Automatic Detection of Image Orientation using Support Vector Machines

Dane A. Walsh

THESIS PRESENTED
IN FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE
AT THE UNIVERSITY OF STELLENBOSCH

Supervisor: Prof. Christian W. Omlin

December, 2002

Stellenbosch University http://scholar.sun.ac.za

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and that I have not previously in its entirety or in part submitted it at any university for a degree.

Stellenbosch University http://scholar.sun.ac.za

Abstract

In this thesis, we present a technique for the automatic detection of image orientation using Support Vector Machines (SVMs). SVMs are able to handle feature spaces of high dimension and automatically choose the most discriminative features for classification. We investigate the use of various kernels, including heavy tailed RBF kernels. We compare the classification performance of SVMs with the performance of multilayer perceptrons and a Bayesian classifier. Our results show that SVMs out perform both of these methods in the classification of individual images. We also implement an application for the classification of film rolls in a photographic workflow environment with 100% classification accuracy.

Stellenbosch University http://scholar.sun.ac.za

Opsomming

In hierdie tesis, gebruik ons 'n tegniek vir die automatiese klassifisering van beeldoriëntasie deur middel van Support Vector Machines (SVM's). SVM's kan kenmerkruimtes van 'n hoë dimensie hanteer en kan automaties die mees belangrike kenmerke vir klassifikasie kies. Ons vors die gebruik van verskeie kerne, insluitende RBF-kerne, na. Ons vergelyk die klassifiseringsresultate van SVM's met die van multilaagperseptrone en 'n Bayes-klassifiseerder. Ons bewys dat SVM's beter resultate gee as beide van hierdie metodes vir die klassifikasie van individuele beelde. Ons implementeer ook a toepassing vir die klassifisering van rolle film in a fotografiese werkvloei-omgewing met 100% klassifikasie akuraatheid.

Stellenbosch University http://scholar.sun.ac.za

Acknowledgements

I would like to thank my supervisor, Prof. C.W. Omlin, for his assistance, persistance, inspiration and funding. I would also like to thank Reg Dodds for his assistance.

Stellenbosch University http://scholar.sun.ac.za

Contents

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

List of Tables

Stellenbosch University http://scholar.sun.ac.za

List of Figures

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Chapter 1

Introduction

1.1 Motivation

The Internet allows people to share photographic libraries across great distances through the use of webpages and email. In order to create these digital photographic libraries, laboratories are offering to convert traditional film negatives into digital images and ship them back to the customer either via the Internet, or on a physical storage device. In traditional film development, image orientation plays no role as a photograph can easily be viewed in the correct orientation. This is not the case with digital images; third party software is neccessary to correctly orientate them.

Today, the only way for laboratories to correct the orientation of images is through the manual manipulation of the images. An operator must look at each image, determine and correct image orientation. This neccessary human intervention considerably slows down the production line; modern negative scanners are capable of scanning an image in half a second. As a result, the operator only adjusts images that are obviously incorrectly orientated in order to keep up with the rate of the scanner. It would be very desirable to design a tool that could fit into the workflow to correctly determine the orientation of the images before they are shipped to the customer.

With the introduction of the digital camera, individuals now have the ability to create their own digital image collections. As these collections grow in size, so too does the need for an image management system that can assist the individual in viewing, manipulating, and organising these images in the database. Currently, incorrectly orientated images need to be manually rotated using software that provides such functionality for the correct viewing of images. Such an operation on a large collection of images can be time consuming; a tool is thus needed that can automate this process. Such a tool must be able to distinguish whether an image is in the landscape or portrait orientation and orientate

Stellenbosch University http://scholar.sun.ac.za

it accordingly. Since most digital cameras already come with accompanying graphics manipulation software, supplying the user with feature-rich software improves the quality of service for the user, and increases the product appeal for the digital camera vendor.

1.2 Problem Statement

As humans, we are able to make use of semantic objects such as trees, buildings, mountains, etc. and their relationships to provide us with information about an image. With this information, we are able to make various observations about an image, such as the focal point, the image orientation, the perspective, etc. Maintaining these relationships while at the same time extracting information from an image is a difficult task, and extremely CPU intensive. This makes object recognition as a basis for image orientation detection infeasible. For this reason, we have to rely on low-level visual features such as colour distribution, textures and contrasts as a means of orientation detection. The large number of such features may make many traditional classifiers inappropriate tools for orientation detection.

Our problem is not a simple one, as there are many factors that contribute to the complexity and difficulty of trying to determine the orientation of an image in a real time system. The first problem to overcome is the large diversity of content that can be present in images. Unlike other applications that operate on a specific subset of images, such as identifying particular objects within images, image content is virtually limitless. In order to develop a system that is able to correctly orientate such images with high accuracy rates, two factors must be considered: The first is the number of images available for use in the training set. The more images there are, the better the system will be able to classify unseen images. The second is the actual generalisation ability of the classifier used. Different classifiers have different generalisation capabilities, and the ideal classifier would be one that maximises generalisation to unseen images while minimising classification time. This brings us to the next problem: the constraint of time. In order to keep up with the speed of modern negative scanners orientation detection and correction must be performed quickly, i.e. in less than one second per image. This hard time constraint prohibits lengthy processing as is required for content-based image analysis. Other factors contributing to the complexity of the problem include finding optimal values for some of the many parameters used and to optimise the set of features to extract from the image.

Stellenbosch University http://scholar.sun.ac.za

1.3 Objectives

The main objective of this thesis is to show that Support Vector Machines (SVMs) can be used for the detection of image orientation from low-level features and that their performance is superior to that of other machine learning techniques, e.g. neural networks and Bayesian learning. We show that SVMs can be used to correctly identify the orientation of images with fast classification times, thus making them ideal for real-time applications where the classification time is critical in the performance of such a system. We also demonstrate that SVMs can perform well, with good classification accuracy, on a wide variety of images. SVMs are able to handle high dimensional feature spaces and can automatically choose the most discriminative features through training; this will provide us with a degree of flexibility when selecting the features from the images.

A current disadvantage of SVMs is that they produce a real-valued output, and not a probabilistic output. Probabilistic outputs are particularly useful when trying to determine the degree of confidence in the correct classification of objects. Currently, SVMs only classify objects based on their distances from the decision surface, and do not produce probabilistic outputs; some work has been done on the conversion of the output of a SVM to a probabilistic value. We incorporate these conversions into our system, and investigate whether performance is improved.

A useful characteristic of SVMs is their ability to make use of different kernels. The advantage is that some kernels perform better on certain problems than others, and so the performance of a SVM can be improved by selecting the most appropriate kernel. We investigate traditional kernels associated with pattern recognition, as well as several newer kernels, and determine the optimal values for some of the parameters involved.

As a final test, we demonstrate how the above objectives can be achieved by implementing a system that can plug into the workflow of a photo-finishing lab, and determine the orientation of the images as they pass through.

1.4 Methodology

For a classifier, we chose to use SVMs because of their documented characteristics of good generalisation to unseen data, fast training times, good handling of input in high dimensions, highly configurable kernel functions, and their ability to select features that separate input samples from one another. It is likely that the number of features presented to the classifier plays a role in the classification accuracy, and since SVMs can handle input samples with many features, we investigate the optimal number of

Stellenbosch University http://scholar.sun.ac.za

features that produce the best detection accuracy. We compare the SVM approach to the automatic detection of image orientation with Bayesian learning [106] and artificial neural networks. We extract the same features as those extracted in [106] so that the detection performance of the system can be evaluated and compared with various other classifiers.

Our study focuses on firstly investigating the performance of the various kernels available in order to determine the kernel with the best training and generalisation performance. We then focus on refining the best performing kernel by optimising the various parameters associated with it. We also investigate how the number of features extracted from an image affects the classification performance and speed of the various kernels. Once we have found the optimal combination of features and parameters, we implement the technique that converts SVM output to probabilistic values, and evaluate the performance of this technique.

1.5 Accomplishments

We have shown that SVMs can be successfully used for the detection of the orientation of images; they outperform both neural networks and Bayesian learning in terms of training time, classification time and accuracy.

We also successfully implemented the technique for the conversion of the real-valued to probabilistic output values. We applied this technique to the probabilistic detection of orientation of both individual images as well as entire films. For the latter, this technique proved to significantly increase the correct orientation detection.

Our implementation also showed the versatility of SVMs to different problems, merely by substituting the kernel. This allows the selection of the kernel to be matched to a particular application. The optimal selection of a kernel, however, remains an open research question. We also gained some valuable insights on how different kernels perform on the same problem, and where each kernel's advantages and disadvantages lie. Along with the selection of the kernel, we also learned the importance of the selection of the correct kernel parameters.

1.6 Thesis Outline

The remainder of this thesis is organised as follows: in Chapter 2, we discuss several techniques for the extraction of features, and provide a brief overview of several of the features that can be extracted from images. This will be followed by detailed presentations of Bayesian learning and neural networks

Stellenbosch University http://scholar.sun.ac.za

and SVMs in Chapter 3. In addition, we will describe several techniques for the improvement and modification of the basic SVM algrithm that have been proposed in the literature. In Chapter 4, we will present results on the application of the above three learning methods for automatic image orientation detection and compare their performances. Chapter 5 applies the work done in Chapter 4 in to a real world application in the photo-finishing industry. Chapter 6 provides a conclusion and several suggestions for future research.

Stellenbosch University http://scholar.sun.ac.za

Chapter 2

Image Processing

2.1 Introduction

The human visual system (HVS) is one of the most advanced and complex systems found in nature. It is also extremely adaptive. At this present time, there is no artificial system that can compete with it. A large amount of research has been done on trying to model the HVS for visual perception and cognition. This research has produced valuable data on how humans process visual information, and on how to extract and present this information to an artificial system. With computer vision becoming more and more of a reality as both computers and algorithms are improved, a wide variety of work has recently been done on feature selection and representation in images. The aim of image processing is to reduce the dimensionality of an image, while at the same time extracting the necessary information from the image. In this chapter, we will discuss some of the feature extraction techniques, along with the advantages and disadvantages of the features themselves.

2.2 Low Level Image Features

First, a definition of image segmentation is proposed in the context of image understanding. Image segmentation is the process whereby an image is broken into consistent regions, with the region boundaries being broadly similar to where a person might believe a boundary to be. The region boundaries will necessarily reflect changes in image properties such as colour and shape, rather than corresponding to real world object boundaries. Extracting features from images that do not contain a common theme is very difficult. Segmenting such images to extract objects is not possible, as there

Stellenbosch University http://scholar.sun.ac.za

are no common objects in the image. The images therefore need to be described as a whole, and techniques need to be developed that do not require segmentation of the image into objects. This excludes all boundary-based techniques and most area-based techniques, leaving low level, pixel-based feature statistics, along with edge detection techniques. These low level features can be divided into three groups, namely colour, texture and shape.

2.2.1 Colour

Humans are able to use colour to effectively differentiate natural objects within an image, but many artificial objects cannot be differentiated on the basis of colour alone [105]. Colour can be effectively used for object segmentation and recognition in images. Recent work includes colour indexing using histogram intersection [92, 100], image retrieval using distance and reference colour methods [61] and Principal Component Analysis (PCA) on colour features [74]. An important characteristic of histogram based approaches is that they are generally invariant to image translation. While this is an advantageous characteristic for image classification, it is counter-productive when trying to determine the orientation of an image. To avoid this invariance, the following techniques are used.

Average Colour

An image is divided into a number of user-defined regions, and for each region an average pixel value is calculated in RGB (or in any other colour space) format. This technique provides information on the spatial colour composition of the image. The image is now represented by a vector of considerably lower dimensionality than the original image, while still containing information about the colour distribution within the image.

Colour Moments

The concept of colour moments was first introduced in [97]. From probability theory, it is given that a probability distribution is uniquely characterised by its moments. We interpret the colour distribution of an image as a probability distribution, thus allowing the colour distribution to be characterised by its moments. The first three moments, namely the mean, the variance and the skewness are then calculated for each colour channel (RGB), in each region; because of the varying dynamic ranges, they are then normalised to zero mean and unit variance.

Stellenbosch University http://scholar.sun.ac.za

2.2.2 Shape Features

Colour alone is not effective for the classification of images in large databases, or for the classification of grey-scale images. Therefore, other features are needed, such as texture and shape. Shape features include area, circularity and moment invariants. Several techniques have been presented using shape features. These include shape matching using relaxation techniques [16], polygon approximation of the shape [92], image representation using strings [12, 41], shape from texture and shape from disparity gradients [26], and polygon approximation combined with PCA [8]. Other techniques include histogram of edge directions, co-occurrence of edge directions, Fourier features, polar Fourier features and log-polar Fourier features [6].

Shape-based representations are generally not invariant to changes in position, size and orientation, which makes them ideal for the determination of image orientation. Unfortunately, extraction of shape features is CPU intensive, and thus it is not a solution when dealing with a real-time system.

2.2.3 Texture

No strict and universally accepted definition of texture exists, even though the notion of image texture is a familiar and intuitive one to most viewers. Texture often refers to the pattern of intensity variations in an image region and can be described by coarseness, contrast, orientation and direction. This assumes that some regularity exists in the image structure to be identified as a pattern. There are many techniques that can be used for texture classification, such as techniques based on co-occurrences [30], generalised co-occurrences [17], simultaneous autoregressive models [44, 58, 25], Tamura features [25], Markov random fields [15], quadrature mirror filters [52], tree structured wavelets [10], multiple Gabor filters [113], colour Gabor texture features [94], multi-orientation filter banks [57], second-moment matrix [21, 26].

Features based on texture are also not invariant to changes in rotation, and are also extremely CPU intensive, thus making them unacceptable for our application. An example of texture features is given in Fig. 1.

2.3 Edge Detection Techniques

Edge detection has been used to identify and isolate objects within images for the use in image identification and classification. Through the use of edge detection, images can be represented by far fewer features while still containing information about object location and interaction. These features are

Stellenbosch University http://scholar.sun.ac.za

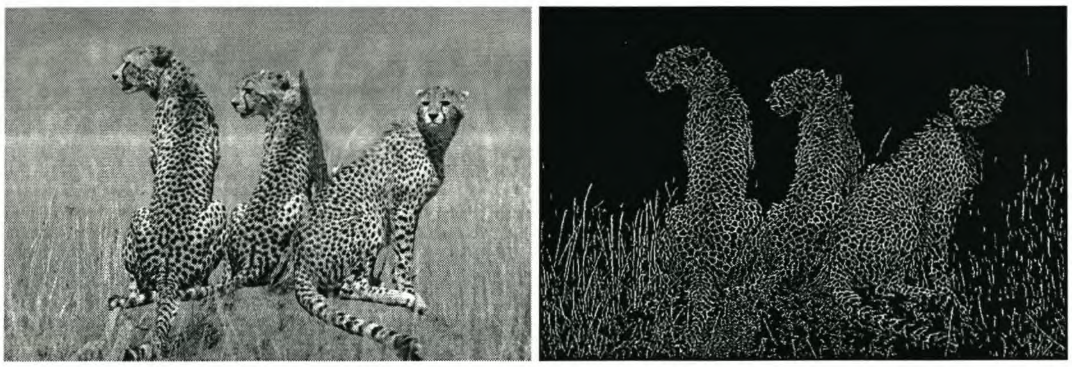


Figure 1: Extraction of regions of texture according to colour and orientation, showing the dimensionality reduction [110].

not invariant to rotation, and thus could be used for the determination of image orientation. There is no fixed definition of an edge within an image because images are inherently discrete, and for a function defined on a discrete domain, there is no natural notion of a discontinuity. Edge detection concerns the localisation of significant variations of the grey level image and the identification of the physical phenomena that produced them [35].

2.3.1 Edge Definition

The use of physical edges as features helps to convey information concerning the shape, form and texture of objects. These physical edges correspond to changes in illumination, reflectance, depth, orientation and texture of the surfaces of objects. In images, physical edges are represented by a change in the intensity function, namely steps, lines, and junctions.

Steps result between two regions of constant but different grey levels, e.g. when one object partially obscures another, or when an object is in partial shadow. These are the most common type of edges, and the step edge is the point at where the grey level discontinuity occurs. Lines are formed by the mutual illumination between objects that are in contact or from a thin object placed against a background. Lines correspond to the local extrema of the image. A junction or corner is formed when at least two edges meet. Illumination effects or occlusions can also cause junctions. A junction is a 2D feature, and is sharp.

Stellenbosch University http://scholar.sun.ac.za

2.3.2 Edge Detection Techniques

Roberts [83] presented a technique for the detection of edges in 1965, and since then numerous other approaches have been proposed for the detection of edges in images. We briefly discuss some common edge detection algorithms.

The Canny edge detection algorithm [9] is a multi-stage base algorithm and is widely used in edge detection. The image is first smoothed by Gaussian convolution to remove any noise within the image. Then a simple 2D first derivative operator similar to the Roberts Cross is applied to the smoothed image to highlight regions with high first spatial derivatives within the image. Edges within the image produce ridges in the gradient magnitude image. The algorithm then tracks along the top of these ridges and sets all the pixels that are not on the ridge top to zero, thus producing a thin line; this process is known as non-maximal suppression. This tracking process exhibits hysteresis, and is controlled by two thresholds: $T1$ and $T2$ with $T1 > T2$. Tracking can only begin at a point on a ridge higher than T1. Tracking then continues in both directions out from that point until the height of the ridge falls below T2. This hysteresis helps to ensure that noisy edges are not broken up into multiple edge fragments.

The Rothwell edge detection algorithm [86] uses dynamic thresholding that varies the edge strength threshold across the image, and is thus unique. This algorithm is very similar to the Canny algorithm mentioned above in that Gaussian smoothing is followed by differentiation. The Rothwell algorithm differs from the Canny algorithm in that edge thinning is done as a post edge detection process and that dynamic thresholding is used instead of hysteresis.

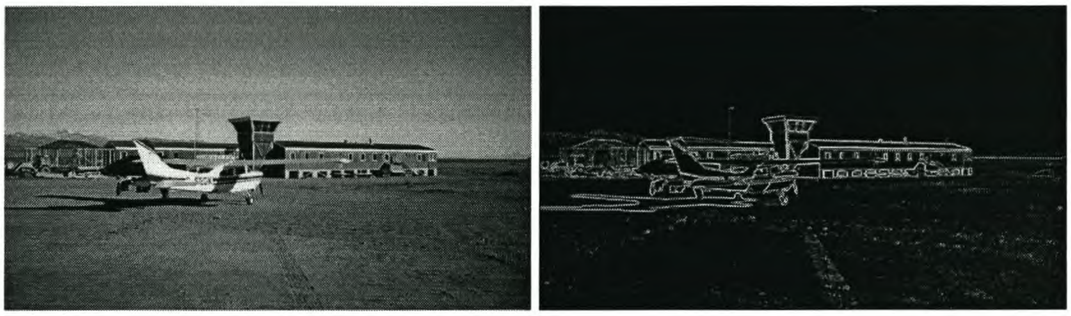
The Iverson logical/linear edge detector algorithm [42] is a technique to improve the performance of a linear edge detection algorithm as it includes logical checks for the existence of an edge. This reduces the number of false positive edges detected without losing sensitivity to the detection of true edges.

