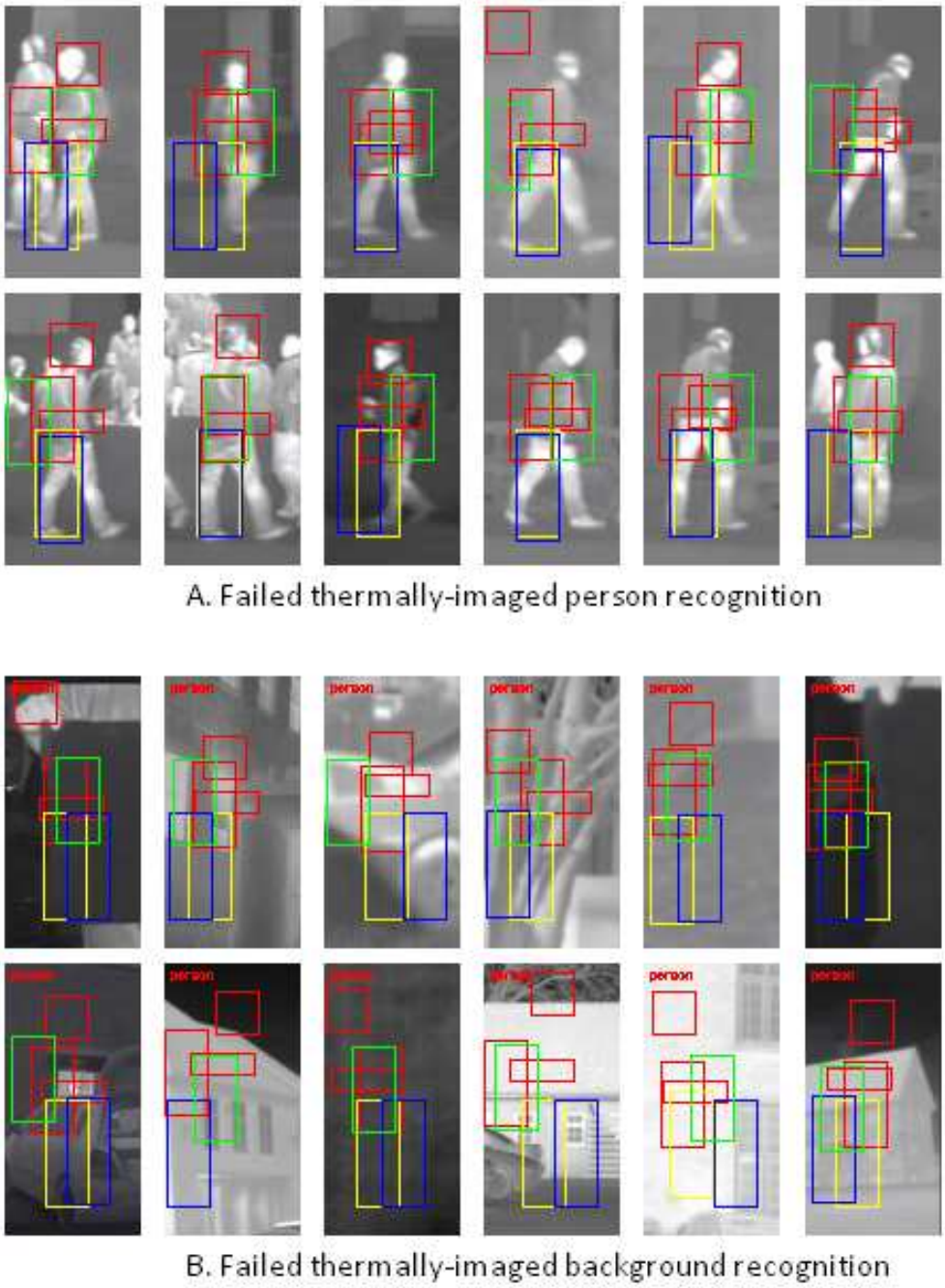


Figure 5.9: *Failed person recognitions. A. People are not recognized in thermal images. Though there are some person features are found, but they do not compose a complete person. The structure configuration among them does not belong to a person. B. These photos are wrongly recognized containing person images. This kind of failure is caused by individual classifiers. They gave false positives and these false positives form a feature map belonging to a person.*

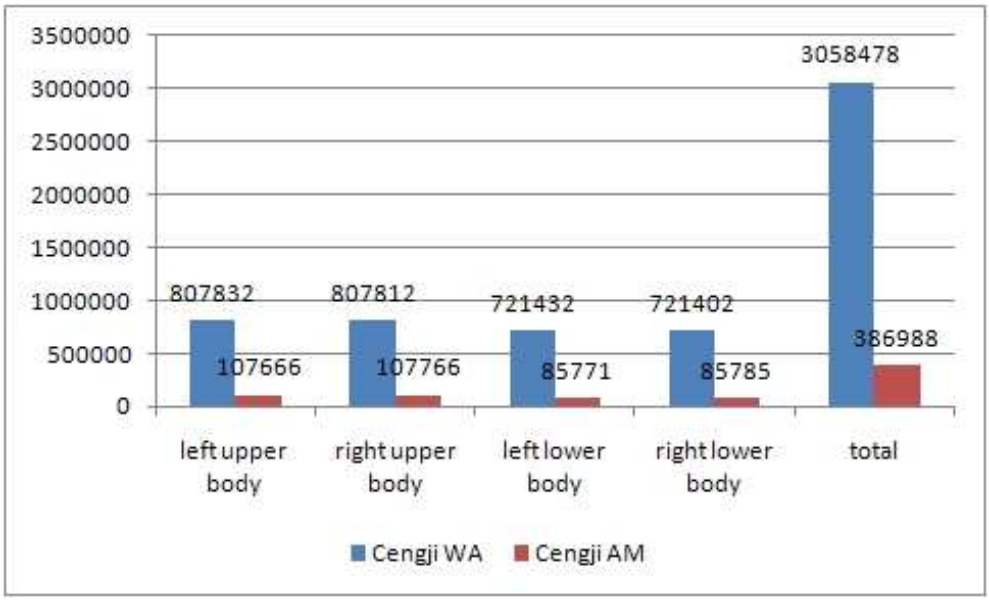


Figure 5.10: *Calls to SVM feature classifiers by* Cengji *with and without attention mechanism. This table shows calls on SVM feature classifiers when applying* Cengji *on 216 test people. Blue bars are of* Cengji WA *and red bars are of* Cengji AM*. The search of the heads and waists are not influenced by the attention mechanism. It is not included in this figure. Attention mechanism saves up to 87.35% SVM calls.*

within *Cengji WA* and *Cengji AM*. The calls to head and waist classifiers to detect these two features are fixed call numbers both for *Cengji AM* and *Cengji WA* because the search for these features are the same. For the remaining features, the search is influenced by the attention mechanism. These figures show that the application of attention mechanism saves SVM calls up to 87.73%, 375,246 times with attention mechanism compared with 3,058,478 times without them on non-person images. For the person images, the saving of SVM classifier calls is also up to 87.35% when applying the attention mechanism.

Experiments on single-layer SVM classifier and Haar like feature classifier. In the additional test on the single-layer SVM classifier, its performance is equivalent to those of the dual-layer system. 211 out of 216 test people were successfully classified as people. 186 non-people were successfully classified from 240 non-

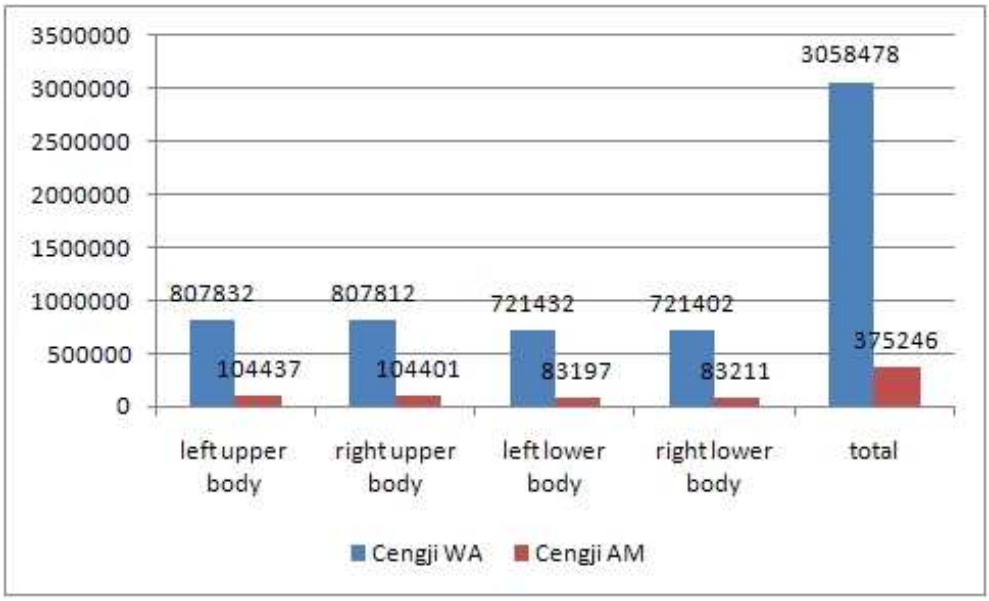


Figure 5.11: *Calls to SVM feature classifiers by* Cengji *with and without attention mechanism. This table shows calls on SVM feature classifiers when applying* Cengji *on 240 test non-people. Blue bars are for* Cengji WA *and red bars are for* Cengji AM*. The search of the heads and waists are not influenced by the attention mechanism. It is not included in this figure. Attention mechanism save up to 87.73% SVM calls.*

Set Name	Classification result			Correct(%)
	Correct	Incorrect	Total	
People	211	5	216	97.69
Non-people	186	54	240	77.5
Total	397	59	456	87.06

Table 5.5: *Person classification results of the single-layer SVM classifier.*

Set Name	Classification result			Correct(%)
	Correct	Incorrect	Total	
People	199	17	216	92.1
Non-people	191	49	240	79.6
Total	390	66	456	85.5

Table 5.6: *Person classification results of the Haar like feature classifier.*

person images. Totally, 397 images were successfully classified from 456 test images. The success rate is 87.06%. See Table 5.5.

With the Haar like feature classifier, 199 out of 216 test people were successfully classified as people. 191 non-people were successfully classified from 240 non-person images. Totally, 390 images were successfully classified from 456 test images. The success rate is 85.5%. See Table 5.6.