The Nalwa edge detection algorithm [68] makes use of surface fitting for edge detection. It differs from the Canny technique in that the derivative of the image is not calculated, but rather hyperbolic and quadratic functions are fitted to the image intensities in a 5×5 pixel window that is scanned across the image. If the quadratic fit has an error greater than the hyperbolic fit, a candidate edge is marked. To reduce the number of spurious edges, the contrast of the candidate edges is then thresholded.

The Bergholm edge detection algorithm [3] uses a scale space representation to find significant edges. By blurring the image with a Gaussian filter and finding the pixels that have a gradient that is both a local maximum and greater than a threshold value, edges are initially detected in the image at a courser resolution. These edges are then tracked back successively through scale space to finer resolutions according to certain threshold values.

Stellenbosch University http://scholar.sun.ac.za

An example of edge detection and extraction is given in Fig. 2 .



(a) (b)

Figure 2: Edge detection using the Canny technique.

2.4 High-Level Features

2.4.1 Object Identification and Recognition

High-level tasks such as object identification and recognition require models and algorithms which deal with the components of the image and how they interact with one another within the image. Object identification is the task of deciding whether two independently observed objects are the same object while object recognition deals with the assignment of objects into different categories. In order to distinguish between objects, more information is required, and this information is generally provided through the use of contextual and/or functional information. Contextual information provides information about the spatial and spatio-temporal relationship of an object and its surroundings. Functional information provides information concerning the purpose or use of an object. On their own, contextual and functional information provide very little help in the identification and recognition of objects. It is the relationships between different features that allows the identification and recognition of objects. This leads to further problems, as object identification and recognition are only as good as the features and relationships used with which to make the informed decisions [95].

2.4.2 Scene Recognition

Scene recognition is the ability to say with some degree of certainty that a particular figural 3D structured environment has been experienced before. Often, scene recognition is broken down into smaller

Stellenbosch University http://scholar.sun.ac.za

tasks of recognising objects within an image, and then using the objects and the relationships between them to recognise a scene. It has also been shown that coloured blobs can mediate the recognition of scenes without the recognition of the objects with the scenes. This implies that scene recognition is more concerned with the relationships between objects, rather than the objects themselves. This method is computationally expensive, and limited to a predetermined genre of image or scene.

2.5 Video Stream Recognition

In essence, video stream recognition is the repeated recognition of individual images or frames. This calls for extremely fast processing of the individual frames and a technique for the correlation of information between frames. Often, detection techniques are needed in order to track objects that are changing in both appearance and position across frames. Video stream recognition thus produces vast amounts of data, and it is therefore paramount that only the required information be extracted from the frames in order to minimise overhead and maximise the classification accuracy.

2.6 Summary

While several techniques for the extraction of information from images have been presented here, only low level features will be investigated further, as the overhead incurred in implementing some of the other feature extraction techniques will not make a real-time application feasible. Low-level features minimise the preprocessing overhead while still containing sufficient information that can be conveyed to a classifier where a decision can be made.

A factor limiting some of the techniques mentioned above is the limitless content that images can contain when trying to determine image orientation. This disqualifies any technique that searches images for specific objects, and leaves all the techniques that perform a function on the image, such as average colour features, texture features and edge features.

SVMs are able to handle input data of high dimensionality; thus, vast amounts of information can be presented to the SVM, and can only improve the classification performance.

Stellenbosch University http://scholar.sun.ac.za

Chapter 3

Machine Learning Methods for Image Classification

3.1 Introduction

Machine learning is a relatively new and exciting branch of Computer Science, and has successfully been applied to a whole host of applications involving the identification and recognition of characteristics buried within data, that traditional methods have failed upon. Researchers have successfully applied machine learning in fields as far apart as breast cancer detection [96] to pilotless helicopters [65] with high levels of success. As new and improved machine learning techniques are discovered, so the field of application of such techniques broadens, with researchers attempting to solve old problems with new solutions. Within the field of machine learning, there is a vast diversity of algorithms and techniques, each suited a particular kind of problem. They include Bayesian learning, artificial neural networks, support vector machines [107], reinforcement learning, genetic algorithms [37, 36] and decision tree learning [80], to name a few. We will focus our attention on the first three techniques, as they have been successfully applied to problems similar to the detection of image orientation.

While it is important to extract the relevant information from the images, selection of an appropriate classifier to work on the extracted information is more important. There has been some recent work by Vailaya in [106] in which a Bayesian classifier was used to determine the orientation of images. It is against this technique that the use of SVMs will be compared. Neural networks are included to provide a comparison between a traditional machine learning technique, and the relatively new technique of SVMs.

Both Bayesian learning and neural networks are introduced and explained in detail along with their

Stellenbosch University http://scholar.sun.ac.za

advantages and disadvantages. Support vector machines are then discussed in great detail, and several techniques for the improvement of the basic algorithm are mentioned, including several methods for the extension of SVMs to multi-class problems as well as algorithms to produce probabilistic outputs from SVMs. The use of traditional kernels is explained, along with the presentation of kernels used in other research and why they were used.

For each technique discussed, several examples are provided where each technique has been used for the task of classification or recognition of images or video streams and the accuracy of each technique obtained.

3.2 Bayesian Learning

Bayesian learning is based on the assumption that the quantities of interest are governed by probability distributions and that optimal decisions can be made by reasoning about these probabilities along with the observed data. Bayesian learning provides a paradigm for the analysis of the operations of other algorithms that do not explicitly manipulate probabilities, but more importantly provides the fundamentals for learning algorithms that directly manipulate probabilities.

3.2.1 Bayes Theorem

Bayes theorem [2] provides a way to calculate the probability P of a hypothesis h based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself. Let $P(h)$ denote the initial probability of hypothesis h, which is called the *prior probability* of h and may reflect any prior knowledge about the chance that h is correct. Similarly, let $P(D)$ denote the prior probability that the training data $D = \langle\langle x_1, d_1\rangle \ldots \langle x_m, d_m\rangle\rangle$ will be observed. To simplify the discussion in this section, we assume the sequence of instances $\langle x_1, \ldots, x_m\rangle$ is held fixed, so that the training data can be written simply as the sequence of target values $D = \langle d_1, \ldots, d_m\rangle$. Next, let $P(D|h)$ denote the probability of observing the training data D given hypothesis h holds. $P(h|D)$ is the probability that h holds given the training data D, and is called the *posterior probability* of h because it reflects the confidence that h will hold after the training data D has been seen. Bayes' theorem provides a way to calculate the posterior probability $P(h|D)$ from the prior probability $P(h)$, $P(D)$ and $P(D|h)$:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Stellenbosch University http://scholar.sun.ac.za

A simple concept learning algorithm, based on Bayes theorem, can be designed to output the *maximum a posteriori* (MAP) hypothesis, as follows:

Brute Force MAP Learning algorithm

1. Calculate the posterior probability for each hypothesis h in hypothesis space H

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

2. Output the hypothesis h_{MAP} with the highest posterior probability

$$h_{MAP} = \underset{h \in H}{\text{argmax}}\ P(h|D)$$

where the following assumptions are made :

1. The target concept c is contained in the hypothesis space H.
2. The training data D is noise free.
3. There is no a priori reason to believe that any one hypothesis is more probable than any other.

Using the assumptions above, we can assign the same prior probability $P(h)$ to each hypothesis h in H, and because the target concept is contained in H, these prior probabilities should add up to 1. Therefore, we choose

$$P(h) = \frac{1}{|H|}$$

$P(D|h)$, where $D = \langle d_1, \ldots, d_m \rangle$ are the target values for the fixed instances $\langle x_1, \ldots, x_m \rangle$ given that hypothesis h holds, is 1 if $d_i = h(x_i)$ and 0 if $d_i \neq h(x_i)$. Therefore

$$P(D|h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

When hypothesis h is inconsistent with training data D, from Eq. (1) $P(D|h)$ is 0. Then

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0 \quad \text{if } h \text{ is inconsistent with } D$$

Stellenbosch University http://scholar.sun.ac.za

In the case where hypothesis h is consistent with the training data D, Eq. (1) defines $P(D|h)$ to be 1. Then

$$\begin{aligned} P(h|D) &= \frac{1 \cdot \frac{1}{|H|}}{P(D)} \\ &= \frac{1 \cdot \frac{1}{|H|}}{\frac{|V\,S_{H,D}|}{|H|}} \\ &= \frac{1}{|V\,S_{H,D}|} \quad \text{if } h \text{ is consistent with } D \end{aligned}$$

where $V\,S_{H,D}$ is the number of hypotheses H that are consistent with D.

Using Bayes theorem, the posterior probability $P(h|D)$ under the assumed $P(h)$ and $P(D|h)$ is

$$P(h|D) = \begin{cases} \dfrac{1}{|V\,S_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$

where every consistent hypothesis is a MAP hypothesis.

3.2.2 Bayes Optimal Classifier

In order to determine the most probable classification of a new instance given the training data, a *Bayes optimal classifier* is defined as follows

$$\underset{v_j \in V}{\text{argmax}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) = P(v_j|D) \tag{2}$$

where v_j is a new example from some set V, then $P(v_j|D)$ is the probability that the v_j is correctly classified. Because the Bayes optimal classifier obtains the best performance from the given training data, it is computationally expensive. This method stores a simple probabilistic summary for each class containing the conditional probability of each attribute value given the class, along with the probability of the class. It has been shown that this method has approximately the same representational power as the perceptron (Section 3.3.1), as they both describe a single decision boundary through the

Stellenbosch University http://scholar.sun.ac.za

instance space. The order of the training instances and the occurrence of classification errors have no effect on the training process.

3.2.3 Gibbs Algorithm

The Gibbs algorithm, proposed by Opper and Haussler [69, 70], is often used as a replacement to a Bayes optimal classifier, as it is not as computationally expensive to implement. This is a useful characteristic, and provides the trade off between complexity and classification accuracy. The Gibbs algorithm applies a hypothesis drawn at random according to the current posterior probability distribution in order to classify a new instance. It is not optimal in the Bayes sense, but it is computationally less expensive. The algorithm is as follows:

1. Randomly chose a hypothesis h from H, according to the posterior probability distribution over H
2. The next instance x can be classified using h.

It has been shown that the Gibbs algorithm produces an misclassification error that is at most twice that of the Bayes optimal classifier [33], and so is ideally suited to problems on which the Bayes optimal classifier achieves a very low error, where an increase in classification speed is required.

3.2.4 Naive Bayes Classifier

The *naive Bayes classifier* is a highly practical Bayesian learning method, whose performance it has been shown to be comparable to that of decision tree and neural network learning. This method is used where the target function $f(x)$ can take on any value from some finite set V and where each instance x is described by a conjunction of attribute values. A training set of the target function is provided, and a new instance $\langle a_1, a_2, \ldots a_n \rangle$ is presented. This new instance is then classified and the target value is predicted.

To classify the new instance, the Bayesian approach assigns the most probable target value v_{MAP} to the new instance.

$$v_{MAP} = \underset{v_j \in V}{\text{argmax}} \; P(v_j | a_1, a_2, \ldots, a_n)$$

This expression can be rewritten using Bayes theorem:

Stellenbosch University http://scholar.sun.ac.za

$$\begin{aligned} v_{MAP} &= \underset{v_j \in V}{\mathsf{argmax}} \frac{P(a_1, a_2, \ldots, a_n | v_j) p(v_j)}{P(a_1, a_2, \ldots, a_n)} \\ &= \underset{v_j \in V}{\mathsf{argmax}}\ P(a_1, a_2, \ldots, a_n | v_j) p(v_j) \end{aligned} \tag{3}$$

The naive Bayes classifier uses the assumption that, given the training value, the attribute values are conditionally independent. The probability of observing the conjunction $a_1, a_2, \ldots, a_n$ given the target value of an instance is the product of the probabilities for the individual attributes:

$$P(a_1, a_2, \ldots, a_n | v_j) = \prod_i P(a_i | v_j)$$

Thus, the naive Bayes classifier can be defined as

$$v_{NB} = \underset{v_j \in V}{\mathsf{argmax}}\ P(v_j) \prod_i P(a_i | v_j) \tag{4}$$

where v_{NB} is the target value that the naive Bayes classifier outputs.

The difference between the naive Bayes method and previous Bayesian methods is that there is no explicit search through the space of possible hypotheses. Counting the various data combinations within the training examples is enough to form the hypothesis.

3.2.5 Bayesian Learning for Image Processing

A system was proposed in [81] for the development of a user interface based on vision and speech using Bayesian networks. In order for a user to interact with a computer system through speech, the speaker first has to be detected. This detection was extended to include more abstract properties such as the user's level of frustration or interest. The speaker detection was accomplished by using a combination of vision sensors: a face detector, a skin colour detector and a mouth motion detector. Using a number of naive Bayes classifiers for the various components, the system was able to achieve an accuracy of 94% on a frontal dataset, and 89% on a non-frontal dataset. The system also achieved 98% on non-faces.

In [106], an algorithm for the automatic detection of image orientation is presented that makes use of Bayes decision theory. The features extracted from the images include colour moments, colour histograms, edge detection histograms and MRSAR (Multi-resolution Simultaneous Autoregressive

Stellenbosch University http://scholar.sun.ac.za

Models) texture features. It was found that the colour moment features produced greater classification accuracy than the other features extracted. A feature clustering algorithm was used to reduce the number of redundant features. Features belonging to the same cluster were averaged to produce new features. The Bayesian classifier was then trained using the spatial colour moments. A maximum classification accuracy of 87.8% was achieved on the test set. The system is computationally fast, classifying an image in approximately 1.3 msec.

3.3 Artificial Neural Networks

In part, artificial neural networks (ANNs) were inspired by the observation that biological learning systems are constructed of complex arrangements of interconnected neurons and attempt to make use of some of the organisational principles present within these systems. Simply put, ANNs are a densely interconnected set of simple units, where each unit takes a number of real-valued inputs and produces a single real valued output.

A biological example of such a network is the human brain, and is estimated to contain approximately 10^{11} neurons in a densely interconnected network, with each neuron connected to approximately 10^4 other neurons. While ANNs are loosely modeled on biological neural systems, many of the complexities of biological neural systems are not modeled by ANNs, and many features of ANNs discussed later are known to be inconsistent with biological systems.

McCulloch and Pitts [60] introduced a computational model of a neuron and a logical calculus of neural networks in 1943. Rosenblatt introduced the perceptron in 1958 [84], and in 1962 [85] presented the first proof of the perceptron convergence theorem. Minsky and Papert [63] showed that certain simple computations such as XOR could not be learned or represented by single layer perceptron networks, thus dampening the interest ANNs received during the 1970's. In 1982, Hopfield [38] introduced the idea of an energy function from statistical physics, and in 1986 Rumelhart [88] reported the development of the backpropagation algorithm which made the multilayer perceptron a popular choice to solve a wide variety of pattern recognition problems. The error backpropagation algorithm was independently discovered by Werbos in 1974 [114] and later rediscovered by Parker [73] and LeCun [54] in 1985.

Artificial neural networks can be used for problems with the following characteristics:

- Instances are described by many real-valued feature-value pairs.
- The target function can be discrete-valued, real-valued, or a vector of discrete- or real-valued

Stellenbosch University http://scholar.sun.ac.za

features.

- The training data may contain errors.
- Long training times are acceptable.
- Fast evaluation of target function is needed.

A brief introduction to ANNs follows [43, 51, 64].

3.3.1 Artificial Neural Network Fundamentals

The Perceptron

A single layer feed-forward network consists of one or more output neurons o, each of which is connected with a weighting factor $\mathbf{w}_{io}$ to all of the inputs i. The total input to the neuron is the sum of the bias θ plus the weighted inputs. The output of the neuron is a function of the input. More precisely, given the inputs x_i where $i = 1, \ldots, n$ the output $y(x_1, \ldots, x_n)$ is given by

$$y = \mathcal{F}\left(\sum_{i=1}^{n} \mathbf{w}_i x_i + \theta\right) \tag{5}$$

The activation function $\mathcal{F}$ can be a number of functions, e.g., a piecewise linear, sigmoid or Gaussian. The original perceptron function was the threshold function:

$$\mathcal{F}(s) = \begin{cases} 1 & \text{if } s > 0 \\ -1 & \text{otherwise} \end{cases} \tag{6}$$

Some specialised forms of the neuron include the Perceptron, proposed by Rosenblatt [84], and the Adaline, proposed by Widrow and Hoff [116].

Perceptron Learning Rule

In order for a neuron to produce the correct output for each training example, the weight vector has to be adjusted. One method to learn the form of the weight vector is to start with randomly assigned weights, and while iterating through the training set, adjust the weights whenever the neuron misclassifies an example. This is repeated until all examples in the training set are correctly classified.

Stellenbosch University http://scholar.sun.ac.za

Let $\mathcal{X}$ be the input vector and $d(\mathcal{X})$ be the desired output, which is usually $+1$ or -1. Then the weights and threshold are updated by

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \Delta\mathbf{w}_i(t)$$

$$\theta(t+1) = \theta(t) + \Delta\theta(t)$$

The value for $\Delta\mathbf{w}_i(t)$ is determined if $y \neq d(\mathcal{X})$ and modified according to: $\Delta\mathbf{w}_i = d(\mathcal{X})x_i$. The threshold $\Delta\theta(t)$ is modified according to:

$$\Delta\theta = \begin{cases} 0 & \text{if perceptron responds correctly} \\ d(\mathcal{X}) & otherwise \end{cases}$$

Convergence Theorem

The perceptron convergence theorem proves that the perceptron learning rule will converge to a solution, provided the training set is linearly separable, in an infinite number of steps for any initial choice of weights in $\mathbf{w}^*$ [63].