Using different dataset. Secondly, I tested *Cengji* with different images from a different dataset. In the first experiment, images are person fronts and backs. Images for this experiment are people taken with a side view. Sometimes some features of these people are occluded. For example, there may be only one arm or one leg visible. There are also some blurry images. Some images have a complex background. All of these set more challenges to a feature detection system. Figure 5.12 shows some samples in this dataset.

In the generality performance test, 210 side body images were used. Among these images, 196 of them were correctly classified as people by *Cengji AM* and 195 by *Cengji WA*. The detection rate is 93.33% and 92.86% respectively. Figure 5.13 shows some sample recognitions from this image set.

Again, the single-layer SVM classifier gave an equivalent generality perfor-

Figure 5.12: *Samples of person sides used to test the generality performance of* Cengji. *These images are side bodies. Some body parts, for example arms or legs, are occluded. There are also complex backgrounds.*

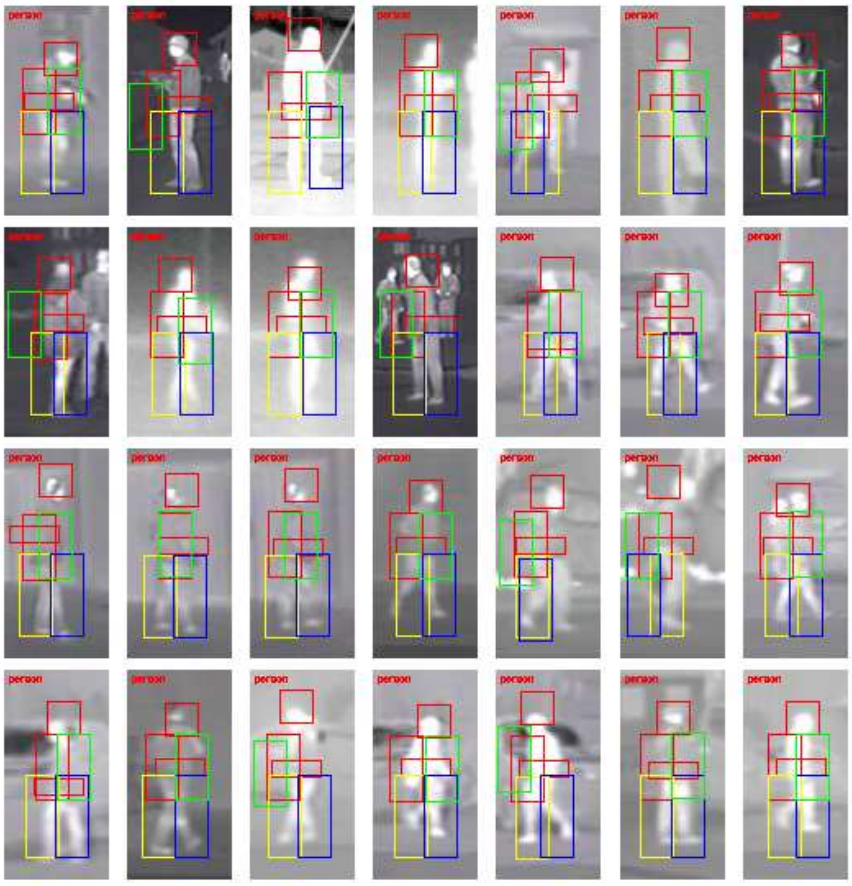


Figure 5.13: *Samples of side person recognition. These images are side bodies. Some body parts, for example arms or legs, are occluded. There are also complex backgrounds.*

mance compared with the dual-layer system. 198 from 210 test images were successfully classified. The classification rate here is 94.28%. The Haar like feature classifier successfully classified 192 "side" people images. The correct rate is 91.42%.

5.5 Discussion

In this chapter, I applied the system to classify people in infrared images. In Chapter 4, face classification experiments have verified the feasibility of the component based hierarchical structure and the positive effects of the attention mechanism. This experiment is to explore the performance of the system using a different dataset.

In the experiments, I applied *Cengji AM* and *Cengji WA* to detect people in IR images. The attention mechanism has displayed good performance in this experiment. One of the main differences between this experiment and previous experiments is the clustering of detected features to form a uniform detected feature patch. This is to reduce the false positive rate due to the more complex target and background. The introduction of feature clustering required a procedure to find the best clustering threshold for each classifier. Otherwise the training strategy is the same as the face classification experiment, the main procedures are also identical.

There are two parts to this experiment. The first part is to use a dataset similar to the training set to demonstrate this system works on different inputs. Another part is to use a difficult and unseen dataset, in which people have different gestures and illuminations, to demonstrate generalisation. In the first part, 201 of 216 people were successfully classified as people by *Cengji AM*. 15 were misclassified. Among non-people, 192 of 240 were correctly classified but 48 were misclassified. The total classification rate is 86.8%. In the second part, 210 images were used for generality performance test. Among these images, 196 of them were correctly classified as people. The classification rate is 93.33%.

The second part of the experiment shows the system works on unseen datasets

with more difficult targets and background. In this part, some target person features, for example arms or legs, were occluded. But these people can still be properly classified. Though some of these difficult images can not be identified as a person because they are too different from the learned people, part of their body components are detected. This gives the system a chance to find target features. A user can set up their own rules to decide the detection of targets. For example, a person exists as long as there are more than two body components are detected. This kind of function is a useful property of component based system to deal with occlusion. In this part, *Cengji AM* gave a correct rate of 93.33%.

Though the detection performance of *Cengji AM* is slightly worse, it saves calls to related classifiers dramatically by up to 87.73% under the search strategy introduced by the previous sections.

The training of these individual classifiers is decisive to the system. Our strategy of training this multi-layer system is effective. The person image database is much more complex than the face database. Targets have much more variety in terms of appearance, gesture and illumination. These variations put a great challenge on the system training. With the selection of feature content and training patch sizes, the system learns these images effectively with the training strategy proposed in this thesis.

The experimental results demonstrate that our system achieves a good result in person detection from infrared images. The hierarchical system structure and feature map concept work properly. The attention mechanism helps the system find features by leading the system to the areas with higher likelihood of containing a new feature. Also this system displays good generalisation performance.

5.6 Extension

This system is being extended to detect people in different sizes in a big real-world image. The challenge is that the ratio between the feature size and the input image size is not stable. In the face classification experiment, the feature sizes were constant. It was not necessary to consider the variety of feature sizes in

the image when the main purpose of the face experiment is to verify the feasibility of this system. But when processing more realistic images, it is inevitable the feature sizes will have some variety therefore the feature search is more difficult than that of the face classification experiment. To cope with the variety of feature sizes, the size of the feature grid will change during each search. For example, the size of the original head feature in the training set is 20×20, but in an input image there might be head features of sizes up to 30×30. In order to find these heads, the grid size will have to increase. The grid with size of 20×20 will scan over the image and record all heads found. Then the grid will increase its size and search the image again until its size reaches 30×30. Figure 5.14 shows some successful person detections by the system with attention mechanism. A is a relative easy detection. Person is similar to those in training set. Person in B is blurry due to the vibration of the camera. But the system successfully found the person. C contains a semi-occluded person. His left arm and leg are occluded by the wall. The system identifies this image patch as a person correctly. D contains no person and therefore there is no detection of person.

Figure 5.15 shows some images with false positive person detections by the system with attention mechanism.

The single-layer SVM classifier and Haar like feature classifier were also applied. They are not good at detecting occluded people. For example, these two classifiers missed the person in Image C. At the same time, they also gave false positive detections. The false positive detections produced by *Cengji* can be reduced by using a higher threshold on feature cluster. The improvement of feature classifier performance will also help to improve the system performance.

I am also aiming to further improve this system's speed to be a practical system. For example, to process PAL (720×576) format in a speed of 1 frame/second. Current speed is 30 seconds per frame on such kind of images using a 2.33 GHz, 3 G RAM computer. To achieve the speed of 1 frame/second, in the first layer, two kinds of classifiers will be used. The first kind of feature classifier will have a fast classification speed, for example, Haar like feature classifiers. This kind of classifier will give a higher false positive rate. So a SVM

Figure 5.14: *Successful samples of person detections.*

Figure 5.15: *False positive detections exist in some images.*

feature classifier, which has better classification performance, will be engaged to verify the output of the Haar like feature classifier. The high speed system has the similar structure as that in Figure 4.3 and Figure 5.1. Figure 5.16 shows the structure of the system which is used to detect people from infrared images with higher speed.

When a thermal image is presented to this system, the Haar like feature classifier of head is called to scan through the image to find patches containing head images. Any feature classifier can be the first one to be called. The benefit of calling the head classifier as the first one is that there are many patches containing features similar to a head hence using head classifier will locate as many of these patches as possible. This will benefit a system which does not want to miss any possible targets. However this may bring in some false positives if each single classifier does not perform well. Therefore this strategy puts a relatively high demand on each feature classifier and their training. But the training method can compensate this to some extent as discussed in Chapter 4. The head classifier will scan over the image pixel by pixel and record every location of head patches for post processing. Patches with similar sizes within a certain area will be clustered together to form a single patch which will be treated as a single head.

These clustered head patches then will be verified by a SVM head classifier. If the SVM head classifier classifies a candidate head patch as a head, its coordinate and size will be passed to the system to create a sub-image for detection of remaining features. The size of this sub-image is decided by the head patch's size. The rule is that the head centre line should be identical with the centre line of the image in width direction and its top should be 1/6 length to the top of the sub-image. After this sub-image is created, this system will search for other features. The second feature to be searched can be chosen randomly. For the simplicity of the experiment, I chose waist as the second feature to be scanned. The scanning strategy is the same as that of the head scanning. The waist classifier finds all candidates and sends them to the system for clustering and verification by a SVM classifier. The SVM verified candidate which has most members will be taken as the final detection. Within this sub-image, there can

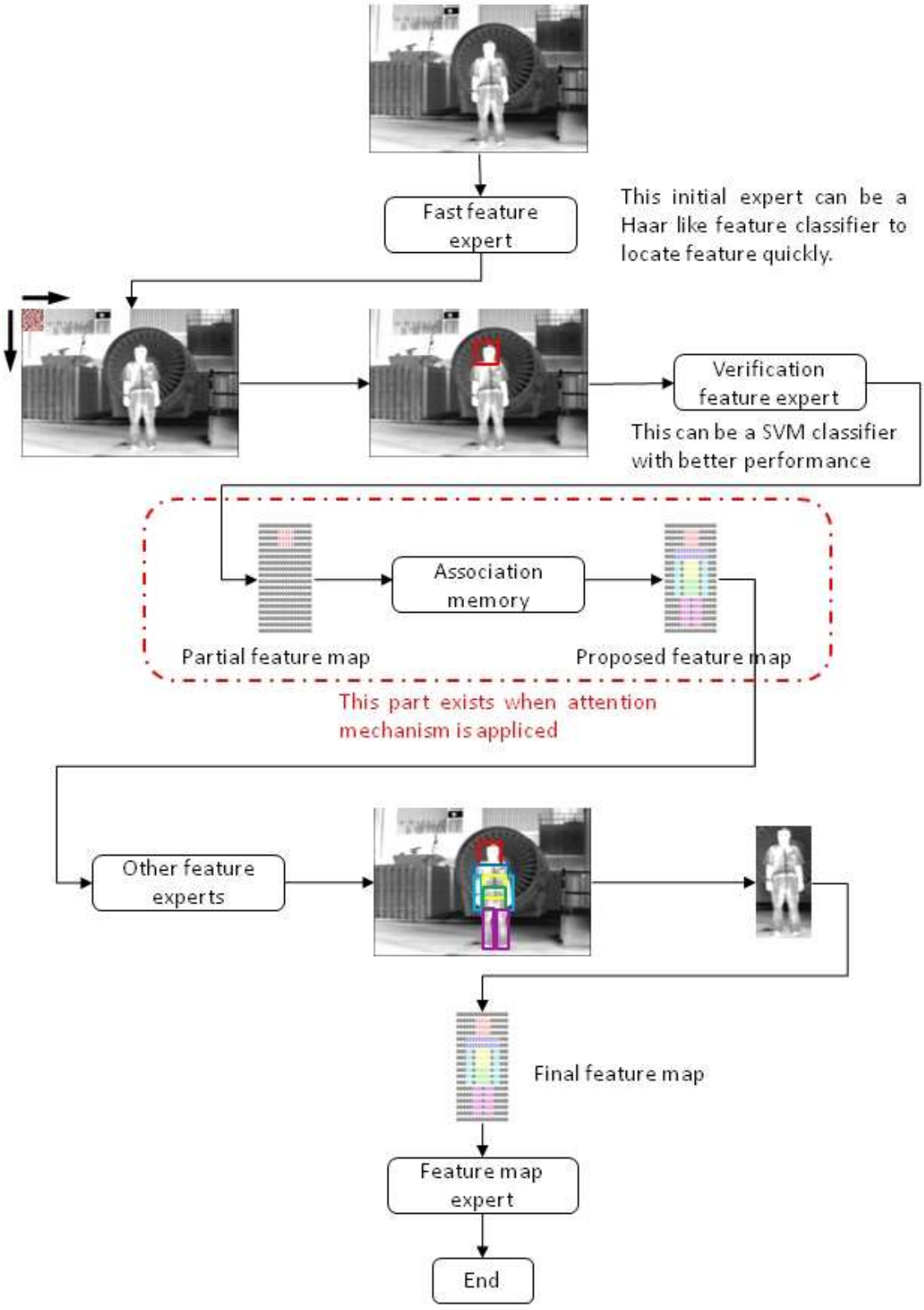


Figure 5.16: *Structure of Dual-Layer* Cengji *for the extended experiment*

also be two kinds of classifiers, one for high speed search, another for verification of detected features.

When the attention mechanism is not engaged, *Cengji* will scan over this sub-image for all remaining features using the same procedures used by previous feature classifiers. There may be more than one sample (cluster of image patches) of a feature accepted by its SVM classifier. I took the one with most members as the finial detection of these features to compose the feature map of a person. This feature map will be the input to the second layer.

When the attention mechanism is engaged, *Cengji* will use the detected head and waist features to compose an expected feature map using association memory. This memory will indicates other features' possible locations and sizes. The system will go to the suggested locations to search for these features in relatively smaller areas compared with areas without using the attention mechanisms. This procedure will confine the search in areas with higher likelihood containing a feature. The search strategy within these areas is same as that of head and waist feature search. Once all feature searches suggested by the expected feature map are complete, the detected features will be sent to the system to compose the final feature map for the second layer to classify.

The trained first layer and second layer SVM classifiers are exactly the same of those used in the person detection experiment without Haar like feature classifiers. The input to the first layer classifiers is the combination of raw pixel value and Laplacian value. The input to the second layer is the feature map generated by the first layer.

Chapter 6

Conclusions and Future Work

In this thesis, I have developed a component-based dual-layer *Cengji* system and applied it to face classification and recognition of people in thermal images. This system applies an attention mechanism to help locate features and improve scanning speed. The experiments verified the feasibility of the hierarchical structure of this system and the application of the attention mechanism. The performance of *Cengji* with and without the attention mechanism was also tested.

6.1 Summary

At the beginning of the thesis, in Chapter 1, I set up my objectives. It aims to make progress towards an object recognition system using a dual-layer classifier. To achieve this objective, a component-based trainable computer vision system which can detect salient features in images and integrate these features into complex, hierarchical representations of meaningful objects is designed.

In Chapter 2, related concepts, algorithms, methodologies and other researchers' applications which are relevant to this thesis are reviewed. Through these reviews and discussions, the background of the developed system has been explained. This object recognition system is inspired by the ability of a human vision system which can recognise an object effortlessly compared with a computer vision system [75, 39, 9]. The ability of the human vision system is based

on the fact that the human brain stores knowledge of objects in a hierarchical manner [100, 38, 2, 12]. The application of the attention mechanism is a significant characteristic of this system. With the help of the attention mechanism, objects are recognised more quickly and more accurately when they fall within a familiar schema [81, 91, 11, 67]. Figure 6.1 shows an extended view of the hierarchical structure of this system.

Chapter 2 also reviewed machine learning algorithms used in the thesis, including Support Vector Machines, Neural Networks and Haar Like Feature Classifier. The SVM has high classification performance. A feed-forward neural network trained with back-propagation method can act as an auto-associator. Haar Like Feature Classifier is used as comparison. Heisele *et al* [53, 55] have developed a dual-layer SVM algorithm learning discriminative components (features) of objects similar to the system introduced in this thesis. These two systems are both dual-layer systems. Their system uses a collection of 14 features on the first layer to classify a face. My system uses only 4 features. Their system uses a vector to connect two layers but my system uses a feature map. Also, I employ an active attention mechanism [44, 45] to speed up the system. The system training strategy also presents some differences. Their system trains each layer separately. My system joins two layers together to get the best training results from limited training samples. The aim in my system was to automate as much of the training as possible.

In Chapter 3, the developed system is described in detail. As an object recognition system, the basic components of this system are individual *classifiers* which classify features, objects, and relations among them. These classifiers are linked together according to the hierarchy among features and higher level objects. This system utilizes a combination of both bottom-up and top-down processing to interpret visual objects. This system then uses detected features and bottom-up knowledge to form a higher level object. This process will stop at the top of the object hierarchy. It can use top-down knowledge to figure out what are other relevant features or objects and where and how to find them when some features have been detected. A *feature map* is introduced to act as a medium to

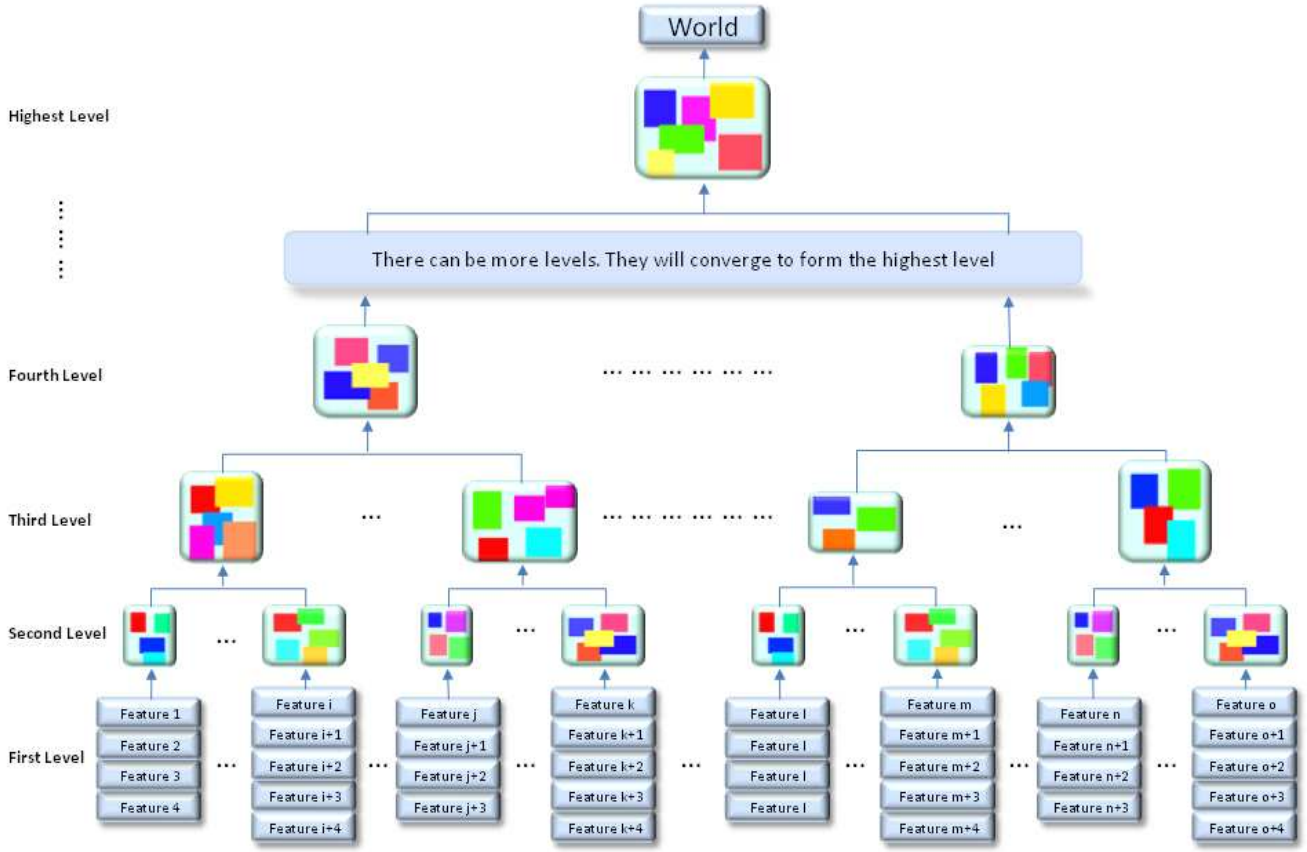


Figure 6.1: *The* Cengji *system. The system takes an image as input, returning a classification of that image. Internally, it consists of two layers, communicating through a feature map. The number of classifiers can be varied, depending on the domain. The autoassociative memory is not shown.*

convey object structure directly; the feature map has the advantage of preserving all relative positions among features explicitly. For this machine learning based system, I designed a training strategy to train it. During the training, two layers are joined together to reach the best performance with limited training samples. The procedure has been shown in Table 3.1.