Let $\mathbf{w}^*$ be the initial connection weights, $\mathbf{w}$ the updated weights after training, $\mathbf{w}^{'}$ the weights after one weight update, and x is a training example. For the proof of this theorem, let $\| \mathbf{w}^* \| = 1$. Because $\mathbf{w}^*$is a correct solution, the dot product $| \mathbf{w}^* \cdot x |$will be greater than 0 or: there exists a $\delta > 0$ such that $| \mathbf{w}^* \cdot x | > \delta$ for all inputs x. Now we define $\cos\alpha \equiv \mathbf{w} \cdot \mathbf{w}^* / \| \mathbf{w} \|$. We know that $\Delta\mathbf{w} = d(\mathcal{X})x$ and that the weight after modification is $\mathbf{w}' = \mathbf{w} + \Delta\mathbf{w}$. From this it follows that:

$$\begin{aligned} \mathbf{w}' \cdot \mathbf{w}^* &= \mathbf{w} \cdot \mathbf{w}^* + d(\mathbf{x}) \cdot \mathbf{w}^* \cdot \mathbf{x} \\ &= \mathbf{w} \cdot \mathbf{w}^* + sgn(\mathbf{w}^* \cdot \mathbf{x})\mathbf{w}^* \cdot \mathbf{x} \\ &> \mathbf{w} \cdot \mathbf{w}^* + \delta \end{aligned}$$

and

$$\begin{aligned} \|\mathbf{w}'\|^2 &= \|\mathbf{w} + d(\mathbf{x})\mathbf{x}\|^2 \\ &= \mathbf{w}^2 + 2d(\mathbf{x})\mathbf{w} \cdot \mathbf{x} + \mathbf{x}^2 \\ &< \mathbf{w}^2 + \mathbf{x}^2 \\ &= \mathbf{w}^2 + M \end{aligned}$$

Stellenbosch University http://scholar.sun.ac.za

After t iterations we have

$$\begin{aligned} \mathbf{w}(t) \cdot \mathbf{w}^* &> \mathbf{w} \cdot \mathbf{w}^* + t\delta \\ \| \mathbf{w}(t) \|^2 &< \mathbf{w}^2 + tM \end{aligned}$$

such that

$$\begin{aligned} \cos \alpha(t) &= \frac{\mathbf{w}^* \cdot \mathbf{w}(t)}{\| \mathbf{w}(t) \|} \\ &> \frac{\mathbf{w}^* \cdot \mathbf{w} + t\delta}{\sqrt{\mathbf{w}^2 + tM}} \end{aligned}$$

From this it follows that

$$\begin{aligned} \lim_{t \to \infty} \cos \alpha(t) &= \lim_{t \to \infty} \frac{\delta}{\sqrt{M}} \sqrt{t} \\ &= \infty \end{aligned}$$

but $\cos \alpha \leq 1$. Therefore there must be an upper limit $t_{\max}$ for t. This means that the perceptron learning rule only modifies the weight vector $\mathbf{w}$ a limited number of times. In other words, after $t_{\max}$ modifications of the weight vector $\mathbf{w}$, the perceptron is correctly classifying all input examples. $t_{\max}$ will be reached when $\cos \alpha = 1$. If the initial weight vector $\mathbf{w} = 0$, then

$$t_{\max} = \frac{M}{\delta^2}$$

Multi-layer Feed-forward Networks and Backpropagation

Minsky and Papert showed that a two layer network could overcome the restrictions of the single layer network, but did not offer a solution on how to adjust the weights from input to hidden units. The solution to this problem, first published by Werbos, is to back-propagate the errors of the neurons in the output layer, and this learning rule is often referred to as *error backpropagation*.

A feed-forward network is made up of layers. Each layer is made up of a number of neurons and receive their input from neurons in the previous layer, and provide input to the next layer. In general, a standard L-layer feed-forward network consists of a single input layer, $L - 2$ hidden layers, and a single output layer which are successively connected with no connections within a layer and no

Stellenbosch University http://scholar.sun.ac.za

feedback between layers. While backpropagation can be applied to networks with multiple hidden layers, it has been shown that networks of only a single hidden layer suffices to approximate any function, provided the activation functions of the hidden layer are non-linear [39].

The backpropagation algorithm for each training pair $\langle \vec{x}, \vec{t} \rangle$, where $\vec{x}$ is the vector of inputs, and $\vec{t}$ is the vector of target outputs, η is the learning rate, δ is the error signal, and o_n is the output of neuron n, can be described as follows:

- Randomly initialise all network weights
- Until the stopping criteria are met, do
  - For each $\langle \vec{x}, \vec{t} \rangle$ in the training set, do
    - *Propagate the input forward through the network:*
      - Input the instance $\vec{x}$ to the network and calculate o_n for every neuron n in the network
    - *Propagate the errors backward through the network:*
      - For each output neuron k, calculate the error term δ_k

        $$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k)$$

      - For each hidden neuron h, calculate the error term δ_h

        $$\delta_h \leftarrow o_h(1 - o_h) \sum_{k \in outputs} w_{kh}\delta_k$$

      - Update each network weight $\mathbf{w}_{ji}$

        $$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

        where

        $$\Delta w_{ji} = \eta \delta_j x_{ji}$$

A variation on the back-propagation algorithm is to alter the weight-update rule above so that the weight update for the nth iteration depends partially on the update that occurred in the $(n-1)$th iteration, as follows:

Stellenbosch University http://scholar.sun.ac.za

$$\Delta w_{ji}(n) = \eta \delta_j x_{ji} + \alpha \Delta w_{ji}(n-1)$$

where $0 \leq \alpha \leq 1$ is a constant called the *momentum*. The second term is thus called the *momentum term.*

Although the backpropagation algorithm is relatively successful, having been successfully applied to numerous machine learning problems, it does have some disadvantages. If a non-optimum learning rate and momentum constant are selected, the training process can take a long time. Outright training failures can also occur, namely from network paralysis and local minima.

Network paralysis. As a network trains, the weights can assume very large values. The total input to a hidden or output neuron can therefore reach very high (either positive or negative) values, and because of the sigmoid function, the neuron will either have an activation close to zero, or close to one. The corresponding weight adjustments will then be very small and the training process virtually halt.

Local minima. The error surface of a complex network is made of of many hills and valleys. The danger of gradient descent based approaches such as backpropagation is that the network can get trapped in a local minimum, when a smaller minimum exists nearby. Two solutions exist: probabilistic methods may help, but tend be be slow. Adding more hidden neurons may work because of the higher dimensionality of the error space and the reduced chance of falling into a local minimum; however, it has been shown that there is an upper limit on the number of hidden neurons that can be added before the system will again get trapped in local minima. [51]

3.3.2 A Posterior Probability Estimation

The output values of trained ANNs are often regarded as estimates of posterior class probabilities. It has been shown that network outputs are the desired posteriors if the global minimum of the cost function is achieved [5, 19, 28].

Neural networks, unlike conventional Bayesian classifiers which derive the Bayesian posterior probabilities from the likelihood indirectly using Bayes Rule, produce outputs that estimate Bayesian posterior probabilities directly under the following assumptions [82] :

1. There exists an infinite amount of training data.

Stellenbosch University http://scholar.sun.ac.za

2. The network is sufficiently complex.
3. The global minimum can be reached by the training error.

These assumptions are not met in practice, and therefore ANNs may fail to approximate Bayesian posterior probabilities well.

Several techniques have been proposed to produce improved probability estimation. In [111] and [112], a technique is proposed that combines ANNs with empirical probability estimation methods. This entails remapping the outputs of the ANNs using histograms. Classification accuracy was improved by as much as 20% when compared to ANN classification without the remapping.

3.3.3 Neural Networks for Image Processing

Artificial neural networks have been used in number of image processing applications ranging from face detection and recognition to driver assistance systems and pedestrian detection.

In [53], a hybrid neural network solution is presented for face recognition which combines local image sampling, and self organising map neural network [49, 50], and a convolutional neural network, similar to the neocognitron [24, 23]. When compared with other techniques such as a Hidden Markov Model approach in [90, 91], or the Eigenface approach of [104], this method attains an error rate of 3.8% compared to 13% and 10.5%, respectively. Classification time using this hybrid neural network approach is less than 0.5 seconds.

Handmann et al. [29] present a system for driver assistance using computer vision. The entire system is complex and contains many elements, constructed from a variety of algorithms. In particular, the initial object detection and the object classification is done using multi-layer perceptrons. Although limited by processing power, the system can effectively detect and classify objects (vehicles) on urban roads and motorways.

Liang and Thorpe [56] present a system for the detection of both stationary and moving pedestrians using stereo-based segmentation and neural network-based recognition. The stereo-based segmentation allows for the extraction of objects in a changing background, while the neural network-based recognition allows for the identification of pedestrians in various clothing, shapes, sizes, poses and occlusion status. By adjusting the threshold of the output of the neural network, a pedestrian detection rate of 85.2% is achieved with a false alarm rate of 3.1%. Speedwise, the system is able to classify an image pair at a frame rate of between 3 and 12 frames per second, depending on the number of objects present.

Stellenbosch University http://scholar.sun.ac.za

In [8], a system is presented that enabled the querying of image databases based on image content. A neural network is used as a pattern classifier using ground-truth classification, and is trained on image features. Two approaches were used, pixel classification and region based classification. In the pixel classification a multi-layer perceptron is trained to label each pixel in an image as belonging to one of four classes using only local information and the response of filters around that pixel. With a single intensity feature, a classification accuracy of 73.4% is achieved. Using colour as a feature, an accuracy of 85% is achieved, while the addition of texture information boosts accuracy to 87.1%. In the region-based classification approach, 28 features are used for classification. These features include the following: shape, contextual features, colour, texture, size and rotation. A high accuracy was obtained, with 82.9% of regions and 91.1% of image area being correctly labelled.

3.4 Support Vector Machines

3.4.1 Support Vector Machine Fundamentals

SVMs operate on an induction principle called structural risk minimisation, which minimises an upper bound of the generalisation error. We give a brief overview of the SVM algorithm, specifically focusing on SVMs for pattern recognition [7, 34, 107].

Optimal Hyperplane for Linearly Separable Patterns

Let $D = (\mathbf{x}_i, y_i)_{1 \leq i \leq N}$ be a set of training examples, where example $\mathbf{x}_i \in R^d$, belongs to class $y_i \in \{-1, +1\}$ and d is the dimension of the input space, and that the pattern (class) represented by y_i is linearly separable. The objective is to determine the hyperplane that separates the two classes. The equation of this hyperplane is

$$\mathbf{w} \cdot \mathbf{x}_i + b = 0 \quad \text{for } i = 1, \ldots, N \tag{7}$$

where $\mathbf{x}$ is an input vector, $\mathbf{w}$ is the weight vector and is normal to the hyperplane, and b is the bias. From Eq.(7) we get

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b \geq 0 \quad \text{for } y_i = +1 \\ \mathbf{w} \cdot \mathbf{x}_i + b < 0 \quad \text{for } y_i = -1 \end{aligned} \tag{8}$$

Let d_+ and d_- be the shortest distance from the optimal hyperplane to the closest positive and negative

Stellenbosch University http://scholar.sun.ac.za

examples respectively. Thus the *margin of separation,* denoted by ρ, is $d_+ + d_-$. The goal of a support vector machine is to find the particular hyperplane for which ρ is maximised, and this hyperplane is thus referred to as the *optimal hyperplane.* Let w_o and b_o be the optimal values for the weight vector and bias respectively, then the *optimal hyperplane* is defined by

$$\mathbf{w}_o \cdot \mathbf{x}_i + b_o = 0 \quad \text{for } i = 1, \ldots, N \tag{9}$$

which is the optimal form of Eq.(7). The discriminant function

$$g(\mathbf{x}) = \mathbf{w}_o \cdot \mathbf{x}_i + b_o \quad \text{for } i = 1, \ldots, N \tag{10}$$

gives the algebraic measure of the distance from $\mathbf{x}_i$ to the optimal hyperplane [19]. Let

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}_o}{\parallel \mathbf{w}_o \parallel} \tag{11}$$

where $\mathbf{x}_p$ is the normal projection of $\mathbf{x}$ onto the optimal hyperplane, and r is the desired algebraic distance, where r is positive if $\mathbf{x}$ is on the positive side of the optimal hyperplane and negative if x is on the negative side. Since $g(\mathbf{x}_p) = 0$, it follows that

$$g(\mathbf{x}) = \mathbf{w}_o \cdot \mathbf{x}_i + b_o = r \parallel \mathbf{w}_o \parallel$$

or

$$r = \frac{g(\mathbf{x})}{\parallel \mathbf{w}_o \parallel} \tag{12}$$

The expression $b_o / \parallel \mathbf{w}_o \parallel$ gives us the distance from the origin to the optimal hyperplane. If $b_o > 0$, the origin (i.e., $\mathbf{x} = 0$) is on the positive side of the optimal hyperplane, and if $b_o < 0$, the origin is on the negative side. If $b_o = 0$, the optimal hyperplane passes through the origin. In order to get the optimal hyperplane, we need to find values for $\mathbf{w}_o$ and b_o, given the training set D. If there exists a hyperplane satisfying Eq. (8), the set is said to be *linearly separable.* In this case, it is always possible to rescale w_o and b_o so that

Stellenbosch University http://scholar.sun.ac.za

$$\begin{aligned} \mathbf{w}_o \cdot \mathbf{x}_i + b_o &\geq 1 \quad \text{for } y_i = +1 \\ \mathbf{w}_o \cdot \mathbf{x}_i + b_o &\leq -1 \quad \text{for } y_i = -1 \end{aligned} \tag{13}$$

This scaling operation leaves Eq. (9) unaffected.

The particular training examples $(\mathbf{x}_i, y_i)$ that satisfy either the first or second lines of Eq. (13) with the equality sign are called *support vectors*, and hence the name "support vector machines". A conceptual definition for support vectors is that they are those training examples that lie closest to the decision surface and are thus the most difficult to classify, and therefore have a direct bearing on the optimum location of the decision surface.

Suppose there is a support vector $\mathbf{x}^{(s)}$ for which $y^{(s)} = \pm 1$, then we have

$$g(\mathbf{x}^{(s)}) = \mathbf{w} \cdot \mathbf{x}^{(s)} \pm b_o = \pm 1 \quad \text{for } y^{(s)} = \pm 1 \tag{14}$$

The algebraic distance from the support vector $\mathbf{x}^{(s)}$ to the optimal hyperplane, from Eq. (12), is

$$\begin{aligned} r &= \frac{g(\mathbf{x}^{(\mathrm{s})})}{\parallel \mathbf{w}_o \parallel} \\ &= \begin{cases} \dfrac{1}{\parallel \mathbf{w}_o \parallel} & \text{if } y^{(s)} = +1 \\ -\dfrac{1}{\parallel \mathbf{w}_o \parallel} & \text{if } y^{(s)} = -1 \end{cases} \end{aligned} \tag{15}$$

where the $y^{(s)} = +1$ indicates that $\mathbf{x}^{(s)}$ lies on the positive side of the optimal hyperplane and $y^{(s)} = +1$ indicates that $\mathbf{x}^{(s)}$ lies on the negative side of the optimal hyperplane. From Eq. (15)

$$\begin{aligned} \rho &= 2r \\ &= \frac{2}{\parallel \mathbf{w}_o \parallel} \end{aligned} \tag{16}$$

As Eq. (16) shows, maximising the margin of separation between the the classes is equivalent to minimising the Euclidean norm of the weight vector $\mathbf{w}$.

Stellenbosch University http://scholar.sun.ac.za

Quadratic Optimisation for Finding the Optimal Hyperplane

A computationally efficient procedure for using the training examples $D = (\mathbf{x}_i, y_i)_{1 \leq i \leq N}$ is needed to find the optimal hyperplane, subject to the constraint

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \quad \text{for } i = 1, \ldots, N \tag{17}$$

Equation (17) combines the two lines of Eq. (13), with $\mathbf{w}$ replacing $\mathbf{w}_o$. We now have to solve the constrained optimisation problem which is stated as follows:

> Find the optimum values of the weight vector w and bias b, given the training example $(\mathbf{x}_i, y_i)_{1 \leq i \leq N}$, to satisfy the constraints
>
> $$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \quad \text{for } i = 1, \ldots, N$$
>
> and the weight vector w that minimises the cost function:
>
> $$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w} \cdot \mathbf{w}$$

For ease of presentation, we include the scaling factor $1/2$. This constrained optimization problem is now called a *primal problem,* and it has the following characteristics

- $\| \mathbf{w} \|^2$ is convex
- the constraints are linear in $\mathbf{w}$

and accordingly it can be minimised with Lagrange multipliers.

Consider the following *Lagrangian function*:

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w} \cdot \mathbf{w} - \sum_{i=1}^{N} \alpha_i [y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1] \tag{18}$$

where α_i, are auxiliary nonnegative variables, called *Lagrange multipliers*. The saddle point of the Lagrangian function $J(\mathbf{w}, b, \alpha)$, which is minimised with respect to w and b, and maximised with respect to α, solves the constrained optimization problem. By differentiating $J(\mathbf{w}, b, \alpha)$ with respect to w and b, and setting the results equal to zero, the following two conditions of optimality are obtained:

Stellenbosch University http://scholar.sun.ac.za

$$\frac{\partial J(\mathbf{w},b,\alpha)}{\partial \mathbf{w}} = 0 \qquad (1)$$

$$\frac{\partial J(\mathbf{w},b,\alpha)}{\partial b} = 0 \qquad (2)$$

Applying the first condition to Eq. (18) results in

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \tag{19}$$

while application of the second condition to Eq.(18) results in

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{20}$$

It is important to note that for each Lagrange multiplier α_i at the saddle point, the product of that multiplier with its corresponding constraint disappears, as follows

$$\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1] = 0 \quad \text{for } i = 1,\ldots,N \tag{21}$$

From the *Karush-Kuhn-Tucker conditions* of optimization theory [20] it follows that only those multipliers meeting Eq. (21) can assume nonzero values.

If we denote $\alpha = (\alpha_1,\ldots,\alpha_N)$ as the N non negative Lagrange multipliers associated with constraint (17), the optimisation problem amounts to maximising

$$W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{N} \alpha_i\alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{22}$$

with $\alpha_i \geq 0$, for $i = 1,...,N$, and the constraint $\sum_{i=1}^{N} \alpha_i y_i = 0$. Also, and more importantly, maximising $W(\alpha)$ depends only on the input patterns in the form of a set of dot products, $\{\mathbf{x}_i \cdot \mathbf{x}_j\}_{(i,j)=1}^{N}$. Equation (22) is now called the *dual problem* which differs from the primal problem in that although they arise from the same objective function, there are different constraints. Also, the primal problem is minimised, while the dual problem is maximised.

Having determined optimum Lagrange multipliers, $\alpha^o = (\alpha_1^o,\ldots,\alpha_N^o)$, of the maximisation problem

Stellenbosch University http://scholar.sun.ac.za

of Eq. (22), the optimum weight vector $\mathbf{w}_o$ has the following expansion, using Eq. (19)

$$\mathbf{w}_o = \sum_{i=1}^{N} \alpha_i^o y_i \mathbf{x}_i \tag{23}$$

To compute the optimum bias b_o, the $\mathbf{w}_o$ obtained in Eq. (23) is used, and we make use of Eq. (14) pertaining to a positive support vector

$$b_o = 1 - \mathbf{w}_o \cdot \mathbf{x}^{(s)} \quad \text{for } y^{(s)} = 1$$

The discriminant function in Eq. (10) can be expanded, using Eq. (23), into

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i^o y_i \mathbf{x}_i \cdot \mathbf{x} + b_o \qquad \text{for } i = 1, \ldots, N \tag{24}$$

Linearly Nonseparable Patterns

For data that is not linearly separable, an optimal hyperplane is still required that minimises the probability of classification error, averaged over the training set. The margin of separation between classes is said to be *soft* if an input point $(\mathbf{x}_i, y_i)$ violates Eq. (17). This violation can arise in one of two ways:

- The input point $(\mathbf{x}_i, y_i)$ falls on the right side of the decision surface, but within the region of separation.
- The input point $(\mathbf{x}_i, y_i)$ falls on the wrong side of the decision surface.