Face classification and thermal image experiments are conducted in Chapter 4 and Chapter 5. In the face classification experiments, a face is represented by four features, two eyes, nose and mouth. In the thermal image experiments, a person is represented by six features, head, waist, left upper body, right upper body, left lower body and right lower body. *Cengji AM*, *Cengji WA*, a single SVM classifier and a Haar like feature classifier are applied to these experiments. The experimental results demonstrate that the system achieves good results in object recognition. The feature map concept works properly for the system. The dual-layer system with and without attention mechanism displayed good performance. The experiments indicate that the performance of the first layer is crucial to the whole system. The first layer detects individual features which form higher level objects. These features are the basis of further analysis. The construction of higher level objects depends on them. The performance of individual feature classifiers decides how the first layer performs.

A significant characteristic of this system is the application of the attention mechanism. The main purpose of applying the attention mechanism is to improve the system performance of locating features, especially the speed. In the experiment of face classification, applying this mechanism has significantly improved the face classification speed compared with *Cengji* without applying it. In the thermal person recognition experiment,the saving on SVM classifier calls is more significant than that of the face classification due to the feature detection strategy which searches over the whole suggested feature areas. After the seed features have been found, detection calls to SVM classifiers within the target area are clearly saved.

6.2 Contribution

The contributions to the object recognition domain made by this thesis are the following:

- A component-based object recognition algorithm. This algorithm represents an object by its salient features and these features are organized in a hierarchy. A dual-layer object recognition system applying this algorithm has been constructed. Face classification and person detection experiments have been conducted to verify and test the feasibility and performance of both the algorithm and the system. This system can discriminate faces from non-faces and detect thermal people from complex background even when there is occlusion on the objects.

- Application of the attention mechanism to help locate features. The system learns the geometrical and hierarchical relations among features composing the object and utilizes it to locate other features when some features are found. The application of the attention mechanism has effectively reduced the calls to feature classifiers therefore improving the system efficiency.

- Comparison between different algorithms. This thesis compared the performance of the developed system with and without attention mechanism, against a single-layer SVM classifier and a Haar like feature classifier on both face and person classification. In the face classification experiments, the single-layer SVM classifier has better detection performance but in the person detection experiments, the system displays equivalent performance. The experiments show that this system has better generalization performance than other algorithms. When applying the attention mechanism, this system has higher efficiency and equivalent object recognition performance.

- A feature map concept is introduced as an internal representation. It is a representation of an object by retaining spatial relations between features.

- An effective training method for training the dual-layer system. The dual-layer system is considered as an integrated system. The creation of feature training sets is coupled to the creation of the feature map training set. The two layers have mutual influence on each other. It is a closed loop to create training sets for both layers. This method is an effective way to create an optimized system with limited training samples.

- Another contribution is the databases created for the component based-face and thermal person classification and detection. The face classification experiment source images come from several existing face databases. Face features are generated from these images. Person images are provided by Cranfield University. Person features are extracted from whole person images. These feature training sets have been optimized for the best performance of *Cengji*. But they can still be used for other systems.

6.3 Conclusions

This thesis has developed a novel trainable algorithm and provided a framework for an object recognition system. Object recognition is often the first step towards scene understanding, that is extracting individual objects from a complete scene. Most scene understanding computer vision systems have a component for detecting objects from background. I have presented a hierarchical component-based object recognition system, *Cengji*, and applied this system to face and person recognition. The first layer is used to detect individual features composing a higher level object. These features are organised using a feature map which mainly contains structure information of the object. The second layer takes a feature map as the input and judges if this feature map belongs to an object. A significant characteristics of this system is the integration of the learning of compound objects with the attention mechanism which guides the system to areas likely to contain relevant features. Because this is a machine learning based object recognition system, I also developed a training routine fitted to this multi-layer structure.

The experiment of the face classification has verified the feasibility of the structure of this system and the positive effect of the attention mechanism. The experiments of thermal person recognition has displayed the generality performance of this system. Compared with the face classification experiment, the targets to be detected have more complex structure with components have more freedom regarding its location. Hence the feature maps of the thermal person have much more varieties than the face feature maps. In these experiments, both *Cengji* with and without attention mechanism depended on their first layer deeply. It decides the input to the second layer. The performance of the first layer has a fundamental influence on the entire system performance. Therefore the training of individual feature classifiers is an emphasis of this component-based system. The strategy designed for this system aims to achieve the best performance with limited training sources. It joined two layers together as a whole. This strategy can compensate the training imperfection existing within each feature classifier by tolerating it in the second layer. This is proved to be very useful to the system because it is difficult to create perfect feature classifiers even with comprehensive training samples. The second layer also helps to improve the first layer training by providing a chance to get some key training samples, especially negative samples.

It is difficult to generate perfect feature classifiers.I adopted several ways to reduce false positives. Adding more negative samples into training sets of first layer feature classifiers can improve the classifier training and reduce some false positives. But it is impossible to collect all kinds of negative samples and adding too many negative samples will make the training unfeasible. Another direct way of reducing false positives is to tune the parameters of classifiers by moving the classification boundary towards the positive side which will give a stricter criteria to take a sample as positive therefore reducing the false positive rate. The negative influence of this method is that more false negatives will be generated. In the thermal person experiments, when detecting thermal targets from complex background, more than one detected targets were clustered to form a final detection. Only the detection having more than certain number

of neighbours with similar size existing within certain areas will be taken as a valid detection by clustering itself with its neighbours. This clustering detection adoption strategy was shown to reduce the false positives effectively.

In order to get the optimized classification performance, I put great emphasis on training classifiers, especially those SVM feature classifiers. Table 3.1 gives the details of our unique training methods for the dual-layer machine learning based system. This method can find the best configuration for a SVM feature classifier, the kernel type and its parameters. The second layer feature map SVM classifier is also optimized at the same time. An important part of this method is the grid search method which helps to optimize classifiers and decide the training sets. One of the notable effects of the grid search method is that it can prevent over-fitting.

The hierarchical component-based structure of this system provides a good platform to apply the attention mechanisms. It represents an object using a hierarchy by decomposing it into several prominent features. Advantage is taken of both the top-bottom and bottom-top knowledge of the object. A feature map acts as the medium to convey this hierarchy. Feature selection is the first task facing this structure. The selected features must be representative for the object to distinguish it from other objects and background. These features also should be distinct itself. The feature selection for the face classification experiment and thermal person detection experiment are manually conducted according to a human being's experience. Therefore they may not be the best choices for a computer. But currently they are good enough to justify the performance of this system.

The main benefit that the attention mechanism has brought is the time reduction. The introduction of the attention mechanism also has a secondary influences on the performance of this system. On the one hand, the attention mechanism leads this system to areas more likely to contain features. This reduces the chance of visiting areas which may contain false positive features. This means that the feature map is more likely to have features within the correct relative positions, which the second layer can correctly identify as an target object. When the fea-

ture classifiers are trained well, they will identify features and reject non-features within areas suggested by the attention mechanism, where a positive feature should be. But on the other hand, if there is imperfection on feature classifiers training, the attention mechanism can cause this system to make some mistakes. When detecting a non-object image, a feature classifier will be led to a possible area suggested by the attention mechanism. If the feature classifier is not well trained, it may take a non-feature as a feature. This false positive feature is in the position that can form a positive feature map. This increases the chance that a non-object image is taken as an target object. This negative influence can be offset to certain extent by the joint training dataset creation method introduced in 4.3. The detection performance of this system is influenced by the interaction of these two effects introduced by the attention mechanism. The less the imperfection of the feature classifiers is, the higher the performance of *Cengji* will be.

The role of the attention mechanism is to modify the search for features. The introduction of it has made this system search intelligently so it directly searches in the most likely areas containing features, ignoring irrelevant information. This is smarter than exhaustively searching everywhere when some knowledge of the target objects is available.

In this thesis, I have introduced and empirically evaluated the attention mechanism in *Cengji*. The dual-layer system with attention mechanism displayed good performance on both face and thermal person recognition. On a held-out face test set, the system with the attention mechanism scored 94% correct, compared with a score of 95% correct when not using the attention mechanism. On the thermal person test set, *Cengji* with the attention mechanism scored 86.8%. With the attention mechanism, this system performs well when there is knowledge about the target objects available. It can robustly find features though there may be false positive detections. The feature map concept works effectively, and provides an intuitive and extendable internal representation for compositional objects. The two-level SVMs demonstrated promising performance on structure classification. This provides a useful way of learning hierarchical structure of the real world.

Though the two-level SVM system is not perfect this time, it gives us an approach to understanding the real world scene, which can not be fulfilled by the single-level SVMs themselves. The benefits of using dual-layer SVM classifier include separately identified features, structural information of object, a more flexible architecture, and higher generality.

6.4 Future Directions

I have been spending lots of time pursuing a so called "Excellent" object recognition system. Obviously the current system has a long way to go to be considered "Excellent". But it is the starting step towards this target. During my limited PhD research period, it is inevitable there are many questions not answered. When the system was developed, there are many potential pathways, including other researchers' applications and some established algorithms, which have attracted my attention.

I will improve the dual-layer SVM classifiers in following directions: *Investigate the feature classifiers.* Currently, the feature classifiers perform well. But there are still some false detections especially false positive detections. These false positive detections give negative influence on the whole system performance. The improvement of the negative training set is one of the directions. Another direction is to investigate the classification algorithm itself, for example, its responses to those false positive detections.

Investigate the auto-associator. A feed forward neural network acts as the auto-associator for current system. It may not be the best option but only for verification purpose. Other algorithms will be applied to find the best option.

Generalise the attention mechanism. Some domain dependent information is used in the current system. This is due to the limitation of the auto-associator performance. I aim to make the application of the attention mechanism more general which can adapt to different domain automatically.

Analyse the impact of the negative data. Find images that are boundary cases. Key negative samples play an important role in the training of feature classifiers.

But it is difficult to find those boundary negative cases compared with positive cases. These boundary cases decide the classification performance of classifiers. Therefore it is useful to analyse the impact of the negative data.

Use different feature classifiers. Only SVM classifiers are used for the current system. In future, I will also use other algorithms.

Feature selection. Currently I select features which compose the object manually according to a human being's experience. We also arbitrarily specify the feature size, location and type. They work well for the current system but it may not be the best choice for a computer. There have been some applications applying certain algorithms to decide the features [55]. I will investigate these algorithms and find a suitable way for this system.

Appendix A

Appendix

A.1 Source Database for Face Classification Experiments

A face database is used in the face classification experiments. In order to focus on the comparison among different classification systems and simplify the training process, faces in this database are frontal, which means the face attitude extremity in all directions is no more than $\pm 30^o$. All face and non-face examples are downloaded from the Internet.

The Database of Faces from AT&T Laboratories, Cambridge This database is from AT&T Laboratories, Cambridge (formerly 'The ORL Database of Faces'). 40 frontal faces are used. See Figure A.1.

The Japanese Female Facial Expression (JAFFE) Database 66 frontal faces from this database are used. See Figure A.2.

The CalTech Database This is Collected by Markus Weber at California Institute of Technology. 158 frontal faces from this database are used. See Figure A.3.

The PIE Database This database is created by Robotics Institute, Carnegie Mellon University. PIE stands for the CMU Pose, Illumination, and Ex-

Figure A.1: *Some faces used from AT&T Laboratories, Cambridge*

Figure A.2: *Some faces used from The Japanese Female Facial Expression (JAFFE) Database*

Figure A.3: *Some faces used from The CalTech Database*

pression (PIE) database. 15 frontal faces from this database are used. See Figure A.4.

The Psychological Image Collection at Stirling (PICS) This database contains a number of sub-databases. See Figure A.5, A.6, A.7. 92 frontal faces from Stirling-faces database and 192 from Nottingham-scans and Nott-faces-originals are used.

Figure A.4: *Some faces used from The PIE Database*

Figure A.5: *Some faces used from Nottingham-scans, The Psychological Image Collection at Stirling (PICS)*

Figure A.6: *Some faces used from Stirling-faces, The Psychological Image Collection at Stirling (PICS)*

Figure A.7: *Some faces used from Nott-faces-originals, The Psychological Image Collection at Stirling (PICS)*

Figure A.8: *Some non-faces used*

The above databases were used for creating positive training samples. The following databases were used for creating negative samples.

BEV1 Dataset One of its sub-datasets, *Extra Background Images*, was used, which is about background.

Caltech Database Most samples are about natural scene. The *Background* sub-dataset was used.

Figure A.8 shows some non-faces.

Figure A.9: *Some faces used in the Harvard Face Database: note varying illumination*

The Harvard Face Database is another database used to test generalization of classifiers with different structures. There are 5 sub-directories containing 651 images. Each set has different illumination and face attitudes extremity. Set 1 has the smallest extremity, set 5 has the biggest. The image size in the database is 84×96. See Figure A.9.

Faces were manually extracted from these datasets and resized to the size of 84×96. 350 face images were chosen for training from above datasets and 150 were chosen for test. 350 non-face images were chosen for training and 150 for test.

A.2 Source Database for Person Detection Experiments

This database is provided by Digital Image Processing Lab at Cranfield University. They are taken by infrared cameras.

Figure A.10 shows some original images used for extracting training samples. They are captured from video shot at Cranfield University.

Figure A.11 shows some person image samples. They are extracted from full

Figure A.10: *Some original images from which the training samples are extracted. Person and non-person images will be extracted from these images and resized to 64×128.*

images which are illustrated in Figure A.10.

Figure A.12 shows some non-person image samples. They are extracted from full images which are illustrated in Figure A.10.

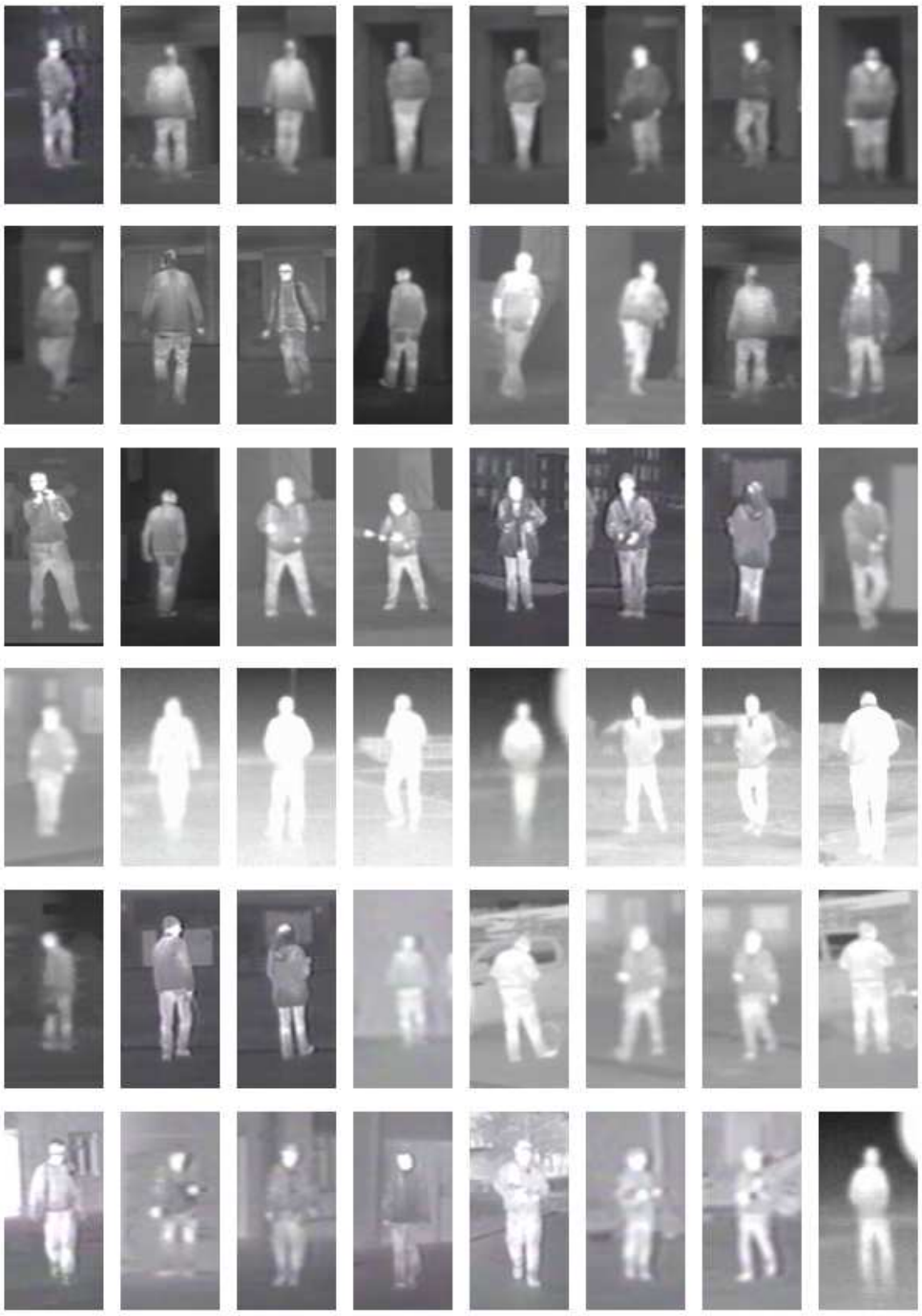


Figure A.11: *Some samples of people. They are used to create feature training datasets. Their sizes are 64×128.*

Figure A.12: *Some samples of non-people. They are used to create feature training datasets. Their sizes are 64×128.*

Appendix B

Appendix

B.1 Source Code of Feed-forward Neural Network

This part of source code is adapted from the Internet for the use of the experiments.

B.1.1 Header File

```
#pragma once
//////////////////////////////////////////////////////
typedef struct NeuroLayer
{
    INT nNode; //number of node
    INT nInput; //input node number
    DOUBLE *pInput; //input to a layer
    DOUBLE *pOutput; //output of a layer
    DOUBLE *pError; //layer error
    DOUBLE **ppWeight; //weights
    DOUBLE **ppDWeight;
} NeuroLayer;
```

```
//////////////////////////////////////////////////////
classCNeuroNet_BP
{
public:
    CNeuroNet_BP();
    CNeuroNet_BP(INT NumLayer, INT* pNumNodeInEachLayer, CString MyFile-
Name);
    ~CNeuroNet_BP(void);
    INT nInput; //feature number of an input sample
    DOUBLE *pInput; //network input
    DOUBLE *pOutput; //network output
    INT nLayer; //layer number of the network
    NeuroLayer *pLayer; //pointer to each layer
    DOUBLE Eta; //learning efficiency
    DOUBLE Alpha; //momentum
    DOUBLE Error; //network error
    DOUBLE Gain; //gain of the sigmoid function
    DOUBLE MError;
    DWORD TrainTimes; //training times
    CString fileName;
private:
public:
    BOOL ReadNetFromFile(void); // read neural network model from a file
    BOOL SaveNetToFile(void); // save neural network model to a file
public:
    // process input and generate output
    BOOL Propagate( DOUBLE* pInput);
    // node function
    DOUBLE func(INT i, INT j);
    void Train(DOUBLE* pInput, DOUBLE* pTarget);
    // randomly assign value to varibles
```

```
    void RandomWeight(void);
    void BackPropagate(void);
    // compute output errors
    void ComputeOutputError(DOUBLE* pTarget);
    // build the network frame and allocate memory space
    void GenerateNet(void);
};
```