Slack variables [13] $\{\xi_i\}_{i=1}^{N}$, with $0 \leq \xi_i \leq 1$, are introduced into the definition of the separating hyperplane such that

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{for } i = 1, \ldots, N \tag{25}$$

to allow the possibility of examples that violate Eq. (17). For $0 \leq \xi_i \leq 1$, the input point falls on the right side of the decision surface, but within the region of separation. For $\xi_i > 1$, it falls on the wrong side of the separating hyperplane. The support vectors are defined exactly the same way for both linearly separable and nonseparable cases.

Stellenbosch University http://scholar.sun.ac.za

To find the separating hyperplane that minimises the misclassification error, we must minimise

$$\Phi(\mathbf{w},\xi) = \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{N}\xi_i \tag{26}$$

Minimising the first term of Eq. (26) controls the learning capacity; the purpose of the second term is to control the number of misclassifications. The parameter C is user-defined, and controls the tradeoff between the number of nonseparable input points and the complexity of the machine, and is chosen experimentally through cross validation.

It is possible to solve the optimization problem for linearly separable patterns using the optimization for nonseparable patterns. A special case, where $\xi_i = 0$ for all i, reduces Eqs. (25) and (26) to the linearly separable form.

Once again, as in the linearly separable case, using Lagrange multipliers, the optimisation problem amounts to maximising

$$W(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i\cdot\mathbf{x}_j \tag{27}$$

with $0 \leq \alpha_i \leq C$, for $i = 1,...,N$ where C is a user defined positive parameter, and the constraint $\sum_{i=1}^{N}\alpha_i y_i = 0$. The difference between the separable and nonseparable case is that the constraint $\alpha_i \geq 0$ is replaced with the stronger $0 \leq \alpha_i \leq C$.

The optimum solution for the weight vector $\mathbf{w}$ is given by

$$\mathbf{w}_o = \sum_{i=1}^{N_s}\alpha_{o,i}y_i\mathbf{x}_i \tag{28}$$

where N_s is the number of support vectors.

The equations

$$\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 + \xi_i] = 0 \quad \text{for } i = 1,\cdot,N \tag{29}$$

and

Stellenbosch University http://scholar.sun.ac.za

$$\mu_i \xi_i = 0 \quad \text{for } i = 1, \cdot, N \tag{30}$$

define the Karush-Kuhn-Tucker conditions. Equation (21) now becomes Eq. (29), except the unity term $(1 - \xi_i)$ is added. The μ_i are Lagrange multipliers that guarantee that the slack variables ξ_i are nonnegative for all i. The derivative of the Lagrangian function at the saddle point with respect to the slack variable ξ_i is zero, which yields

$$\alpha_i + \mu_i = C \tag{31}$$

The combination of Eqs. (30) and (31) results in

$$\xi_i = 0 \quad \text{if } \alpha_i < C \tag{32}$$

To determine the optimum bias b_o, the average over all the training points in the training set for which $0 < \alpha_{o,i} < C$ and $\xi_i = 0$ applied to Eq. (29), is taken [7].

Nonlinear Support Vector Machines

Is it possible for the above methods to be generalised to the case where the decision function is not a linear function of the data? By using a rather old method [1], this can be accomplished quite easily [4]. The only way the input points appear in the training problem, Eq. (22), is in the form of dot products, $\mathbf{x}_i \cdot \mathbf{x}_j$. Using Cover's theorem, a multidimensional input space made up of nonlinearly separable points may be transformed into a new feature space where such points are linearly separable, provided two conditions are satisfied. Firstly, the transformation is nonlinear, and secondly, the dimension of the feature space is high enough. Now, the separating hyperplane is defined as a linear function of the vectors drawn from the feature space, instead of the original input space, and this is in accordance with the principle of structural risk minimisation. Now, we need a *kernel function* to construct the optimal hyperplane in the feature space without having to consider the feature space itself.

Define a vector $\phi(\mathbf{x}) = [\phi_0, \phi_1(\mathbf{x}), \ldots, \phi_{m_1}(\mathbf{x})]^T$ so that if $\mathbf{x}$ is a vector drawn from the input space of dimension m_0, $\{\phi_j(\mathbf{x})\}_{j=1}^{m_1}$ is a nonlinear transformation from the input space to the feature space, where m_1 is the dimension of the feature space and $\phi(\mathbf{x})$ is defined *a priori* for all j. A hyperplane acting on the decision surface can be defined as follows

Stellenbosch University http://scholar.sun.ac.za

$$\sum_{j=1}^{m_1} w_j \phi_j(\mathbf{x}) + b = 0 \tag{33}$$

where $\{w_j\}_{j=1}^{m_1}$ is a set of linear weights that connects the feature space to the output space, and b is the bias. Equation (33) can then be simplified to

$$\sum_{j=0}^{m_0} w_j \phi_j(\mathbf{x}) = 0 \tag{34}$$

where it is assumed that $\phi_0(\mathbf{x}) = 1$ for all $\mathbf{x}$, so that w_0 denotes the bias b. The quantity $\phi_j(\mathbf{x})$ us the input supplied to the weight w_j via the feature space. Equation (7) can then be rewritten as

$$\mathbf{w} \cdot \phi(\mathbf{x}) = 0 \tag{35}$$

while substitution of $\phi_j(\mathbf{x})$ into Eq. (19) yields

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \phi(\mathbf{x}_i) \tag{36}$$

The decision surface in the feature space can be computed by substituting Eq. (35) into Eq. (36) resulting in

$$\sum_{i=1}^{N} \alpha_i y_i \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) = 0 \tag{37}$$

The inner product of two vectors, the input pattern pertaining to the ith example $\mathbf{x}_i$ and the input vector $\mathbf{x}$, in the feature space is presented by $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x})$. The kernel function, which is a special case of *Mercer's condition*, denoted by $K(\mathbf{x}, \mathbf{x}_i)$, is defined

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}_i) &= \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_i) \\ &= \sum_{j=1}^{m_1} \phi_j(\mathbf{x}) \phi_j(\mathbf{x}_i) \end{aligned} \tag{38}$$

From the above, it is evident that the kernel is a symmetric function, i.e.

Stellenbosch University http://scholar.sun.ac.za

$$K(\mathbf{x}, \mathbf{x}_i) = K(\mathbf{x}_i, \mathbf{x}) \quad \text{for all } i \tag{39}$$

By substituting Eq. (38) into Eq. (37), it is possible to construct the optimum hyperplane in the feature space without the need to deal with the feature space itself, and the optimal hyperplane can be defined as

$$\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) = 0$$

Mercer's Condition

Mercer's condition [62] can be stated as follows:

> There exists a mapping Φ and an expansion
>
> $$K(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{\infty} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{x}')$$
>
> where $K(\mathbf{x}, \mathbf{x}')$ is a continuous symmetric kernel that is defined in the interval $\mathbf{a} \leq \mathbf{x} \leq \mathbf{b}$ and $\mathbf{a} \leq \mathbf{x}' \leq \mathbf{b}$, and $\lambda_i > 0$ for all i, if and only if, for every $\psi(\cdot)$ such that
>
> $$\int_b^a \psi^2(\mathbf{x}) d\mathbf{x} < \infty$$
>
> then
>
> $$\int_b^a \int_b^a K(\mathbf{x}, \mathbf{x}') \psi(\mathbf{x}) \psi(\mathbf{x}')\, d\mathbf{x}\, d\mathbf{x}' \geq 0$$

The coefficients λ_i are called *eigenvalues*, and the function $\phi_i(\mathbf{x})$ are called *eigenfunctions*.

Quadratic Optimisation for Nonlinear Support Vector Machines

Although the decision surface is nonlinear in the input space, Mercer's condition ensures that the expansion of $K(\mathbf{x}, \mathbf{x}_i)$ in Eq. (38) is linear in the feature space.

As in the linear case, the optimisation of the support vector machine for the nonlinear case is as follows:

Stellenbosch University http://scholar.sun.ac.za

$$W(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{40}$$

with $0 \leq \alpha_i \leq C$, for $i = 1, ..., N$, and the constraint $\sum_{i=1}^{N} \alpha_i y_i = 0$. The difference between the linear and nonlinear case is that the inner product $\mathbf{x}_i \cdot \mathbf{x}_j$ has been replaced by the kernel function $K(\mathbf{x}, \mathbf{x}_i)$. By adapting the formula of Eq. (28) to the nonlinear case the optimum solution for the weight vector w_o is given by

$$\mathbf{w}_o = \sum_{i=1}^{N_s} \alpha_{o,i} y_i \phi(\mathbf{x}_i) \tag{41}$$

where $\phi(\mathbf{x}_i)$ is the image induced in the feature space due to $\mathbf{x}_i$, as we already have the optimum values of the Lagrange multipliers $\alpha_{o,i}$.

The discriminant function in Eq. (10) can be expanded for the nonlinear case into

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \qquad \text{for } i = 1, \ldots, N \tag{42}$$

3.4.2 Kernels

The choice of the kernel is an integral part of the architecture of the SVM. The only requirement is that the kernel $K(\mathbf{x}, \mathbf{x}_i)$ satisfies Mercer's condition. An inappropriate kernel can lead to poor classification performance. As there are no techniques to learn the form of the kernel, the kernels initially used in SVMs were borrowed from traditional pattern recognition literature:

$$\begin{aligned} K_{poly}(\mathbf{x}, \mathbf{x}_i) &= (\mathbf{x} \cdot \mathbf{x}_i + 1)^p \\ K_{gauss}(\mathbf{x}, \mathbf{x}_i) &= e^{-\rho ||\mathbf{x} - \mathbf{x}_i||^2} \\ K_{percep}(\mathbf{x}, \mathbf{x}_i) &= \tanh(\beta_0 \mathbf{x} \cdot \mathbf{x}_i + \beta_1) \end{aligned}$$

where p, ρ, β_0 and β_1 are parameters specified *a priori* by the user.

K_{poly} produces a classifier that is a polynomial of degree p in the data. K_{gauss} produces a Gaussian Radial Bias Function (RBF) classifier. K_{percep} produces a particular kind of two-layer sigmoidal

Stellenbosch University http://scholar.sun.ac.za

neural network. For K_{gauss}, the number of centers, the centers themselves, the weights α_i, and the threshold b are all determined by the SVM training; and give better results than classical RBF classifiers trained with standard methods [93]. For K_{percep}, the first layer consists of N_s sets of weights, each set consisting of d_L weights where d_L is the dimension of the input data; the second layer consists of N_s weights α_i. Thus, an evaluation requires the computation of a weighted sum of the sigmoids, which in turn are dot products of the support vectors with the test data. The structure of the neural network is thus determined by the SVM training algorithm. K_{percep} only satisfies Mercer's condition for certain values of β_0 and β_1.

More general forms of RBF kernels were investigated in [11] such as

$$K_{d-RBF}(\mathbf{x}, \mathbf{x}_i) = e^{-\rho d(\mathbf{x}, \mathbf{x}_i)} \tag{43}$$

where $d(\mathbf{x}, \mathbf{y})$ can be chosen to be any distance in the input space. It was shown that non-Gaussian kernels with exponential decay rates that are less than quadratic can lead to very good performance [11]. The kernel in Eq. (43) can be further generalised to

$$K_{d-RBF}(\mathbf{x}, \mathbf{x}_i) = e^{-\rho d_{a,b}(\mathbf{x}, \mathbf{x}_i)} \tag{44}$$

with

$$d_{a,b}(\mathbf{x}, \mathbf{x}_i) = \sum_j |x_j^a - (x_i)_j^a|^b$$

Special forms of the kernel in Eq. (44) can be obtained by varying b :

$$K_{gauss}(\mathbf{x}, \mathbf{x}_i) = e^{-\rho(\mathbf{x}^a - \mathbf{x}_i^a)^2} \qquad \text{for } b = 2$$

$$K_{laplac}(\mathbf{x}, \mathbf{x}_i) = e^{-\rho|\mathbf{x}^a - \mathbf{x}_i^a|} \qquad \text{for } b = 1$$

$$K_{sublinear}(\mathbf{x}, \mathbf{x}_i) = e^{-\rho\sqrt{\mathbf{x}^a - \mathbf{x}_i^a}} \qquad \text{for } b = 0.5$$

The above kernels satisfy Mercer's condition if and only if $0 \leq b \leq 2$ [108]. The value of a has no effect on Mercer's condition as the remapping of the input variables simply amounts to a change in the input variables.

Stellenbosch University http://scholar.sun.ac.za

Along with the three traditional kernels, the above modified RBF kernels were investigated, and their performances compared, in Section 4.4.

3.4.3 "Sigmoidal" SVMs

Classifiers that produce a posterior probability $P(\text{class}|\text{input})$, such as neural networks and Bayesian classifiers, are useful when making further decisions about the test data. Unlike other classifiers, SVMs only produce an uncalibrated value that is not a probability, but rather the distance from the optimal hyperplane to the input point being classified.

Several authors have suggested methods of producing probabilistic outputs for SVMs. Wahba [109] proposed a method of producing posterior probabilities outputs from a kernel machine using the logistic link function:

$$P(\text{class}|\text{input} = P(y=1|\mathbf{x}) = p(\mathbf{x}) = \frac{1}{1+e^{(-g(\mathbf{x}))}}$$

where $g(\mathbf{x})$ is the discriminant function in Eq. (42). Wahba then proposed minimizing a negative multinomial likelihood with a term that penalises the norm in a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{F}$:

$$-\frac{1}{m}\sum_{i=1}\left(\frac{y_i+1}{2}\log(p_i)+\frac{1-y_i}{2}\log(1-p_i)\right)+\lambda \parallel h \parallel_{\mathcal{F}}^2$$

where $p_i = p(\mathbf{x})$. The output $p(\mathbf{x})$ of such a SVM will be a posterior probability.

Vapnik [108] presented a method whereby the output of SVMs is mapped to probabilities by decomposing the feature space $\mathcal{F}$ orthogonally to the separating hyperplane, along with all the $N-1$ other dimensions of the feature space. This orthogonal direction is parameterized by t, which is a scaled version of $g(\mathbf{x})$, while all of the other directions are parameterized by $\mathbf{u}$. The posterior probability depends on both t and $\mathbf{u}$: $P(y=1|t,\mathbf{u})$. Vapnik then proposes fitting this probability with a sum of cosine terms:

$$P(y=1|t,\mathbf{u}) = a_0(\mathbf{u}) + \sum_{n=1}^{N} a_n(\mathbf{u})\cos(nt)$$

Stellenbosch University http://scholar.sun.ac.za

Another method is proposed by Hastie and Tibshirani [32] whereby Gaussians are fitted to the class-conditional densities $p(g|y = 1)$ and $p(g|y = -1)$, and a single tied variance is estimated for both Gaussians. Thus the posterior probability rule $P(y = 1|g)$ is a sigmoid, and the slope is determined by the tied variance. The bias of the sigmoid is then adjusted so that the point $P(y = 1|g) = 0.5$ occurs at $g = 0$. Although this sigmoid is monotonic, the true posterior probability may not be accurately modeled by the single parameter derived from the variances.

A more flexible version of the Gaussian fitted to $p(g|y = \pm 1)$ can be used, and the mean and the variance for each Gaussian is determined from the data set. The posterior can be computed using Bayes' rule:

$$P(y = 1|g) = \frac{p(g|y = 1)P(y = 1)}{\sum\limits_{i=-1,1} p(g|y = i)P(y = i)}$$

where $P(y = i)$ are the prior probabilities that can be computed from the training set. The posterior probability is thus an analytic function of g:

$$P(y = 1|g) = \frac{1}{1 + e^{(ag^2+bg+c)}} \tag{45}$$

The posterior estimate derived from the two-Gaussian approximation violates the strong monotonic prior, Eq. (45) is non-monotonic. Also, the assumption of Gaussian class-conditional densities is often violated [76].

Platt [76] suggests a method whereby instead of estimating the class-conditional densities $p(g|y)$, a parametric model is fitted to the posterior $P(y = 1|g)$ directly. The sigmoid

$$P(y = 1|g) = \frac{1}{1 + e^{(Ag+B)}}$$

is parametric in form and is equivalent to assuming that the SVM output is proportional to the log odds of a positive example. The parameters A and B are trained discriminatively using maximum likelihood estimation for a training set $D = (\mathbf{x}_i, y_i)$.

First, a new training set $T(\mathbf{x}_i, t_i)$ is defined, where t_i are the target probabilities defined as

$$t_i = \frac{y_i + 1}{2}$$

Stellenbosch University http://scholar.sun.ac.za

Parameters A and B are found using the cross-entropy error function which minimises the negative log likelihood of the training data

$$\min - \sum_{i=1}^{N} t_i \log(p_i) + (1 - t_i) \log(1 - p_i) \tag{46}$$

where

$$p_i = \frac{1}{1 + e^{(Ag_i + B)}}$$

The minimisation of Eq. (46) is a two-parameter minimisation. The model-trust minimization algorithm of [27], based on the Levenberg-Marquardt algorithm [79], was used for the experiments presented in Chapter 5.

3.4.4 Multi-Class SVM

Support vector machines were originally designed for binary classification. The natural progression was to produce SVMs that were able to classify multi-class data sets. There are two approaches for the construction of multi-class SVMs, the first consisting of a combination of several binary classifiers, and the second where the optimisation formulation directly considers all the data. Because the number of variables in a SVM is proportional to the number of classes, multi-class SVMs are in general computationally more expensive to train than binary SVMs.

Compound Binary Classifiers

Several methods have been suggested for multi-class SVMs based on binary classification. These include *one-against-one* [46], *one-against-all* , DAGSVM [77], the stacking approach [59], and multi-class classification with error codes [45] to name a few.

The one-against-one method was introduced in [46] and first used in [22]. In this method, $k(k-1)/2$ classifiers are constructed where each classifier is trained on data from two classes. The following binary classification problem from Eq. (26) for training data from the mth and nth classes is minimised:

$$\Phi(\mathbf{w}^{mn}, \xi^{mn}) = \tfrac{1}{2}\mathbf{w}^{mn} \cdot \mathbf{w}^{mn} + C \textstyle\sum_{i=1}^{N} \xi_i^{mn}$$

Stellenbosch University http://scholar.sun.ac.za

with the constraints

$$\begin{aligned} \mathbf{w}^{mn} \cdot \phi(\mathbf{x}_i) + b^{mn} &\geq 1 - \xi_i^{mn} \quad \text{if } y_t = m \\ \mathbf{w}^{mn} \cdot \phi(\mathbf{x}_i) + b^{mn} &\leq -1 + \xi_i^{mn} \quad \text{if } y_t = n \end{aligned}$$

where $\xi_t^{mn} \geq 0$. The following voting system suggested in [22] is used: if $\mathbf{x}$ is classified into the mth class, then the vote for the mth class is increased by one, otherwise the nth class is increased by one. If the two classes have the same number of votes, the class with the lower index is selected.

The one-against-all method was one of the original implementations of multi-class SVMs, used in [55]. In this method, k SVMs are constructed, where k is the number of classes. Positive labels are applied to the *pth* class for the *pth* SVM, and all other examples are labelled negative. The pth SVM minimises

$$\Phi(\mathbf{w}^p, \xi^p) = \frac{1}{2}\mathbf{w}^p \cdot \mathbf{w}^p + C\sum_{i=1}^{N} \xi_i^p$$

with the constraints

$$\begin{aligned} \mathbf{w}^p \cdot \phi(\mathbf{x}_i) + b^p &\geq 1 - \xi_i^p \qquad \text{if } y_i = p \\ \mathbf{w}^p \cdot \phi(\mathbf{x}_i) + b^p &\leq -1 + \xi_i^p \qquad \text{if } y_i \neq p \end{aligned}$$

where $\xi_i^p \geq 0$. The classification of $\mathbf{x}$ is determined by the SVM which returns the largest value of the decision function

$$\underset{k}{\mathsf{argmax}} \; [(\mathbf{w}_i \cdot \mathbf{x}) + b_i] \quad \text{for } i = 1, \ldots, k$$

The Direct Acyclic Graph Support Vector Machines (DAGSVM) [77] algorithm uses the same training phase as the one-against-one method by solving $k(k-1)/2$ binary SVMs. It differs from the one-against-one method in that in the testing phase it uses a rooted binary directed acyclic graph which has k leaves and $k(k-1)/2$ internal nodes. Each node is a binary SVM of the mth and nth classes. Starting at the root node, for each test sample $\mathbf{x}$, the binary decision function is evaluated. Depending on the output value, it then either moves to the left or the right. Therefore, a path through the tree is traversed before reaching a leaf node which indicates the predicted class.

The stacking approach [59] suggests the use of k SVMs stacked with a single layer neural network for

Stellenbosch University http://scholar.sun.ac.za

k classes. The neural network can be seen as a $k \times k$ mixture matrix $\mathbf{M}$ that combines the the outputs of the individual SVMs into a single, more robust answer. This approach is similar to the traditional way of solving the one-against-all method in that in the mixture matrix used in the one-against-all decomposition method is the identity mixture matrix.

All Together Classifiers

An idea similar to the one-against-all method was proposed where multi-class SVMs can be solved by solving a single optimisation problem [108, 115]. In this approach, the optimisation problem of Eq (26) is generalised to minimise the following:

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2} \sum_{m=1}^{k} \mathbf{w}_m \cdot \mathbf{w}_m + C \sum_{i=1}^{N} \sum_{m \neq y_i} \xi_i^m \tag{47}$$

with constraints

$$\mathbf{w}_{y_i} \cdot \phi(\mathbf{x}_i) + b_{y_i} \geq \mathbf{w}_m \cdot \phi(\mathbf{x}_i) + b_m + 2 - \xi_i^m \tag{48}$$

where $\xi_i^m \geq 0, i = 1, \ldots, N$ and $m \in \{1, \ldots, k\} \setminus y_i$. This gives the following decision function:

$$g(\mathbf{x}) = \arg\max_{k} [\mathbf{w}_i \cdot \phi(\mathbf{x}) + b_i] \quad \text{for } i = 1, \ldots, k$$

Following [115], the dual problem of of Eq. (47) and Eq. (48) is minimised in terms of α

$$W(\alpha) = 2 \sum_{i,m} \alpha_i^m + \sum_{i,j,m} [-\frac{1}{2} c_j^{y_i} A_i A_j + \alpha_i^m \alpha_j^{y_i} - \frac{1}{2} \alpha_i^m \alpha_j^{y_i}](\mathbf{x}_i) \cdot (\mathbf{x}_j) \tag{49}$$

with linear constraints

$$\sum_{i=1}^{N} \alpha_i^n = \sum_{i=1}^{N} c_i^n A_i \quad \text{for } n = 1, \ldots, k$$

and

Stellenbosch University http://scholar.sun.ac.za

$$\begin{array}{ll} 0 \leq \alpha_i^m \leq C & \alpha_i^{y_i} = 0 \\ i = 1, \ldots, N & m \in \{1, \ldots, k\} \setminus y_i \end{array}$$

where

$$A_i = \sum_{m=1}^{k} \alpha_i^m$$

and

$$c_i^n = \begin{cases} 1 & \text{if } y_i = n \\ 0 & \text{if } y_i \neq n \end{cases}$$

.

The decision function is:

$$\arg\max_n [\sum_{i=1}^{N} (c_i^m A_i - \alpha_i^n)(\mathbf{x}_i) \cdot (\mathbf{x}) + b_n] \tag{50}$$

The inner products of $(\mathbf{x}_i) \cdot (\mathbf{x}_j)$ in Eq. (49) and Eq. (50) can be replaced with the convolution of the inner product $K(\mathbf{x}_i, \mathbf{x}_j)$.

The method proposed by Crammer and Singer [14] also solves a single optimisation problem for multi-class classification. The following primal problem is minimised:

$$\Phi(\mathbf{w}_m, \xi_i) = \frac{1}{2} \sum_{m=1}^{k} \mathbf{w}_m \cdot \mathbf{w}_m + C \sum_{i=1}^{N} \xi_i \tag{51}$$

with the constraints

$$\mathbf{w}_{y_i} \cdot \phi(\mathbf{x}_i) - \mathbf{w}_m \cdot \phi(\mathbf{x}_i) \geq e_i^m - \xi_i \quad \text{for } i = 1, \ldots, N$$

where $e_i^m \equiv 1 - \sigma_{y_i,m}$ and

Stellenbosch University http://scholar.sun.ac.za

$$\sigma_{y_i,m} \equiv \begin{cases} 1 & \text{if } y_i = m \\ 0 & \text{if } y_i \neq m \end{cases}$$

This gives the decision function

$$\arg\max_{m}[\mathbf{w}_m \cdot \phi(\mathbf{x})] \quad \text{for } m = 1, \ldots, k$$

The significant difference from the method in [108] and [115] is that only l slack variables ξ_i, where $i = 1, \ldots, l$, are used instead of using ξ_i^m. Also, there are no coefficients b_i, where $i = 1, \ldots, l$. The dual problem, minimised in terms of α is:

$$W(\alpha) = \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} K_{i,j}\overline{\alpha}_i \cdot \overline{\alpha}_j + \sum_{i=1}^{l} \overline{\alpha}_i \cdot \overline{e}_i \tag{52}$$

with

$$\sum_{m=1}^{k} \alpha_i^m = 0 \quad \text{for } i = 1, \ldots, l$$

and

$$\begin{aligned} \alpha_i^m &\leq 0 \quad \text{if } y \neq m \\ \alpha_i^m &\leq C \quad \text{if } y = m \end{aligned}$$

for$i = 1, \ldots, l$and $m = 1, \ldots k$, where $K_{i,j} \equiv \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$. Then

$$\mathbf{w}_m = \sum_{i=1}^{l} \alpha_i^m \phi(\mathbf{x}_i)$$

The dual objective function can be written as

$$W(\alpha) = \frac{1}{2}\alpha \cdot (K \otimes I)\alpha + e \cdot \alpha$$

Stellenbosch University http://scholar.sun.ac.za

where$\otimes$ is the Kronecker product and I is a $k \times k$ identity matrix. The decision function is now

$$\arg\max_{m}[\sum_{i=1}^{l} \alpha_i^m K(\mathbf{x}_i, \mathbf{x})]$$

Comparison of Techniques

Hsu and Lin [40] compare the binary methods of one-against-all, one-against-one and DAGSVM with the "all together" methods of [108] and [115] and the method by Crammer and Singer, and found that no one method was statistically better then the others. With respect to training time, the one-against-one and DAG methods are the best, while DAG slightly outperformed the one-against-one method with respect to testing time.

3.4.5 SVMs for Image Processing

Support Vector Machines have been successfully used in both detection and recognition of images and image components.

In [71, 72], SVMs were used to detect vertically orientated and unoccluded frontal views of human faces in grey level images. The system works by testing candidate image regions for local patterns that resemble faces; thus only binary SVMs were used to classify the two categories: faces or non-faces. The system achieved 97.12% and 74.19% on two separate test sets respectively, and performed comparably with the system described in [99] and [98].

Phillips [75] presented a system for face recognition using SVMs with a radial basis function (RBF) kernel. The SVM-based algorithm was demonstrated on both verification and identification applications. In verification or authentication, the system is presented with an image and a claimed identity for that image. The system either accepts or rejects the claim. In identification, the system is presented with an image of an unknown person and then has to estimate the identity of the person from a database of known individuals. For comparison, a principal component analysis (PCA) based algorithm was used. Experiments showed that the SVM-based approach outperformed the PCA-based approach significantly. For verification, the SVM reported an error rate of 7% as opposed to the 13% of the PCA algorithm. For identification, the SVM error rate of 23% was half that of the PCA error rate of 46%.

Pontil and Verri [78] made use of SVMs for the recognition of three dimensional objects. A system

Stellenbosch University http://scholar.sun.ac.za

was proposed whereby colour images were transformed into grey level images, and then the resolution was reduced. To compare the system, perceptrons were used, and the SVMs outperformed the perceptrons by as much as 29.3%. The SVM approach was able to classify the transformed images with an error rate of 0.03%, while it was able to classify noise corrupted images with a maximum error rate of 11%. It was found that the SVMs performed better on noise corrupted images with higher resolutions. Shifting the images horizontally with a wrap around effect produced a maximum error rate of 18.6% for the maximum shift. Finally, the system was tested against occlusions where regions of the image were randomly selected and replaced with random pixel values. A minimum error rate of 12.7% was achieved for the largest region. The performance of this system were also compared with those reported in [66] using a system based on parametric eigenspaces and compared favourably.

In [67], a system is presented for the recognition of people and estimation of pose using SVMs. Two similar techniques were used for the construction of multi-class classifiers based on a combination of biclass SVMs: the top-down decision graph proposed in [77] and the bottom-up decision tree described in [78]. As a means of comparison, a *k*-nearest neighbour classifier was used. The two SVM approaches achieved similar performance, both achieving accuracy rates of 99.5% for the recognition of individuals, and 84.5% and 84.3%, respectively, for the pose estimation as opposed to the *k*-nearest neighbour which achieved maximum accuracy rates 94.7% and 82.7% for people recognition and pose estimation respectively.

Tian, Hong and Huang [101] use SVMs to update the weights of preference for relevant images for content-based image retrieval (CBIR). In traditional CBIR, the user selects the most relevant image and provides a weight of preference for each relevant image. The system captures the user's perception, subjectivity and high-level query by dynamically updating low-level feature weights based on the user's feedback. In MARS [87], only positive examples or relevant images were considered. The SVM approach used both positive and negative examples, and the SVM was used to assign preferences to relevant images. The SVM approach was compared with relevance feedback, and in general, performed marginally better.

In [11], SVMs were used to classify high dimension image histograms using heavy tailed RBF kernels of the form defined in Eq. (43) and Eq. (44). These kernels were shown to outperform traditional polynomial or Gaussian RBF kernels. It was also observed that remapping of the input can improve the performance of linear SVMs to such an extent that it makes them a viable alternative to RBF kernels. Two image databases were used, and a maximum classification accuracy of 11% and 16.3% respectively was achieved.

Stellenbosch University http://scholar.sun.ac.za

3.5 Summary

In this chapter we have presented and discussed two traditional machine learning techniques, namely artificial neural networks and Bayesian learning. We discussed the nature of each technique's algorithm and implementation, and presented several examples where each technique has been successfully used in the classification of an image related problems. We also presented and discussed the SVM algorithm in great detail, and discussed techniques presented for the improvement and modification of the basic SVM algorithm to handle multi-class problems. A technique for the transformation of the real-valued outputs of the SVM to probabilistic values was also presented. Finally, several successful examples using SVMs as classifiers were presented showing the versatility and classification accuracy of SVMs.

Stellenbosch University http://scholar.sun.ac.za

Chapter 4

Automatic Detection of Image Orientation

4.1 Introduction

This chapter presents a system for the automatic detection of single image orientation in two orientations; images being normally orientated or images rotated by 180°. The image database and its contents are described and explained, as well as the feature extraction algorithm. Some kernel specific issues are discussed, and comparisons between the performance of SVMs, Bayesian classifiers and neural networks are presented. The use of multi-class SVMs with various kernels for the classification of images in all four orientations is also discussed.

4.2 Image Database

The image database contained 5229 images of various dimensions. The images were taken from the Corel Stock Photo Library, and included some of the following groups : *Sunrises*, *Deserts*, *Aviation*, *Glaciers*, *Lions* and *Landscapes*. A selection of images from the database can be seen in Fig. 3. All the images in the database consisted of unposed, natural scenes, with few close-up images, low contrast images or images of uniform texture. Close-up images offer very little contextual and functional information, while the extraction of the mean and variance from images of low contrast and/or uniform texture yield very similar values for each local region. We thus preferred images that contained information about the surroundings of the subject in order to provide the necessary contextual and functional information.

Stellenbosch University http://scholar.sun.ac.za

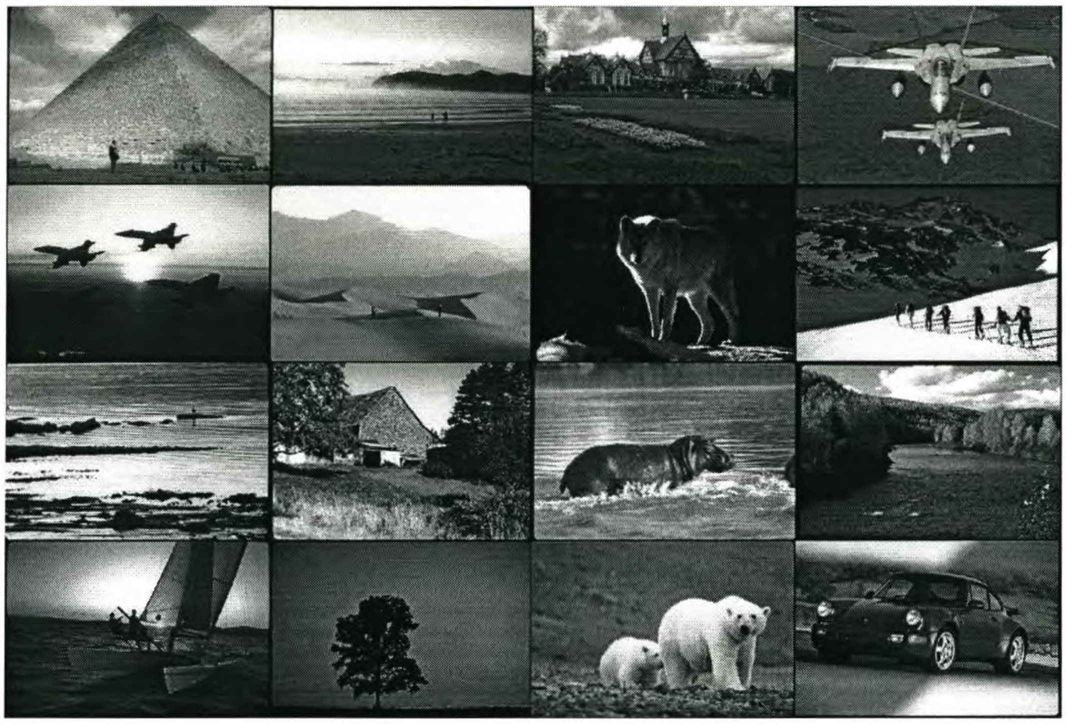


Figure 3: A selection of images within the database.

4.3 Image Features

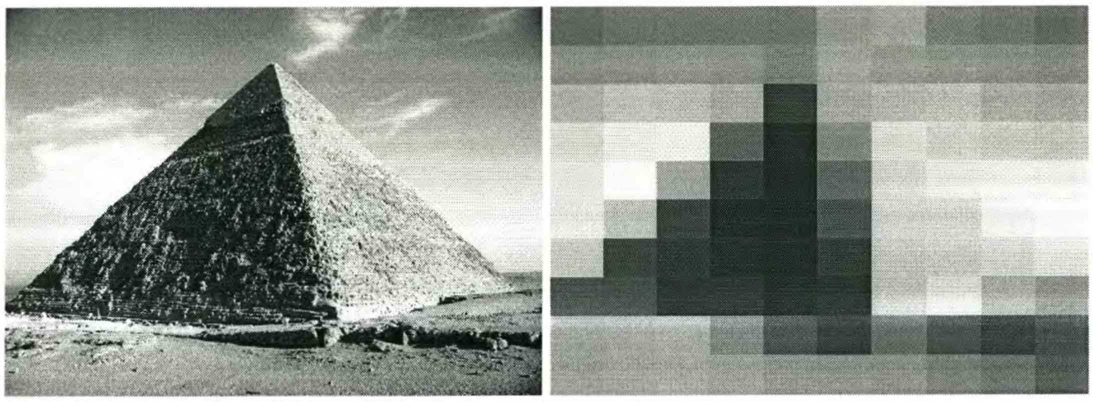
4.3.1 Low Level Image Features

Global image features are invariant to image rotation; thus, local regions within the images were selected, and low-level features extracted. The extraction of low-level features is a simple and fast way to extract the required information from images, while not being excessively CPU intensive. Another reason for the use of low-level features is the vast diversity in content the images can contain, i.e. no common object information exists in all the images. Of the low-level features discussed in Section 2.2, colour features were selected over the other features because all the other features require some information about the image, or certain elements have to be present within the image. These features are more suited to the tasks of identification and recognition.

To extract the low-level colour features, each image was divided into $N \times N$ blocks; from each block, the mean and variance of RGB pixel values were calculated. The image is now represented by several regions of mean and variant colour; high resolution images are not needed to improve performance. An example of this feature extraction is shown in Fig. 4. Thus, $6N^2$ features were extracted for each

Stellenbosch University http://scholar.sun.ac.za

image, where $N \in \{6, 8, 10, 12, 15, 17, 20\}$.



(a) Original Image. (b) Average of local regions for N = 10

Figure 4: The representation of the image in terms of the average of each local region.

4.3.2 Number of Local Regions

As mentioned in Section 4.3.1, the number of local regions in an image influences both the classification performance and the processing time. The advantage of using a larger number of regions to represent an image is the slight improvement in classification of test images, and the significant improvement in classification of the training set (see Table 2). However, this advantage is offset by the effect that an increase in the number of local regions has on the processing time and is particularly noticeable for SVMs that use RBF kernels (see Table 1).

Table 1: The number of local regions per image, the number of features produced, and classification time when classified with a Gaussian kernel.

N	Number of features	Time (sec)
6	216	0.107
8	384	0.209
10	600	0.332
15	1350	0.743
20	2400	1.366

Stellenbosch University http://scholar.sun.ac.za

Table 2: Correct classification performance of SVMs of individual images with linear kernel on training and test set as a function of the number of local regions.