B.1.2 Source File

```
#include"StdAfx.h"
#include"neuronet_bp.h"
#include"math.h"
CNeuroNet_BP::CNeuroNet_BP()
{
}
CNeuroNet_BP::CNeuroNet_BP(INT NumLayer, INT* pNumNodeInEachLayer,
CString MyFileName)
{
    Alpha = 0.9;
    Eta = 0.1;
    Gain = 1;
    Error = 0;
    nLayer = NumLayer;
    pLayer = new NeuroLayer[nLayer];
    this->TrainTimes =0;
    fileName = MyFileName;
    //build the neural network
    for(INT i=0; i<nLayer; i++)
    {
        pLayer[i].nNode = pNumNodeInEachLayer[i];
```

```
    }
    this->nInput = pLayer[0].nNode;
    //allocate memory for varibles
    this->GenerateNet();
    //randomly setup weight values
    this->RandomWeight();
}
CNeuroNet_BP::~CNeuroNet_BP(void)
{
}
// randomly setup weight values
voidCNeuroNet_BP::RandomWeight(void)
{
    for(INT i=1; i<nLayer; i++)
    {
        for(INT j=0; j<pLayer[i].nNode; j++)
        {
            for(INT k=0; k<pLayer[i].nInput; k++)
            {
                pLayer[i].ppWeight[j][k] = (double(rand())-0.5*
                double(RAND_MAX))/double(RAND_MAX);
                //weight values are between (-0.5, 0.5)
                pLayer[i].ppDWeight[j][k] = 0.0;
            }
        }
    }
}
// read weights values from model file
BOOL CNeuroNet_BP::ReadNetFromFile(void)
{
    INT i;
```

```
CFile file;
if(file.Open(fileName, CFile::modeRead)==FALSE)
{
    returnFALSE;
}
char *pChar = new char[10];
file.Read((void*)pChar, 10);
file.Read((void*)&this->TrainTimes, sizeof(DWORD));
file.Read((void*)&this->Error, sizeof(DOUBLE));
file.Read((void*)&this->Alpha, sizeof(DOUBLE));
file.Read((void*)&this->Eta, sizeof(DOUBLE));
file.Read((void*)&this->Gain, sizeof(DOUBLE));
file.Read((void*)&this->nInput, sizeof(INT));
file.Read((void*)&this->nLayer, sizeof(INT));
//read out node numbers of each layer
INT* pNumNode = new INT[nLayer];
for(i=0; i<nLayer; i++)
{
    file.Read((void*)&pNumNode[i], sizeof(INT));
}
pLayer = new NeuroLayer[nLayer];
for(i=0; i<nLayer; i++)
{
    pLayer[i].nNode = pNumNode[i];
}
this->GenerateNet();
//assign weights to each neuron
for(i=1; i<nLayer; i++)
{
    for(INT j=0; j<pLayer[i].nNode; j++)
    {
```

```
                file.Read(pLayer[i].ppWeight[j],      pLayer[i].nInput*sizeof(DOUBLE));
//read out weights of each layer
                file.Read(pLayer[i].ppDWeight[j], pLayer[i].nInput*sizeof(DOUBLE));
            }
        }
        returnTRUE;
}
// save weights to file
BOOL CNeuroNet_BP::SaveNetToFile(void)
{
        INT nTemp = 0;
        INT i;
        char ID[10] = "NeuroBP";
        CFile file(fileName,CFile::modeCreate|CFile::modeWrite);
        file.Write((void*)ID, 10); //write file type tag
        file.Write((void*)&TrainTimes, sizeof(DWORD)); //write training times
        file.Write((void*)&this->Error, sizeof(DOUBLE)); //write the latest network er-
ror
        file.Write((void*)&this->Alpha, sizeof(DOUBLE)); //write training parameters
        file.Write((void*)&this->Eta, sizeof(DOUBLE));
        file.Write((void*)&this->Gain, sizeof(DOUBLE));
        file.Write(&(this->nInput), sizeof(INT));
        file.Write(&(this->nLayer), sizeof(INT));
        //write node numbers of each layer
        for(i=0; i<nLayer; i++)
        {
            file.Write(&(pLayer[i].nNode), sizeof(INT));
        }
        //write weights of each node within each layer
        for(i=1; i<nLayer; i++)
        {
```

```
            for(INT j=0; j<pLayer[i].nNode; j++)
            {
                file.Write(pLayer[i].ppWeight[j], pLayer[i].nInput*sizeof(DOUBLE));
                file.Write(pLayer[i].ppDWeight[j], pLayer[i].nInput*sizeof(DOUBLE));
            }
        }
        file.Close();
        returnTRUE;
}
// process input and generate output
BOOL CNeuroNet_BP::Propagate( DOUBLE* pInput)
{
        this->pInput = pInput;
        //calculate output
        for(INT i=0; i<this->nLayer; i++)
        {
            if(i==0)
            {
                pLayer[i].pInput = this->pInput;
                for(INT j=0; j<pLayer[i].nNode-1; j++)
                {
                    pLayer[i].pOutput[j] = 1.0/(1.0 + exp(-pLayer[i].pInput[j]*
                    this->Gain));
                }
            }
            else
            {
                pLayer[i].pInput = pLayer[i-1].pOutput;
                intm_onumber;
                m_onumber=pLayer[i].nNode;
                for(INT j=0; j<m_onumber; j++)
```

```
                {
                    pLayer[i].pOutput[j] = func(i, j);
                }
            }
        }
        returnTRUE;
}
// calculate output of each neuron
DOUBLE CNeuroNet_BP::func(INT i, INT j)
{
        DOUBLE sigma = 0.0;
        for(INT k=0; k<pLayer[i].nInput; k++)
        {
            sigma = sigma + pLayer[i].pInput[k] * pLayer[i].ppWeight[j][k];
        }
        sigma = 1.0/(1.0 + exp(-sigma*this->Gain));
        return sigma;
}
//train the neural network with samples
voidCNeuroNet_BP::Train(DOUBLE* pInput, DOUBLE* pTeach)
{
        Propagate(pInput); //process input data
        ComputeOutputError(pTeach); //calculate errors
        BackPropagate(); //backpropagate and adjust weights
        this->TrainTimes++;
}
//********************************************************************************
// backpropagate and adjust weights
// ********************************************************************************
// calculate output errors
voidCNeuroNet_BP::ComputeOutputError(DOUBLE* pTarget)
```

```
{
    DOUBLE Out, Err;
    this->Error=0;
    this->MError=0;
    for(INT i=0; i<pLayer[nLayer-1].nNode; i++)
    {
        Out = pLayer[nLayer-1].pOutput[i];
        Err = pTarget[i] - Out;
        this->pLayer[nLayer-1].pError[i] = Out*(1-Out)*Err;
        this->Error += 0.5*pow(Err, 2);
        if(abs(Err)<0.5)
        Err=0;
        else
        Err=1;
        this->MError=MError+Err;
    }
}
//backpropagate errors
void CNeuroNet_BP::BackPropagate(void)
{
    DOUBLE Out, Err;
    INT i, j, k;
    //calculate backwards errors
    for(i=nLayer-2; i>=0; i–)
    {
        if(i!=0)
        {
            for(j=0; j<pLayer[i].nNode; j++)
            {
                Out = pLayer[i].pOutput[j];
                Err = 0;
```

```
                for(INT k=0; k<pLayer[i+1].nNode; k++)
                {
                    Err += pLayer[i+1].pError[k] * pLayer[i+1].ppWeight[k][j];
                }
                pLayer[i].pError[j] = Out * (1-Out) * Err;
            }
        }
        else
        {
            for(j=0; j<pLayer[i].nNode; j++)
            {
                Out = pLayer[i].pOutput[j];
                Err = 0;
                for(k=0; k<pLayer[i+1].nNode-1; k++)
                {
                    Err += pLayer[i+1].pError[k] * pLayer[i+1].ppWeight[k][j];
                }
                pLayer[i].pError[j] = Out * (1-Out) * Err;
            }
        }
    }
    //adjust weights
    for(i=nLayer-1; i>0; i–)
    {
        if(i==nLayer-1)
        {
            for(j=0; j<pLayer[i].nNode; j++)
            {
                for(k=0; k<pLayer[i].nInput; k++)
                {
                    Out = pLayer[i-1].pInput[k];
```

```
                    Err = pLayer[i].pError[j];
                    pLayer[i].ppWeight[j][k] += pLayer[i].ppDWeight[j][k];
                    pLayer[i].ppDWeight[j][k] = this->Eta *Err * Out +
                    this->Alpha*pLayer[i].ppDWeight[j][k];
                }
            }
        }
        else
        {
            for(j=0; j<pLayer[i].nNode; j++)
            {
                for(k=0; k<pLayer[i].nInput; k++)
                {
                    Out = pLayer[i-1].pInput[k];
                    Err = pLayer[i].pError[j];
                    pLayer[i].ppWeight[j][k] += pLayer[i].ppDWeight[j][k];
                    pLayer[i].ppDWeight[j][k] = this->Eta *Err * Out +
                    this->Alpha*pLayer[i].ppDWeight[j][k];
                }
            }
        }
    }
}
// build the network and allocate memory for varibles
voidCNeuroNet_BP::GenerateNet(void)
{
    for(INT i=0; i<nLayer; i++)
    {
        if(i==0)
        {
            pLayer[i].nInput = this->nInput;
```

```
            pLayer[i].pOutput = new DOUBLE[pLayer[i].nNode]; //allocate mem-
ory for output
            pLayer[i].pError = new DOUBLE[pLayer[i].nNode];
        }
        else
        {
            pLayer[i].nInput = pLayer[i-1].nNode;
            pLayer[i].pOutput = new DOUBLE[pLayer[i].nNode];
            pLayer[i].pError = new DOUBLE[pLayer[i].nNode]; //allocate momery
for neuron errors
            pLayer[i].ppWeight = new DOUBLE*[pLayer[i].nNode ]; //allocate
memory for weight pointers
            pLayer[i].ppDWeight = new DOUBLE*[pLayer[i].nNode ]; //allocate
memory for weight increments
            for(INT j=0; j<pLayer[i].nNode; j++)
            {
                pLayer[i].ppWeight[j] = new DOUBLE[pLayer[i].nInput]; //allocate
memory for weights of each neuron
                pLayer[i].ppDWeight[j] = new DOUBLE[pLayer[i].nInput]; //allo-
cate memory for weight increments
            }
        }
    }
    this->pOutput = pLayer[nLayer-1].pOutput;
}
```