N^2	Test Set (%)	Training Set (%)
6	77.44	80.10
8	77.86	81.59
10	78.24	82.24
12	77.97	83.70
15	78.28	84.27
17	77.82	85.46
20	78.09	86.53

4.4 SVM Kernel Particulars

4.4.1 Kernel Selection

Currently, no techniques for learning the form of a kernel are known, and thus the kernels investigated here are standard kernels already used in pattern recognition. These kernel forms were discussed in Section 3.4.2. Due to the success of the RBF kernels in other pattern recognition tasks, RBF kernels were applied to the problem of automatic detection of image orientation. A set of experiments were conducted to compare the performance of the various RBF kernels against that of the linear kernel. Input mapping, whereby the input to the SVM is mapped non-linearly to a higher dimensional space as discussed in Section 3.4.2, was not performed on the input data for these experiments, as we wanted to compare the performance of the kernels without other influencing factors. We observe that none of the SVMs with RBF kernels perform as well on the test set as the SVMs with linear kernels.

Table 3: Results for the various kernels as a function of the number of local regions. Note: no input remapping was done (i.e. $a = 1$)

N	Classifier			
	Linear	Gaussian	Laplacian	Sublinear
6	70.71%	65.66%	69.70%	68.69%
8	72.73%	64.65%	69.70%	69.70%
10	73.74%	60.61%	68.69%	71.72%
12	74.75%	54.55%	70.71%	71.72%
15	73.74%	49.49%	71.72%	70.71%
20	75.86%	48.48%	68.69%	71.72%

Stellenbosch University http://scholar.sun.ac.za

4.4.2 Input Remapping

The RBF kernels discussed in Section 3.4.2 allow for input to be nonlinearly remapped by choosing values for parameter a. The values selected for a fell into the range $\{0.125, 0.25, 0.5, 1.0\}$. The classification performance of the SVM with Gaussian, Laplacian and sublinear kernels are shown in Figs. 5, 6 and 7, respectively, for each value of a.

We observe that the classification performance of the SVMs with a Gaussian kernel decreases as the number of local regions increases. Furthermore, the particular value of a plays a diminishing role as the number of regions increases; for $N = 20$ the performance of all SVMs with Gaussian kernels and remapping is the same. Interestingly, SVMs trained with Laplacian and sublinear kernels do not show this behaviour. Their classification performance depends far less on the particular remapping used. Amongst the SVMs trained with RBF kernels, the sublinear kernel shows the best classification performance; its performance is also the most robust with respect to the number of local regions.

4.5 Optimal Bayesian Classifier

The performance of the Bayesian classifier is comparable to the performance of SVMs with linear kernels, although the RBF kernels consistently outperform the Bayesian classifier (see Fig. 8). The advantage of the Bayesian classifier is that both training and classification times are a fair amount quicker than those of the SVM.

4.6 Neural Network Classifier

The performance of SVMs trained with linear and RBF kernels was compared to the performance of a multilayer perceptron. The multilayer perceptron consisted of an output layer of one neuron, and a hidden layer of 30 neurons. The multilayer perceptron was then trained on the same images as the SVMs were trained on. The performance of the multilayer perceptron does not reach the performance level of SVMs trained with either Laplacian or sublinear kernels; furthermore, training multilayer perceptrons takes substantially more time than training SVMs. However, multilayer perceptrons do outperform SVMs with either linear or Gaussian kernels (see Fig 8).

Stellenbosch University http://scholar.sun.ac.za

4.7 Support Vector Machines

The architecture of a SVM is relatively simple and mainly requires the choice of the kernel, and the selection of parameter C, the penalty term for misclassifications. With all the RBF kernels, a value of 300 for C was significantly large enough to enforce full separability, meaning that the SVM correctly classifies all the data in the training set. For the linear kernel, the choice of C is dependent on the renormalisation factor used to normalise the data for both the training and the test set. This value for C was found to be 10^{-2}.

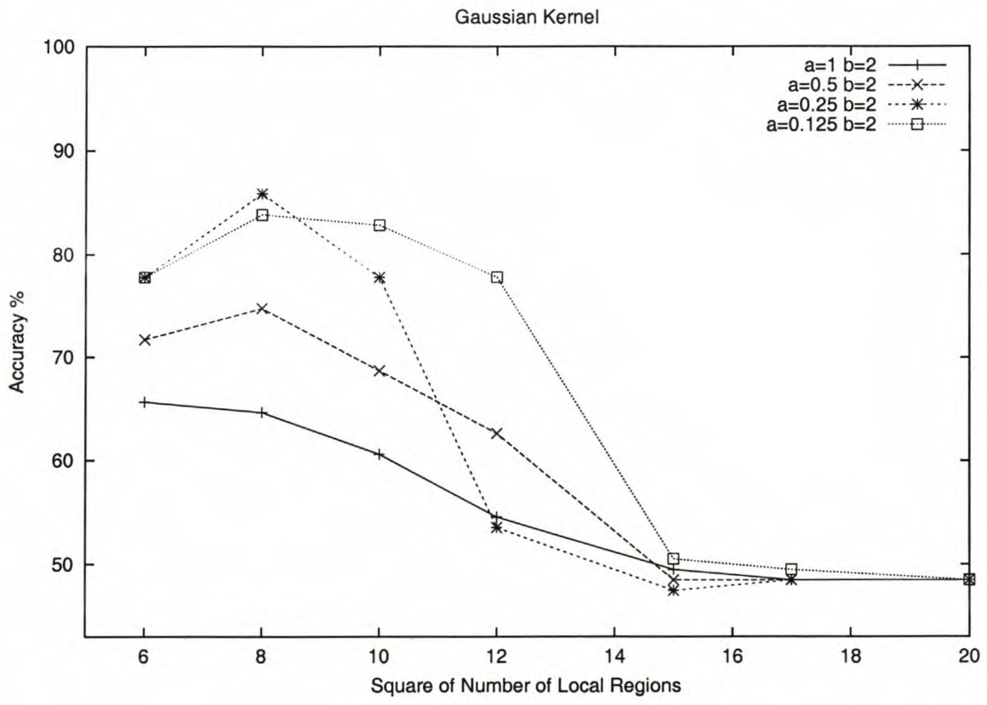


Figure 5: Accuracy of the Gaussian kernel as a function of the number N of local regions for $a = 1, a = 0.5, a = 0.25, a = 0.125$.

As Fig. 5 shows, SVMs with Gaussian RBF kernels perform well when the image consists of a small number of regions. As soon as the number of regions per images are increased, the performance of the SVM drops away rapidly towards 50%. By applying input remapping, this performance drop off can be delayed slightly. The reason for this sudden decrease in performance is not known, but we believe

Stellenbosch University http://scholar.sun.ac.za

that it could be similar to the neural network property of over-fitting, whereby the training patterns are learnt instead of the underlying relationship between each input. There is also quite a significant difference in the performance of the SVM with a Gaussian kernel for different values of a. As a is decreased, the SVM produces better performance, and is more tolerable to an increase in the number of regions.

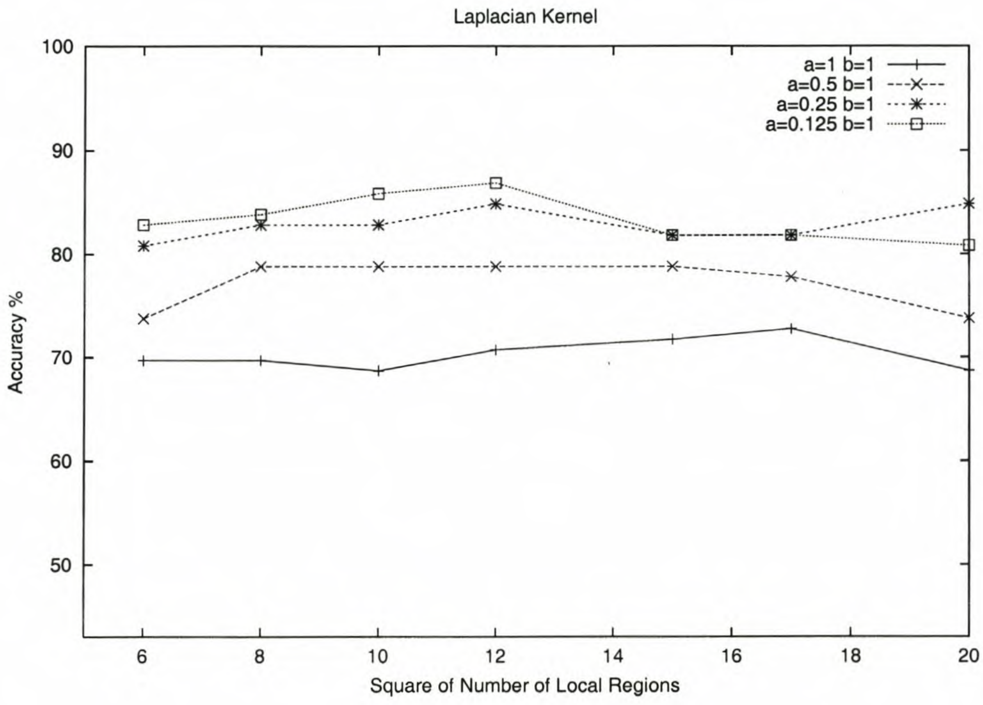


Figure 6: Accuracy of the Laplacian kernel as a function of the number N of local regions for $a = 1, a = 0.5, a = 0.25, a = 0.125$.

Support vector machines with Laplacian kernels produce far more predictable results over the varying number of image regions, as is seen in Fig. 6. Although for each value of a there is an optimal number of regions, the difference in the performance over the varying regions is significantly smaller compared to SVMs with the Gaussian kernel. As a is decreased, the performance of the SVM is increased significantly.

Sublinear kernels produce very similar results when compared to the Laplacian kernel (see Fig.7). The only noticeable difference is that the sublinear kernel reaches its optimal performance level for

Stellenbosch University http://scholar.sun.ac.za

higher values of the parameter a. Also, the sublinear kernel is slightly more stable in its classification performance over the change in number of regions than the Laplacian kernel.

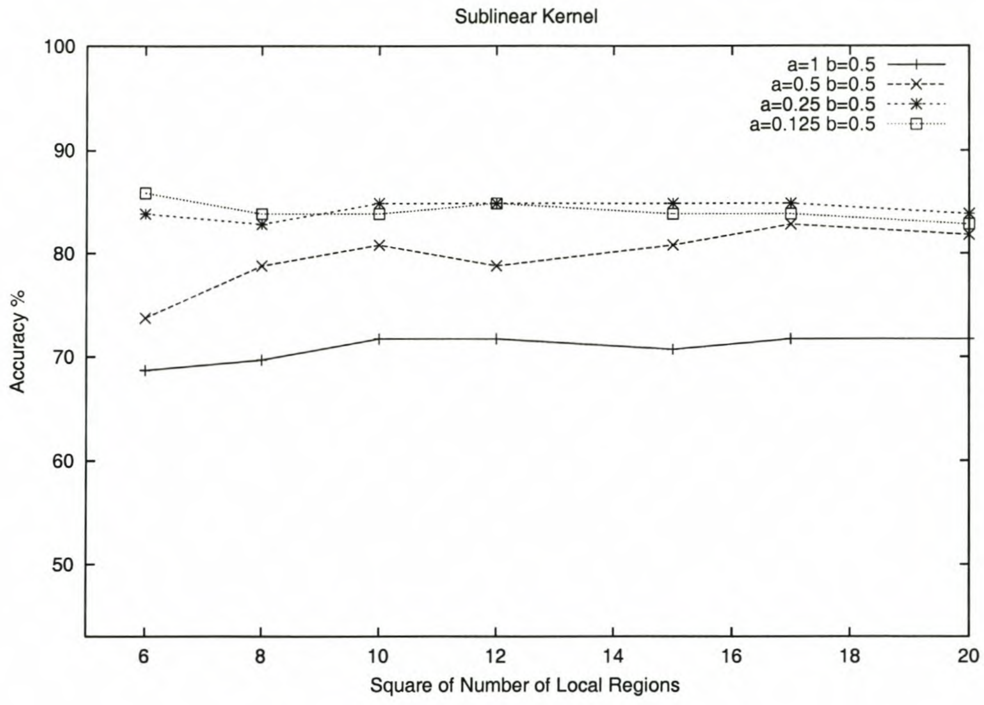


Figure 7: Accuracy of the sublinear kernel as a function of the number N of local regions for $a = 1, a = 0.5, a = 0.25, a = 0.125$.

4.8 Performance Comparison

As already mentioned, SVMs with RBF kernels outperform other traditional pattern recognition techniques, including multilayer perceptrons and optimal Bayesian classifiers. With the exception of the Gaussian kernel, both the sublinear and Laplacian kernels as well as the linear kernel perform comparatively if not better than the other two classifiers (see Fig. 8). An added advantage of SVMs over multilayer perceptrons is that the training time for a SVM is considerably faster than that of a neural network.

Stellenbosch University http://scholar.sun.ac.za

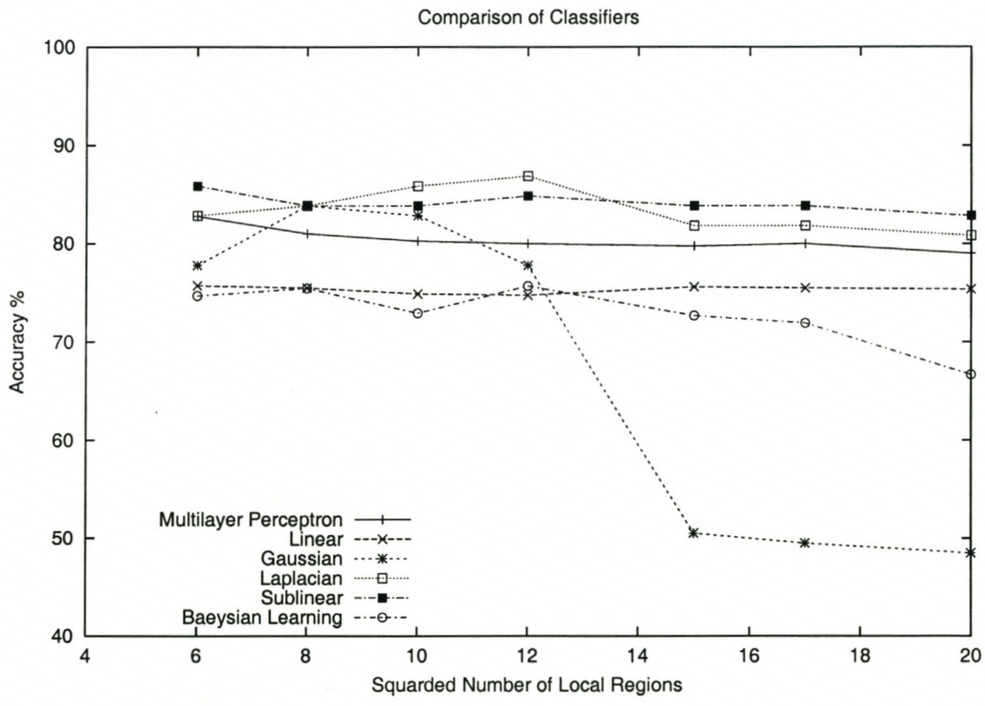


Figure 8: A comparison of the various classifiers

Stellenbosch University http://scholar.sun.ac.za

4.9 Multi-Class Orientation Detection

The results discussed so far only deal with images in two orientations, namely 0° and 180°. The other two orientations, namely 90° and 270°, were then added to the training and test sets. The same image database where the images were randomly rotated through 0°, 90°, 180°, and 270°. The multi-class SVM technique of one-against-all (Section 3.4.4) was then used to train a SVM. Only the linear and Gaussian kernels were investigated, and because of the results achieved, several issues had to be addressed before using the Laplacian and sub-linear kernels. The results, shown in Figs. 9 and 10, were obtained through ten-fold cross-validation, and the optimal value for c, the penalty parameter for misclassifications, was investigated. Our results did not measure up to the successes of the binary image orientation classification. For the Gaussian kernel, as in the case of two class classification, as the number of regions increases, the classification accuracy decreases until it is classifying every image into a single class. Further work is needed to determine why this happens, and how it can be prevented.

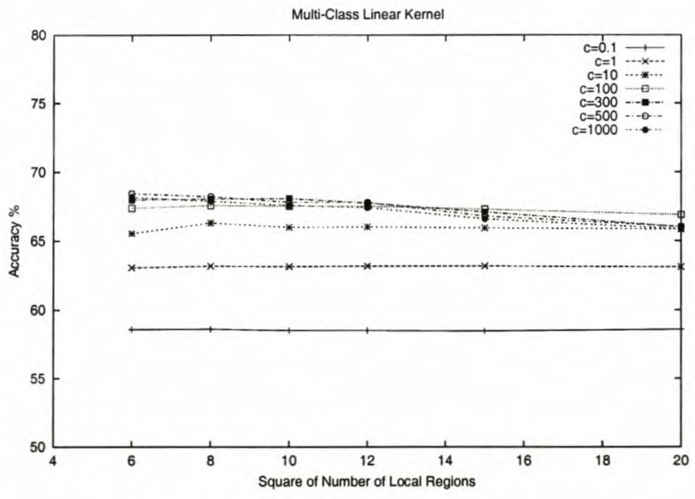


Figure 9: Multi-class classification with a linear kernel as a function of the trade-off parameter c.

4.10 Summary

This chapter has shown that SVMs for the detection of image orientation can be used to replace traditional classifiers such as Bayesian learning and artificial neural networks. Our results show that SVMs performed as well, if not better, when compared to neural networks and Bayesian learning. The advantage of SVMs over neural networks is the increased speed in training time, as well as the as

Stellenbosch University http://scholar.sun.ac.za

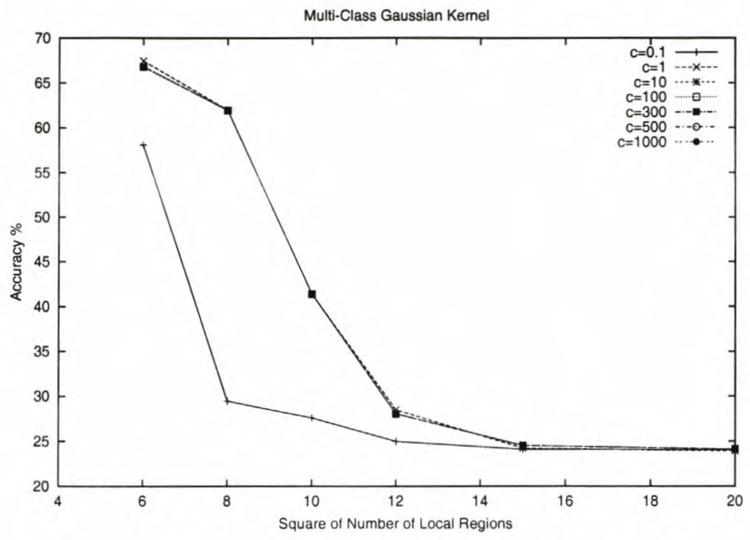


Figure 10: Multi-class classification with a Gaussian kernel as a function of the trade-off parameter c.

classification performance. The training time for SVMs is comparable to that of Bayesian learning, while the classification accuracy is the same, if not better, depending on the kernel used in the SVM.

We also investigated the effect that the number of regions has on classification accuracy. The increase in the number of regions only produced an increase in the classification accuracy when using the linear kernel, while the classification accuracy of the Gaussian kernel deteriorated as the number of regions was increased. Both the Laplacian and sublinear kernels showed very little difference in classification accuracy as the number of regions was increased. An obvious side effect to the increase in the number of regions was the increase in the classification time, and this was particularly noticeable in the RBF kernels. We can conclude that when using RBF kernels the optimal number of regions for good classification accuracy and classification speed should be kept rather low, while when using linear kernels the performance improves as the regions are increased, with a linear increase in classification time.

The initial results are encouraging with highly competitive classification performances for the two class problem, although a change in methodology is needed to produce similar results for multi-class classification that are competitive with the Bayesian learning technique [106].

Stellenbosch University http://scholar.sun.ac.za

Chapter 5

Colour Management and Image Workflow

5.1 Introduction

Colour management and image workflow are now considered synonymous with each other. Where there is an image workflow system in place, a colour management system (CMS) will have been implemented. Image workflow systems have been around for a long time; a photographic laboratory is a good example of an image workflow system. Images are transferred from negatives to photographic paper, and to ensure the colour is accurately transferred, a colour management system monitors the chemicals involved in the development of the images. Similarly, in a digital workflow system, where images are transfered from negatives to a digital format, the colour management system ensures that the colour reproduction quality is maintained.

To implement the research of Chapter 4 in a real world application, we turned to the process of transferring images from film negatives to a digital format via a negative scanner. We wanted to investigate if our work could efficiently determine the orientation of an entire film negative with high accuracy.

This chapter will describe the purpose of a colour management system, as well as the unique problem of image orientation detection in a negative scanning image workflow system. We will also discuss the implementation of our technique and report on its performance.

Stellenbosch University http://scholar.sun.ac.za

5.2 Colour Management and Image Workflow

Colour Management

A Colour Management System (CMS) [31] is the calibration of input devices with output devices to produce consistent, predictable colour. Examples of input devices are scanners and digital cameras. Output devices include monitors, desktop printers, film recorders, and even the printed page. Colour management involves specialized software that communicates the calibration of each device to the computer which then makes adjustments to match the colour gamut (palette or range of colors a device can produce) of one device to another. The aim of color management is WYSIWYG (what you see is what you get) across any number of devices.

The need for colour management arises because different devices describe colours in different ways. For example, printers represent colour in the CMYK (Cyan Magenta Yellow blacK) colour space while monitors represent colour in the RGB colour space. The Internet allows the sharing of images between a vast number of people using a wide variety of hardware devices for viewing and reproduction; thus, it was necessary to develop a system that can control the way different devices interpret and manipulate colours. In order to do this, the device-dependent colour representations of the colour imaging devices are linked to a device independent colour space, and through calibration, a mapping can be generated for each device. This then produces an image that is device independent, which allows for the exchange of images that conform to an international standard.

Image reproduction devices can be classified into two categories: additive and subtractive devices. Additive devices produce colour by adding different combinations of the primary colours, i.e. red, green and blue. An example of an additive colour space is the RGB colour space, and an example of an additive device is a computer monitor. Subtractive devices produce colour by multiplying a white spectrum by the transmission curves of the three subtractive primary colours, namely cyan, magenta and yellow. This means that for each subtractive primary, frequency components are subtracted from the white spectrum. An example of a subtractive colour space is the CMYK colour space, and an example of a subtractive device is a printer.

A colour management system (CMS) has two main tasks in order to produce reliable colour reproduction: First, colourmetric characterization of the peripherals is required so that device-dependent colour representations can be linked to a device-independent colour space. This is done by the creation of a profile for each device which maps the device-dependent colour space to the device-independent colour space. Secondly, the CMS provides an efficient means for processing images and converting them from one device-dependent colour space to the next.

Stellenbosch University http://scholar.sun.ac.za

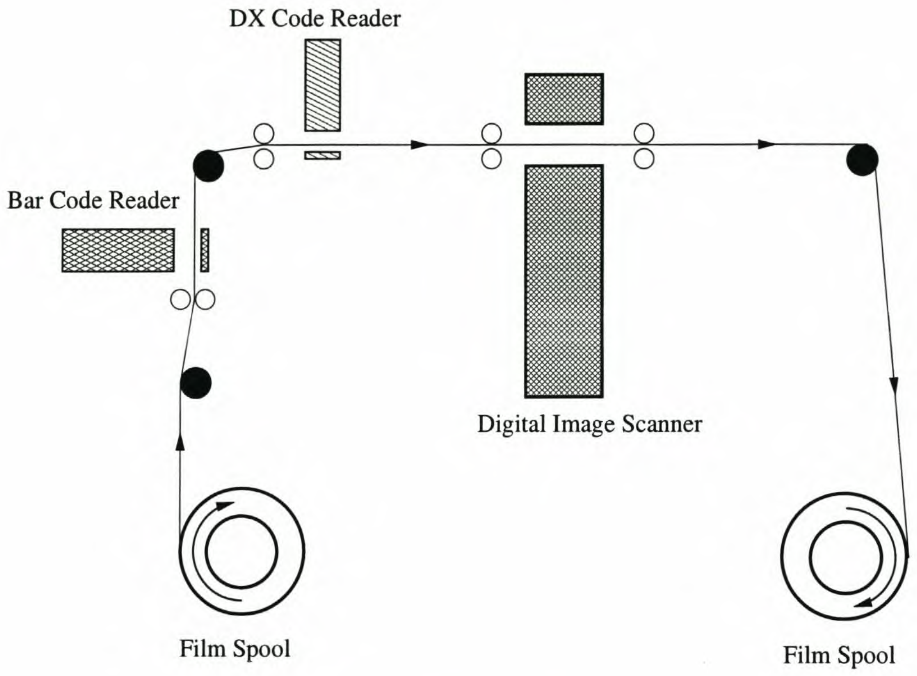


Figure 11: Schematic representation of a scanner.

Image Workflow

In the transfer of images from film negatives to a digital storage device, it is crucial that colour management be applied. In this specific application, the images are transformed from a negative-dependent colour space to a device-independent colour space. This is necessary because the different chemical properties of each film type that produce slight differences in colour. In order to map the images from the various negative-dependent colour spaces to a device-independent colour space, each negative is profiled. Once in the the device-independent colour space, the images are ready to be distributed in various forms to the end user. The end user will then calibrate his equipment and produce profiles to convert the images back into device-dependent colour spaces specific to his hardware configuration.[1]

5.3 System Description

In the transfer of images from film to digital format, the developed film negative is mounted on a spool, which is later fed into the digital image scanner (Fig. 11). Before the film can be scanned, it is

[1]This entire process can be done with commercial products such as GretagMacbeth's *ProfileMaker* and Monaco Systems' *MonacoPROOF*.

Stellenbosch University http://scholar.sun.ac.za

passed through two bar code readers, which retrieve information about the film. The first reader scans a bar code that is located at the front of the film and contains information pertaining to the customer such as batch number and film number. The second reader scans the DX code (Fig. 12) that is located on the top edge of the film; it identifies the type and make of the film, e.g. Kodak Gold 200. Because the DX code is only located on one edge of the film, the film has to be correctly orientated when fed into the scanner so that the DX code reader can read it.

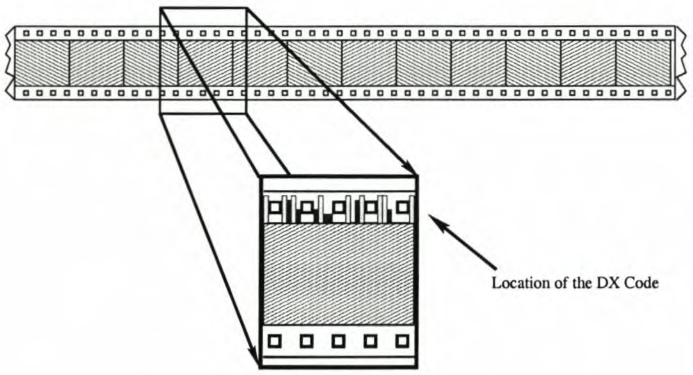


Figure 12: The DX Code location on a film negative.

There would be no need to determine the orientation of a film if all cameras fed the film in the same direction; but this is not the case. A spool of film can either be placed on the left hand side of the camera and fed towards the right, or placed on the right hand side and fed towards the left (see Fig. 13). In order to do this, the spool has to be inserted "upside-down"; thus an entire film may appear incorrectly orientated once the images have been digitally scanned.

The system mentioned above was simulated by training a SVM on a training set as in Chapter 4. The remaining images in the test set were then put into groups of 36 images, according to their orientation, to represent a complete film (Fig. 14).

Once the SVM has been trained on the training set, the film of images is then classified. The output f of the SVM for each image derived in Eq. (15) is then passed through a sigmoidal function

$$P(y = 1|g) = \frac{1}{1 + e^{(Ag+B)}}$$

proposed in [76] and discussed in Section 3.4.3.

The further away an input vector is from the separating hyperplane, the higher the probability is that it has been correctly classified. A negative value means that the image is classified as inverted, and a

Stellenbosch University http://scholar.sun.ac.za

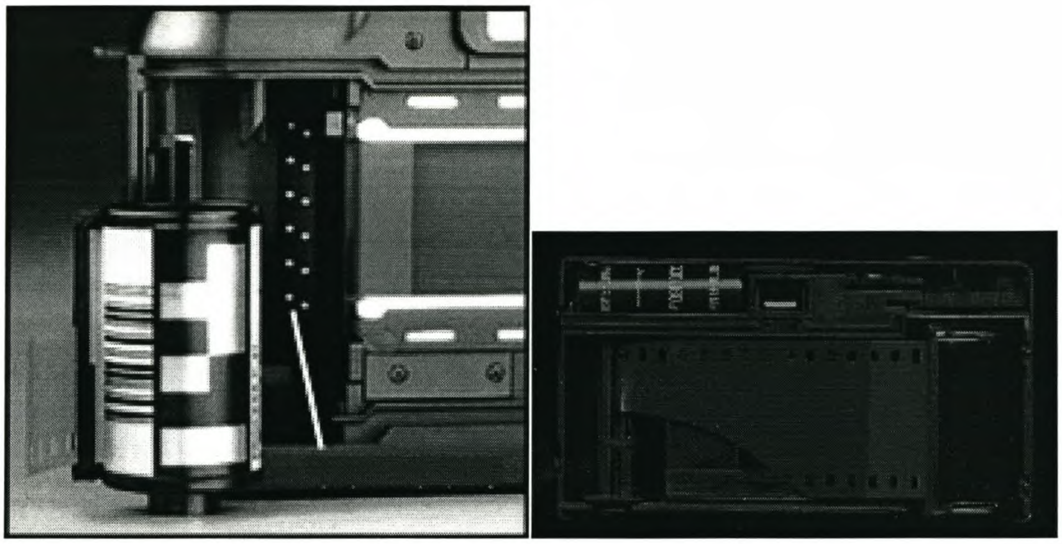

(a) Left to right (b) Right to left

Figure 13: Film fed (a) from left to right, and (b) right to left.

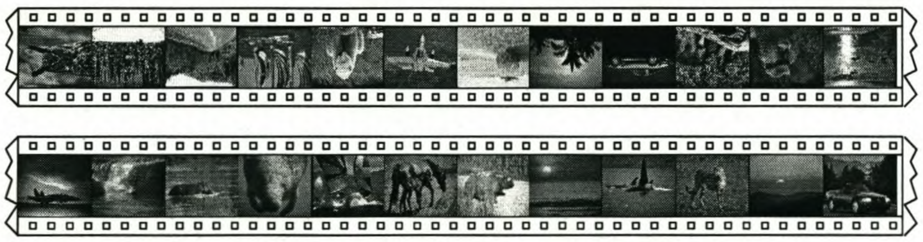


Figure 14: Grouping of images to form an inverted and correctly orientated film.

Stellenbosch University http://scholar.sun.ac.za

positive value means that the image is classified as correctly orientated.

The equations

$$P_{0^\circ} = \log(P_{0^\circ}1) + \cdots + \log(P_{0^\circ}n) + \log(1 - P_{180^\circ}1) + \cdots + \log(1 - P_{180^\circ}n) \qquad (53)$$

$$P_{180^\circ} = \log(P_{180^\circ}1) + \cdots + \log(P_{180^\circ}n) + \log(1 - P_{0^\circ}1) + \cdots + \log(1 - P_{0^\circ}n) \qquad (54)$$

with $0 \leq P_{180^\circ}n \leq 1$ and $0 \leq P_{0^\circ}n \leq 1$, are then defined to determine the most likely orientation of an entire film. All the positive probabilities for the entire film are then logarithmically added together (Eq. 53), as are all the negative probabilities (Eq. 54), and the larger of the two numbers is used to determine the orientation of the film. We assume that the orientation of an image is independant of the orientation of all other images on the film.

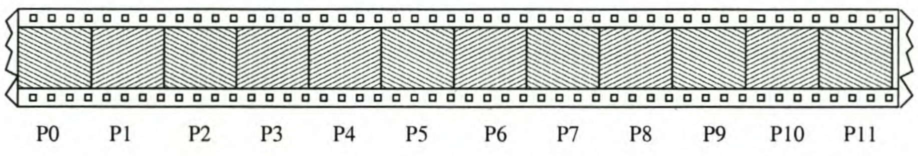


Figure 15: Probabilities calculated for each image on the film.

5.4 Performance Evaluation

All the kernels mentioned previously were used, and all except the Gaussian RBF kernel obtained a 100% classification accuracy. The Gaussian kernel suffered from over-generalisation, just as in Chapter 4, as the number of local regions per image was increased. Table 4 shows the classification times of a film (36 images) for the various kernels. As is evident, the linear kernel considerably outperforms the RBF kernels with respect to time, and this difference in classification time is more noticeable as the number of local regions is increased.

Stellenbosch University http://scholar.sun.ac.za

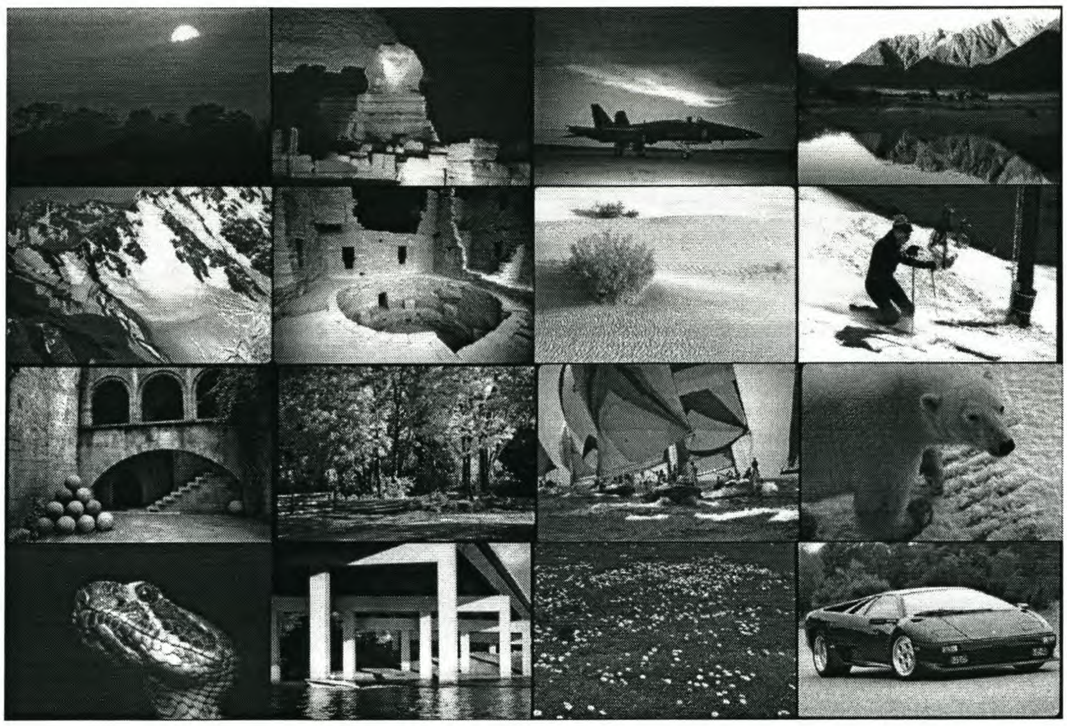


Figure 16: A selection of images incorrectly classified.

Table 4: Classification time (secs) of an entire film using different kernels as a function of the number of local regions.

N^2	Classifier			
	Linear	Gaussian	Laplacian	Sublinear
6	<1	19	23	17
8	<1	37	44	30
10	<1	59	54	48
12	<1	85	106	71
15	<1	135	137	114
17	<1	170	171	147
20	1	235	241	208

Stellenbosch University http://scholar.sun.ac.za

5.5 Summary

In this chapter, we have shown that SVMs with probabilistic outputs can be effectively used to provide fast and efficient solutions in the orientation detection of entire films with an accuracy of up to 100% [110]. Classification times can be kept to under a second for the classification of 36 images per film when using the linear kernel. Currently, image workflow systems use a human operator to manually analyse and rotate entire films, depending on the orientation of the film. By incorporating our system, the overall processing time of an image workflow system could be considerably reduced, resulting in increased productivity and freeing the operator to perform more important tasks.

Stellenbosch University http://scholar.sun.ac.za

Chapter 6

Conclusions and Directions for Future Research

6.1 Conclusions

In general, the research reported here produced the results that we expected. We have shown that the use of SVMs as a means of classification is a viable alternative to both Bayesian learning and artificial neural networks in determining the orientation of an image. Support vector machines train quickly and generalise well to unseen data, as is evident from the maximum classification accuracy of 87% for two-class classification with the Laplacian kernel. The selection of the kernel also plays an important role in both the classification accuracy and training time; the more complex RBF kernels take longer to train, but also produce better results. The use of input mapping further improves the classification accuracy of the RBF kernels.

The use of SVMs for multi-class classification produces reasonable results when using the linear kernel, although the same cannot be said for the Gaussian kernel. Several improvements are required here to produce results that are comparative with Bayesian learning and artificial neural networks. Increasing the size of the image database is probably the easiest improvement to implement, and should increase the classification accuracy quite significantly. Improving the choice of features extracted from the images should also produce better results, along with implementing some pre-processing technique that can select those features that provide information during the classification process.

The implementation of a system for the correct classification of film orientation produced results that were better than expected. All kernels except for the Gaussian kernel achieved 100% classification accuracy when coupled with a probabilistic output generator. This system produces fast and accurate

Stellenbosch University http://scholar.sun.ac.za

real time classification of film orientation, and could prove to be a useful workflow tool in the photo-finishing industry.

6.2 Directions for Future Research

We have identified several areas from our current research that raise questions and warrant further questions.

6.2.1 Low-Level Features for Classification

Although the use of the low level features of mean and variance provided satisfactory results, the extraction of features that contain more semantic information about the image and the objects there in would improve classification performance. Shape extraction from within an image could also provide information about the relationships within the image, as can texture. The use of edge detection could also play a role in determining the orientation of images. The use of these low level features could provide enough significant features so as to improve classification.