B.2 Source Code of Locating Features in a Proposed Feature Map

OpenCV image processing library is used in this part of source code.

```
CvPoint* CCellNuclearDetectorView::LocateFeatures(IplImage* m_pImage)
{
    //locates features in a proposed face feature map, returns their coordinates
    //takes proposed feature map as input
    IplImage* img;
    CvPoint* points = new CvPoint [3];
    //makes sure the input image is a 256 gray scale image
    if(m_pImage->nChannels == 1)
    {
        img= (IplImage*)cvClone(m_pImage);
    }
    else
    {
        img = cvCreateImage( cvGetSize(m_pImage), 8, 1 );
        cvCvtColor(m_pImage, img, CV_BGR2GRAY);
    }
    cvSmooth( img, img, CV_MEDIAN, 3, 3, 0, 0 );
    //thresholds the image into a binary image
    cvThreshold(img, img, 30, 255, CV_THRESH_BINARY_INV);
    //morphological operations to form pixel blocks
    cvErode( img, img, NULL, 3);
    cvDilate( img, img, NULL, 3);
    //locates the centres of pixel blocks
    CvMemStorage* storage = cvCreateMemStorage(0);
    CvSeq* contours = 0;
    CvSeq tmp;
    CvMat* color_tab;
    int comp_count = 0;
//finds feature patches
    cvFindContours( img, storage, &contours, sizeof(CvContour),
CV_RETR_CCOMP, CV_CHAIN_APPROX_SIMPLE );
//finds pathch corner coordinates
    int left, right, top, bottom;
```

```
    int i=0;
    for( ; contours != 0; contours = contours->h_next, comp_count++ )
    {
        if((contours->first->count)>4 && i<3) //a patch must have more than 4
pixels
        {
            left=10000;
            right=0;
            top=0;
            bottom=10000;
            for(i=0; i<contours->first->count; i++) //finds the centre of the small-
est rectangle surrounding the patch
            {
                CvPoint* k;
                k=(CvPoint*)cvGetSeqElem(contours,
                    contours->first->start_index+i);
                if(k->x<left)
                    left=k->x;
                if(k->x>right)
                    right=k->x;
                if(k->y<bottom)
                    bottom=k->y;
                if(k->y>top)
                    top=k->y;
            }
            points[i].x=(left+right)/2;
            points[i].y=(top+bottom)/2;
        }
        i=i+1;
    }
    return points;
}
```

Bibliography

[1] T. Acharya and A. K. Ray. *Image processing: principles and applications*. John Wiley & Sons, Inc., New Jersey, USA, 1st edition, 2005.

[2] M. Ahissar and S. Hochstein. The reverse hierarchy theory of visual perceptual learning. *Trends in Cognitive Sciences*, 8(10):457–464, October 2004.

[3] V. Ahl and T. F. H. Allen. *Hierarchy Theory*. Columbia University Press, 136 S. Broadway, Irvington, New York, 1996.

[4] Subutai Ahmad. *Visit: an efficient computational model of human visual attention*. PhD thesis, University of Illinois at Urbana-Champaign, Champaign, IL, USA, 1992.

[5] J. C. Alonso-Bayal, R. Santiago-Mozos, J. M. Leiva-Murillo, M. Lazaro, and A. Artes-Rodriguez. Real-time tracking and identification on an intelligent ir-based surveillance system. In *IEEE Conference on Advanced Video and Signal Based Surveillance, 2007*, pages 277 – 282, 2007.

[6] E. ALPAYDIN. *Introduction to Machine Learning*. The MIT Press, Cambridge, Massachusetts, London, England, 2004.

[7] M. A. Arbib. *The Handbook of Brain Theory and Neural Networks*. The MIT Press, Cambridge, USA, 2002.

[8] P. Artal, A. Benito, and J. Tabernero. The human eye is an example of robust optical design. *Journal of Vision*, 6:1–7, 2006.

[9] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice Hall, Englewood Cliffs, New Jersey, USA, 1982.

[10] S. Behnke and R. Rojas. Neural abstraction pyramid: a hierarchical image understandingarchitecture. In *The 1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence.*, volume 2, pages 820–825. 1998.

[11] I. Biederman. On the semantics of a glance at a scene. In M. Kobovy and J. R. Pomerantz, editors, *Perceptual Organization*, pages 213–254. Hillsdale, NJ: Lawrence Erlbaum, 1981.

[12] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.

[13] Stanley Michael Bileschi. *StreetScenes: Towards Scene Understanding in Still Images*. PhD thesis, Massachusetts Institute of Technology, Cambridge, USA, 2006.

[14] A. L. Blum and P. Langley. Selection of relevent features and examples in machine learning. *Artificial Intelligence*, 97:245–271, 1997.

[15] I. Bosch, S. Gomez, L. Vergara, and J. Moragues. Infrared image processing and its application to forest fire surveillance. In *IEEE Conference on Advanced Video and Signal Based Surveillance, 2007*, pages 283–288, 2007.

[16] N. K. Bose and P. Liang. *Neural Network Fundamentals with Graphs, Algorithms, and Applications*. McGraw-Hill, Inc., New York, USA, 1996.

[17] T. P. Breckon, S. E. Barnes, M. L. Eichner, and K. Wahren. Autonomous real-time vehicle detection from a medium-level uav. In *Proc. 24th International Unmanned Air Vehicle Systems*, pages 29.1–29.9, March 2009.

[18] T. P. Breckon, M. L. Eichner, and S. E. Barnes. Autonomous threat detection in urban environments from mobile robotic platforms. In *British*

Machine Vision Association Symposium on Vision and Robotics. British Computer Society, London, UK, 2008.

[19] R. Brunelli. *Template Matching Techniques in Computer Vision: Theory and Practice*. John Wiley & Sons, Ltd, NJ, USA, 2009.

[20] C. J. C. Burges. A tutorial on support vector machines for pattern. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[21] P. J. Burt. Attention mechanisms for vision in a dynamic world. In *9th International Conference on Pattern Recognition, 1988.*, volume 2, pages 977–987, 1988.

[22] T. Caelli and W. F. Bischof. Cite-scene understanding and object recognition. *Machine Learning and Image Interpretation*, pages 119–187, 1997.

[23] V. Cantoni and L. Lombardi. Hierarchical architectures for computer vision. In *Euromicro Workshop on Parallel and Distributed Processing*, pages 392–398. San Remo, Italy, 1995.

[24] C. C. Chang and C. J. Lin. Libsvm – a library for support vector machines. http://www.csie.ntu.edu.tw/ cjlin/libsvm/.

[25] M. Chester. *Neural Networks A Tutorial*. PTR Prentice Hall, New Jersey, USA, 1993.

[26] N. Cristianini and J. Shawe-Tayor. *An Introduction to Support Vector Machines*. The Press Syndicate of The University of Cambridge, 2000.

[27] N. Davey and R. Adams. High capacity associative memories and connection constraints. *Connection Science*, pages 177–182, 2004.

[28] C. Dillon and T. Caelli. Learning image annotation: the CITE system. *Journal of Computer Vision Research*, 1:89–122, 1998.

[29] R. O. Duda, P. E. Hart, and D. G. Stock. *Pattern Classification*. Wiley-Interscience, New York, USA, 2001.

[30] A. E.Bryson and Y. C. Ho. *Applied optimal control: optimization, estimation, and control*. Waltham, MA: Blaisdell, 1969.

[31] M. L. Eichner and T.P. Breckon. Augmenting gps speed limit monitoring with road side visual information. In *Proc. IET/ITS Conference on Road Transport Information and Control*, pages 1–5. IEEE, May 2008.

[32] I. El-Naqa, Y. Y. Yang, M. N. Wernick, N. P. Galatsanos, and R. Nishikawa. Support vector machine learning for detection of microcalcifications in mammograms. In *IEEE Transactions on Medical Imaging*, volume 21, pages 1552–1563. 2002.

[33] A. Elgammal and L. S. Davis. Probabilistic framework for segmenting people under occlusion. In *Proceeding of IEEE 8th International Conference on Computer Vision*, pages 145–152, 2001.

[34] M. W. Eysenck and M. T. Keane. *Cognitive Psychology: A Student's Handbook*. Psychology Press Ltd, 27 Church Road, Hove, East Sussex BN3 2FA, UK, 2005.

[35] M. J. Farah. *The cognitive neuroscience of vision*. Blackwell Publishers Inc., USA, 1st edition, 2000.

[36] T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

[37] W. A. Fellenz and G. Hartmann. Preattentive grouping and attentive selection for early visual computation. In *Proceedings of International Conference on Pattern Recognition*, pages 340–345, 1996.

[38] L. Ferrarini, I. M. Veer, E. Baerends, v. T. Marie-José, R. J. Renken, V. D. W. Nic J. A., Dirk, A. Aleman, F. G. Zitman, B. W. J. H. Penninx, V. B. Mark A., J. H. C. Reiber, S. A. R. B. Rombouts, and J. Milles. Hierarchical functional modularity in the resting-state human brain. *Human Brain Mapping*, 30(7):2220–31, 2008.

[39] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2002.

[40] F. Gobet, P. C. R. Lane, S. Croker, P. C-H. Cheng, G. Jones, I. Oliver, and J. M. Pine. Chunking mechanisms in human learning. *Trends in Cognitive Science*, 5:236–243, 2001.

[41] R. C. Gonzales and R. E. Woods. *Digital image processing*. Pearson Education Inc., New Jersey, USA, 3rd edition, 2008.

[42] K. Gurney. *Introduction to Neural Networks*. CRC Press, London, 3rd edition, 2003.

[43] J. W. Han, T. P. Breckon, D. A. Randell, and G. Landini. Radicular cysts and odontogenic keratocysts epithelia classification using cascaded haar classifiers. In *Proceeding of the 12th Annual Conference on Medical Image Understanding and Analysis Conference (MIUA)*, 2008.

[44] J. W. Han, P. C. R. Lane, N. Davey, and Y. Sun. Comparing the performance of single-layer and two-layer support vector machines on face detection. In *Proceedings of The Seventh UK Workshop on Computational Intelligence*, 2007.

[45] J. W. Han, P. C. R. Lane, N. Davey, and Y. Sun. A dual-layer model of high-level perception. In R.M. French and E. Thomas, editors, *From Associations to Rules: Connectionist Models of Behavior and Cognition, Proceedings of the Tenth Neural Computation and Psychology Workshop, (World Scientific) Progress in Neural Processing*, volume 17, pages 139–149, 2007.

[46] J. W. Han, P. C. R. Lane, N. Davey, and Y. Sun. Attention mechanisms and component-based face detection. In *Proceedings of International Conference on Methods and Models in Computer Science, ICM2CS09*, pages 1–6. IEEE, 2009.

[47] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, pages 147–151, Manchester, UK, 1988.

[48] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, New York, USA, 2001.

[49] J. He, X. B. Dai, and X. C. Zhao. Associative artificial neural network for discovery of highly correlated gene groups based on gene ontology and gene expression. In *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 17–24, Honolulu, Hawaii, USA, 2007.

[50] M. A. Hearst. Support vector machines. *IEEE Intelligent Systems*, 13:18–28, 1998.

[51] B. Heisele, P. Ho, J. Wu, and T. Poggio. Face recognition: Component-based versus global approaches. *Computer Vision and Image Understanding*, 91:6–21, 2003.

[52] B. Heisele, T. Serre, M. Pontil, and T. Poggio. Component-based face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 657–662, 2001.

[53] B. Heisele, T. Serre, M. Pontil, T. Vetter, and T. Poggio. Categorization by learning and combining object parts. In *Advances in Neural Information Processing Systems*, volume 2, pages 1239–1245. 2001.

[54] C. W. Hsu, C. C. Chang, and C. J. Lin. A practical guide to support vector classification. citeseer.ist.psu.edu/689242.html.

[55] J. Huang, V. Blanz, and B. Heisele. Face recognition using component-based SVM classification and morphable models. In *SVM 2002*, pages 334–341, 2002.

[56] A. K. Jain, P. W. Duin, and J. Mao. Statistical pattern recognition : A review. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 22, pages 4–37, 2000.

[57] T. Joachims. Making large-scale svm learning practical. *Advances in Kernel Methods - Support Vector Learning*, 1998.

[58] S. Kastner and L. G. Ungerleider. Mechanisms of visual attention in the human cortex. *Annual Review of Neuroscience*, 23:315–341, 2000.

[59] S. Kastner, P. D. Weed, R. Desimone, and L. G. Ungerleider. Mechanisms of directed attention in the human extrastriate cortex as revealed by functional mri. *Science*, 282:108–111, 1998.

[60] G. Kerschen1 and J. Golinval. Feature extraction using auto-associative neural networks. *Smart Materials and Structures*, 13:211C219, 2004.

[61] Yves Kodratoff and Stephane Moscatelli. Machine learning for object recognition and scene analysis. *IJPRAI*, pages 259–304, 1994.

[62] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, page 1137C1143, 1995.

[63] M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37:233–243, 1991.

[64] K. Krawied, D. Howard, and M. Zhang. Overview of object detection and image analysis by means of genetic programming techniques. *Frontiers in the Convergence of Bioscience and Information Technologies*, pages 779–784, 2007.

[65] G. Landini, D. A. Randell, T. P. Breckon, and J. W. Han. Morphological characterisation of cell neighbourhoods in neoplastic and pre-neoplastic epithelium. *International Journal of Analytical and Quantitative Cytology and Histology*, in press:in press, 2009.

[66] P. C. R. Lane, F. Gobet, and R. Ll. Smith. Attention mechanisms in the chrest cognitive architecture. In *Attention in Cognitive Systems*, volume 5395, pages 183–196. Springer Berlin / Heidelberg, 2009.

[67] P. C. R. Lane, A. K. Sykes, and F. Gobet. Combining low-level perception with expectations in CHREST. In F. Schmalhofer, R. M. Young, and G. Katz, editors, *Proceedings of EuroCogsci*, pages 205–210. Mahwah, NJ: Lawrence Erlbaum Associates, 2003.

[68] P. Langley. Crafting papers on machine learning. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1207–1216. Morgan Kaufmann Publishers Inc., 2000.

[69] Y. Lerner, T. Hendler, D. Ben-Bashat, M. Harel, and R. Malach. A hierarchical axis of object processing stages in the human viusal cortex. *Cerebral cortex*, 11:287–297, 2001.

[70] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *2002 International Conference on Image Processing. 2002. Proceedings.*, volume 1, pages 900–903, 2002.

[71] Z. L. Lu, C. Q. Liu, and B. A. Dosher. Attention mechanisms for multi-location first- and second-order motion perception. *Vision Research*, 40:173–186, 2000.

[72] F. Marini, A. L. Magra, and R. Buccia. Multilayer feed-forward artificial neural networks for class modeling. *Chemometrics and Intelligent Laboratory Systems*, 88:118–124, 2007.

[73] D. Marr. *Vision: A computational investigation into the human representation and processing of visual information*. San Francisco: W. H. Freeman, 1982.

[74] P. Mcleod, K. Plunkett, and E. T. Rolls. *Introduction to Connectionist Modelling of Cognitive Processes*. Oxford University Press Inc., New York, 1998.

[75] P. Meer, C. V. Stewart, and D. E. Tyler. Robust computer vision: An interdisciplinary challenge. *Computer Vision and Image Understanding*, 78:1–7, 2000.

[76] R. Milanese1y, H. Wechsler, S. Gill, J. Bostl, and T. Pun. Integration of bottom-up and top-down cues for visual attentionusing non-linear relaxation. In *1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94.*, pages 781–785. 1994.

[77] T. M. Mitchell. *Machine Learning*. The McGraw-Hill Companies, Inc., 1997.

[78] E. Mjolsness and D. DeDoste. Machine learning for science: State of the art and future prospects. *Science*, 293:2051–2055, 2001.

[79] F. Mosteller. A k-sample slippage test for an extreme population. In *The Annals of Mathematical Statistics*, volume 19, pages 58–65. Institute of Mathematical Statistics, 1948.

[80] T. W. Nattkemper, H. J. Ritter, and W. Schubert. A neural classifier enabling high-throughput topological analysis of lymphocytes in tissue sections. *IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE*, 5:138–149, 2001.

[81] U. Neisser. *Cognitive Psychology*. New York: Appleton-Century-Crofts, 1966.

[82] S. Nicole. Feedforward neural networks for principal components extraction. *Computational Statistics & Data Analysis*, 33:425–437, 2000.

[83] E. Niebur and C. Koch. Control of selective visual attention: Modelling the "where" pathway. In *Advances in Neural Information Processing Systems*, volume 8, pages 802–808. The MIT Press, 1996.

[84] C. P. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. *iccv*, 00:555–562, 1998.

[85] H. H. Pattee. *Hierarchy Theory: The Challenge of Complex Systems*. NY: Georges Braziller, New York, 1973.

[86] P. D. Picton. *Introduction to Neural Networks*. The Macmillan Press Ltd., London, UK, 1994.

[87] M. A. Pinsk, G. M. Doniger, and S. Kastner. Push-pull mechanism of selective attention in human extrastriate cortex. *Journal of Neurophysiology*, 92:622–629, 2004.

[88] A. R. Pope and D. G. Lowe. Probabilistic models of appearance for 3-d object recognition. *International Journal of Computer Vision*, 40:149–167, 2000.

[89] D. Purves and R. B. Lotto. *Why We See What We Do: An Empirical Theory of Vision*. Sinauer Associates, Sunderland, Massachusetts, USA, 2003.

[90] R. A. Rensink and J. T. Enns. Early completion of occluded objects. *Vision Research*, 38:2489–2505, 1998.

[91] H. B. Richman and H. A. Simon. Context effects in letter perception: Comparison of two theories. *Psychological Review*, 3:417–432, 1989.

[92] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2:1019–1025, 1999.

[93] M. Riesenhuber and T. Poggio. Computational models of object recognition in cortex: A review. Technical report, Artificial Intelligence Laboratory and Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, 2000.

[94] M. Robinson, C. G. Castellano, R. Adams, N. Davey, and Y. Sun. Identifying binding sites in sequential genomic data. In *Artificial Neural Networks C ICANN 2007*, pages 100–109, Porto, Portugal, 2007. Springer.

[95] J. C. Russ. *The Image Processing Handbook*. CRC Press, Inc. Boca Raton, Florida, USA, 1995.

[96] S. J. Russell and P. Norvig. *Artificial Intelligence: Modern Approach*. Prentice Hall, 1995.

[97] B. Schölkopf and A. J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, Massachusetts, London, England, 2002.

[98] L. G. Shapiro and G. C. Stockman. *Computer Vision*. Prentice Hall, Inc, Upper Saddle River, New Jersey, 2001.

[99] M. Shear. *Cysts of the oral regions*. Wright, Oxford, 3rd edition, 1992.

[100] H. A. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 106(6):467–482, 1962.

[101] H. A. Simon. *The sciences of the artificial*. Cambridge (Mass.) ; London : M.I.T. Press, 1969.

[102] K. Sung. *Learning and Example Selection for Object and Pattern Detection*. PhD thesis, MIT AI Lab and Center for Biological and Computational Learning, 1995.

[103] X. Y. Tan, S. C. Chen, Z. H. Zhou, and F. Y. Zhang. Recognizing partially occluded, expression variant faces from single training image per person with som and soft k-nn ensemble. In *IEEE TRANSACTIONS ON NEURAL NETWORKS*, pages 875–886, 2005.

[104] A. Torralba. Contextual priming for object detection. *International Journal of Computer Vision*, 53:169–191, 2003.

[105] A. Torralba, A. Oliva, M. S. Castelhano, and J. M. Henderson. Contextual guidance of eye movements and attention in real-world scenes: The role of global features on object search. *Psychological Review*, 114(4):766–786, 2006.

[106] J. K. Tsotsos, S. M. Culhane, W. Y. K. Winky, Y. Z. Lai, N. Davis, and F. Nuflo. Modeling visual attention via selective tuning. *Artificial Intelligence*, 78(1-2):507–545, 1995.

[107] K. Veropoulos, N. Cristianini, and C. Campbell. Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence, (IJCAI99)*, Stockholm, Sweden, 1999.

[108] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, page 511, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

[109] P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57:137–154, 2004.

[110] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrains using patterns of motion and appearance. *International Journal of Computer Vision*, 63:153–161, 2005.

[111] L. R. Williams. Perceptual completion of occluded surfaces. *COMPUTER VISION AND IMAGE UNDERSTANDING*, 64:1–20, 1996.

[112] J. M. Wolfe. Guided search 2.0: A revised model of visual search. *Psychonomic Bulletin and Review*, 1:202–238, 1994.

[113] M. H. Yang, D. J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 24, pages 34–58. 2002.

[114] Z. Y. Zhang, R. Deriche, O. D. Faugeras D., and Q. T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1-2):87–119, 1995.

[115] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35:399–458, 2003.