Another issue that begs further research is the selection of the colour space, and how it impacts the classification performance. Several colour spaces were created to provide more information about illumination, hue, saturation, luminocity, chromaticity, and contrast about colour, along with several device-dependent and device-independent colour spaces. By using different colour spaces, different information could be extracted through the use of low-level features that might separate the classes more effectively. Another way to extract more information through low-level feature extraction could be the use of colour spaces of greater depth. Traditionally, colour has been described in either 16 or 24 bits, although colour can be described in depths of 32 or even 64 bits. Although the increase in colour space depth would incur an increase in complexity, perhaps more information from within the image could be extracted through low-level feature extraction.

6.2.2 Pre-Processing

Although the use of features extracted directly from the images provided reasonable classification accuracies, perhaps by making use of various pre-processing techniques apart from the low level feature extraction, such as Kohonen's Self-Organising Maps (SOMs) [47, 48] and Learning Vector Quantisation (LVQ), Principal Component Analysis (PCA), Gaussian Mixture Model (GMM) and GMM Estimation Maximisation (GMM/EM) could further increase the classification performance and

Stellenbosch University http://scholar.sun.ac.za

accuracy by reducing the size of the feature vector and selecting those features that provide sufficient information on class separation.

It was also shown in [53] that the use of SOMs and PCA for dimension reduction can significantly improve the classification accuracy of a face recognition system. In this example, SOMs outperform PCA by as much as 5% when used for feature detection and dimension reduction in the pre-processing step of face recognition using a convoluted neural network as a classifier.

6.2.3 Image Exposure Level Detection and Correction

The successful application of SVMs in a workflow tool for the detection of film orientation gave rise to several other areas within the photo-finishing industry where they might be successfully applied. The first is the successful detection and correction of the exposure level of an image. In order to accomplish this, low level features will need to be extracted, along with information about the relationships between pixels. This information could be in the form of histograms or other low level methods, such as the use of intensity, saturation, hue, and illumination. Such a system could improve the quality of the image delivered to the customer, and make any other operations on the image easier since the correct exposure level would convey the relationship between pixels and objects within the image.

The second area that SVMs could be applied to is the successful analysis of an image in order to apply the correct profile to it when using a colour management system. This is very similar to the exposure detection problem, and a similar approach can be adopted. The relationship between colours would have to be investigated, and the SVM would then determine the colour space that the image is in, and which profile must be applied to the image to convert it to the current colour space. This technique could also be used to neutralise the effect that different film types have on the resultant image by converting images from all film types to a film-independent colour space, from where further operations can be performed on the image.

6.2.4 Knowledge Extraction

Knowledge extraction is the process of determining what rules and inferences a system makes on the data it is trained on. In [102], several conclusions about rule extraction on neural networks were derived. These include the conclusion that the rules extracted from neural networks can closely reproduce the accuracy of the original network, that the rules extracted are superior to the rules that are arrived at by methods that directly refine symbolic rules, and that the rules are understandable by humans.

Stellenbosch University http://scholar.sun.ac.za

Unfortunately, such techniques do not yet exist for SVMs. The extraction of knowledge from an SVM would enable us to determine what features in the training set contained information that enabled the SVM to learn, and thus provide either more features that are similar, or reduce the size of the input vector by removing those features that do not contain any significant meaning for the SVM. Rule extraction could also produce better classification accuracies and classification performance.

6.2.5 Use of Prior Knowledge

The use of prior knowledge in the training of SVMs is closely linked with the extraction of knowledge from SVMs. Towell and Shavlik proposed a method that maps problem-specific domain theories into neural networks and then using backpropagation, refines the reformulated knowledge in [103] called Knowledge-Based Artificial Neural Networks (KBANN). Their research shows that the KBANN technique considerably outperforms various other techniques including the Nearest Neighbour technique, the Perceptron [84], ID3 [80], backpropagation [89] and Cobweb [18].

If a similar technique to the KBANN algorithm could be developed for SVMs, performance could be further improved in both classification performance and training time.

Stellenbosch University http://scholar.sun.ac.za

Bibliography

[1] M. Aizerman, E. Braverman, and L. Rozonoér. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[2] T. Bayes. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London*, 53:370–418, 1763.

[3] F. Bergholm. Edge programming. *IEEE Transaction on Pattern Analysis and Machine Inteligence*, PAMI-9(6):726–741, November 1987.

[4] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proc. 5th Annu. Workshop on Comput. Learning Theory*, pages 144–152. ACM Press, New York, NY, 1992.

[5] H. Bourlard and N. Morgan. *Connectionist Speech Recognition. A Hybrid Approach.* Kluwer Academic Publishers, Boston MA, 1994.

[6] S. Brandt, J. Laaksonen, and E. Oja. Statistical shape features in content-based image retrieval. In *Proceedings of 15th International Conference on Pattern Recognition*, volume 2, pages 1066–1069, Barcelona, Spain, September 2000.

[7] C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[8] N. Campbell, W. Mackeown, B. Thomas, and T. Troscianko. Interpreting image databases by region classification. *Pattern Recognition (Special Edition on Image Databases)*, 30(4):555–563, 1997.

[9] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, November 1986.

[10] T. Chang and C. Kuo. Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions on Image Processing*, 2(4):429–441, October 1993.

Stellenbosch University http://scholar.sun.ac.za

[11] O. Chapelle, P. Haffner, and V. Vapnik. Svms for histogram-based image classification. *IEEE Trans. on Neural Networks*, 9, 1999.

[12] G. Cortelazzo, G. Mian, G. Vezzi, and P. Zamperoni. Trademark shapes description by string-matching techniques. *Pattern Recognition*, 27(8):1005–1018, August 1994.

[13] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273, 1995.

[14] K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. In *Computational Learning Theory*, pages 35–46, 2000.

[15] G. Cross and A. Jain. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(1):25–39, 1983.

[16] L. Davis. Shape matching using relaxation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(1):60–72.

[17] L. Davis, S. Johns, and J. Aggarwal.

[18] D.Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172, 1987.

[19] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, New York, 1973.

[20] R. Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, Inc., New York, 2nd edition, 1987.

[21] W. Forstner. A framework for low level feature extraction. In J. Eklundh, editor, *Computer Vision - ECCV*, volume 2, pages 383–394. Springer, 1994.

[22] J. Friedman. Another approach to polychotomous classication. Technical report, Stanford Department of Statistics, 1996.

[23] K. Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–130, 1988.

[24] K. Fukushima and S. Miyake. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, 15(6):455–469, 1982.

[25] B. Furht, S. Smoliar, and H. Zhang. *Image and video indexing and retrieval techniques*. Kluwer Academic Publishers, 1995.

Stellenbosch University http://scholar.sun.ac.za

[26] J. Garding and T. Lindeberg. Direct computation of shape cues based on scale-adapted spatial derivative operators. *International Journal of Computer Vision*, 17(2):163–191, 1996.

[27] P. Gill, W. Murray, and M. Wright. *Practical Optimization.* Academic Press, Boston, MA, USA, 1981.

[28] J. Hampshire and B. Perlmutter. Equivalence proofs for multilayer perceptron classifiers and the bayesian discriminant function. In D. Touretzky, J. Elman, T. Sejnowski, and G. Hinton, editors, *Proceedings of the 1990 Connectionist Models Summer School*, San Mateo, CA., 1990. Morgan Kaufmann.

[29] U. Handmann, T. Kalinke, C. Tzomakas, M. Werner, and W. von Seelen. An image processing system for driver assistance. In *IEEE International Conference on Intelligent Vehicles*, pages 481–486, Stuttgart, Germany, 1998. IEEE.

[30] R. Haralick, K. Shanmugan, and I. Dinstein. Texture features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, 3:610–621, 1973.

[31] J. Hardeberg. Color management: Principles and solutions. *NORSIGnalet, the quarterly magazine of the Norwegian Signal Processing Society*, (3), 1999.

[32] T. Hastie and R. Tibshirani. Classification by pairwise coupling. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

[33] D. Haussler, M. Kearns, and R. Schapire. Bounds on the sample complexity of bayesian learning using information theory and the vc dimension. *Machine Learning*, 14:83–113, 1994.

[34] S. Haykin. *Neural Networks - A Comprehensive Foundation.* Pretence Hall, New Jersey, 2nd edition, 1999.

[35] M. Heath, S. Sarkar, T. Sanocki, and K. Bowyer. A robust visual method for assessing the relative performance of edge detection algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-19(12):1338–1359, December 1997.

[36] J. Holland. Outline for a logical theory of adaptive systems. *Journal of the Association for Computing Machinery*, 3:297–314, 1962.

[37] J. Holland. *Adaption in Natural and Artificial Systems.* University of Michigan Press, 1975.

[38] J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Scientists*, 79:2554–2558, 1982.

Stellenbosch University http://scholar.sun.ac.za

[39] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[40] C. Hsu and C. Lin. A Comparison of Methods for Multi-class Support Vector Machines. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2001.

[41] P. Huang and Y. Jean. Using 2d c + -strings as spatial knowledge representation for image database systems. *Pattern Recognition*, 27(9):1249–1257, 1994.

[42] L. Iverson and S. Zucker. Logical/linear operators for image curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-17(10):982–996, October 1995.

[43] A. Jain, J. Mao, and K. Mohiuddin. Artificial neural networks: A tutorial. *IEEE Computer*, 29(3):31–44, 1996.

[44] R. Kashyap and A. Khotanzad. A model-based method for rotation invariant texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:472–481, 1986.

[45] J. Kindermann, E. Leopold, and G. Paass. Multi-class classification with error correcting codes.

[46] S. Knerr, L. Personnaz, and G. Dreyfus. Single-layer learning revisited: a stepwise procedure for building and training a neural network, 1990.

[47] T. Kohonen. Self-organised formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[48] T. Kohonen. *Self-Organisation and Associative Memory*. Springer-Verlag, 1984.

[49] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78:1464–1480, 1990.

[50] T. Kohonen. *Self-Organization Maps*. Springer-Verlag, Berlin, Germany, 1995.

[51] B. Kröse and P. van der Smagt. *An Introduction to Neural Networks*. The University of Amsterdam, 1996.

[52] A. Kundu and J. Chen. Texture classification using QMF bank-based subband decomposition. *Computer Vision, Graphics, and Image Processing. Graphical Models and Image Processing*, 54(5):369–384, 1992.

[53] S. Lawrence, C. Giles, A. Tsoi, and A. Back. Face recognition: A convolutional neural network approach. *IEEE Transactions on Neural Networks*, 8(1):98–113, 1997.

Stellenbosch University http://scholar.sun.ac.za

[54] Y. le Cun. A learning scheme for asymmetric threshold networks. In *Proceedings of Cognitiva 85*, pages 599–604, Paris, France, 1985.

[55] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J.Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, P.Simard, and V. Vapnik. Comparison of learning algorithms for handwritten digit recognition. In F. Fogelman and P. Gallinari, editors, *International Conference on Artificial Neural Networks*, pages 53–60, Paris, 1995. EC2 CIE Publishers.

[56] Z. Liang and C. Thorpe. Stereo- and neural network-based pedestrian detection. In *Proceedings of the IEEE Intelligent Transportation Systems Conference*, Tokyo, Japan, Spring 1999.

[57] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of Optical Society of America A*, 5(5):923–932, May 1990.

[58] J. Mao and A. Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, 25(2):173–188, 1992.

[59] E. Mayoraz and E. Alpaydin. Support vector machines for multi-class classification. In *IWANN (2)*, pages 833–842, 1999.

[60] W. McCullough and W. Pitts. A logical calculus of the ideas imminent in nervous activity, 1943.

[61] B. Mehtre, M. Kankanhalli, D. Narasimhalu, and G. Man. Color matching for image retrieval. *PRL*, 16(3):325–331, March 1995.

[62] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. In *Philosophical Transactions of the Royal Society*, pages 415–446, London, 1909.

[63] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, Mass., 1969.

[64] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[65] J. Montgomery and G. Bekey. Learning helicopter control through 'teaching by showing'. In *IEEE Conference on Decision and Control*, 1998.

[66] H. Murase and S. Nayar. Visual learning and recognition of 3-d objects from appearance. *International. Journal of Computer Vision*, 14(1):5–24, 1995.

[67] C. Nakajima, M. Pontil, and T. Poggio. People recognition and pose estimation in image sequences. *ICJNN*, 2000.

Stellenbosch University http://scholar.sun.ac.za

[68] V. Nalwa and T. Binford. On detecting edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):699–714, November 1986.

[69] M. Opper and D. Haussler. Calculation of the learning curve of bayes optimal classification algorithm for learning a perceptron with noise. In *Computational Learing Theory*, pages 75–87, 1991.

[70] M. Opper and D. Haussler. Generalization performance of bayes optimal prediction algorithm for learning a perception, 1991.

[71] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. Technical Report AIM-1602, 1997.

[72] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *IEEE CVPR'97*, June 1997.

[73] D. Parker. Learning logic. Technical Report TR-47, Center for Computational Research and Managment Science. Massachusetts Institute of Technology, Cambridge, MA, 1985.

[74] A. Pentland, R. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *SPIE Storage and Retrieval for Image and Video Databases*, 2(2185):34–47, 1994.

[75] P. Phillips. Support vector machines applied to face recognition. In M.Kearns, S.Solla, and D.Cohen, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 803–809. MIT Press, 1998.

[76] J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In A.J. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classiers*. MIT Press, 1999.

[77] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. In .A. Solla, T.K. Leen, and K.-R. Muller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 547–553. MIT Press, 2000.

[78] M. Pontil and A. Verri. Support vector machines for 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6):637–646, 1998.

[79] W. Press, B. Flannery, S. Teukolsky, and W. Vettering. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1988.

[80] J. Quinlan. Induction of Decision Trees. *Machine Learning*, 1:81–106, 1986.

Stellenbosch University http://scholar.sun.ac.za

[81] J. Rehg, K. Murphy, and P. Feiguth. Vision-based speaker detection using bayesian networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*. Ft. Collins CO, 1999.

[82] M. Richard and R. Lippman. Neural network classiers estimate bayesian a posteriori probabilities. 3(4):461–483, 1991.

[83] L. Roberts. Machine perception of three-dimensional solids. In J. Tippett et al., editors, *Optical and Electro-Optical Information Processing*, pages 159–197. MIT Press, Cambridge, MA, 1965.

[84] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[85] F. Rosenblatt. *Principles of Neurodynamics*. Spartan Books, Washington D.C., 1962.

[86] C. Rothwell, J. Mundy, W. Hoffman, and V. Nguyen. Driving vision by topology. In *International Symposium on Computer Vision*, pages 395–400, Coral Gables, Florida, 1995.

[87] Y. Rui, T. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in mars. In *Proc. Of IEEE International Conference on ICIP'97*, Santa Barbara, CA, 1997.

[88] D. Rumelhart, G. Hinton, and R. Williams. Learning internal representations by error propagation. In D. Rumelhart and J. McClelland, editor, *Parallel Distributed Processing. Exploration in the Microstructure of Cognition*, volume 1, Cambridge, MA, 1986. MIT Press.

[89] D. Rumelhart, G. Hinton, and R. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland and others, editor, *Parallel Distributed Processing: Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, 1987.

[90] F. Samaria. *Face Recognition using Hidden Markov Models*. PhD thesis, Trinity College, University of Cambridge, Cambridge, 1994.

[91] F. Samaria and A. Harter. Parametrisation of a stochastic model for human face identification. In *Proceedings of the 2nd IEEE workshop on Applications of Computer Vision*, pages 138–142, Sarasota, Florida, 1994.

[92] R. Schettini. Multicolored object recognition and location. *Pattern Recognition Letters*, 15:1089–1097, 1994.

[93] B. Scholkopf, K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Trans. Signal Processing*, 45(11):2758–2765, 1997.

Stellenbosch University http://scholar.sun.ac.za

[94] C. Setchell and N. Campbell. Using colour gabor texture features for scene understanding. In *7th. International Conference on Image Processing and its Applications*, pages 372–376. Institution of Electrical Engineers, 1999.

[95] J. Stone. Computer vision: What is the object? In *Prospects for AI. Artificial Intelligence and Simulation of Behaviour*, pages 199–208, Birmingham, England, 1993. IOS Press, Amsterdam.

[96] W. Nick Street. A neural network model for prognostic prediction. In *Proc. 15th International Conf. on Machine Learning*, pages 540–546, San Francisco, CA, 1998. Morgan Kaufmann.

[97] M. Stricker and M. Orengo. Similarity of color images. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 381–392, 1995.

[98] K. Sung. *Learning and Example Selection for Object and Pattern Detection*. PhD thesis, Massachusetts Institue of Technology, 1995.

[99] K. Sung and T. Poggio. Example Based Learning for View-Based Human Face Detection. Technical Report AIM-1521, Massachusetts Institue of Technology, 1994.

[100] M. Swain and D. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[101] Q. Tian, P. Hong, and T. Huang. Update relevant image weights for content-based image retrieval using support vector machines. In *IEEE International Conference on Multimedia and Expo (II)*, pages 1199–1202, 2000.

[102] G. Towell and J. Shavlik. The extraction of refined rules from knowledge-based neural networks. *Machine Learning*, 13(1):71–101, 1993.

[103] G. Towell and J. Shavlik. Knowledge-based artificial neural networks. *Artificial Intelligence*, 70(1–2):119–165, 1994.

[104] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[105] A. Vailaya. Shape-based image retrieval. Master's thesis, Michigan State University, 1996.

[106] A. Vailaya, H. Zhang, and A. Jain. Automatic image orientation detection. In *In Proc. IEEE ICIP'99 (Intl' Conf. on Image Processing)*, October 1999.

[107] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, Heidelberg, DE, 1995.

[108] V. Vapnik. *Statistical Learning Theory*. John Wiley, 1998.

Stellenbosch University http://scholar.sun.ac.za

[109] G. Wahba. Support vector machines, reproducing kernel hilbert spaces, and randomized gacv. In B. Schoelkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, chapter 6, pages 69–87. MIT Press, 1998.

[110] D. Walsh and C. Omlin. Automatic detection of film orientation with support vector machines. In T. Hendtlass and M. Ali, editors, *Developments in Applied Artificial Intelligence*, page 36 ff. 15th International Conference on Industrial and Engineering. Applications of Artificial Intelligence and Expert Systems, IEA/AIE, LNAI 2358, Springer, 2002.

[111] PERMM: Image Analysis Website. AT&T: `http://www.uk.research.att.com/permm/image_analysis.html`.

[112] W. Wei, E. Barnard, and M. Fanty. Improved probability estimation with neural network models. In *Proc. ICSLP '96*, volume 1, pages 502–505, Philadelphia, PA, 1996.

[113] W. Wei, T. Leen, and Etienne Barnard. A fast histogram-based postprocessor that improves posterior probability estimates. 11(5):1235–1248.

[114] T. Weldon and W. Higgins. Integrated approach to texture segmentation using multiple gabor filters. In *Proceedings of IEEE International Conference on Image Processing (ICIP96)*, volume 3, pages 955–958, Lausanne, Switzerland, September 1996.

[115] P. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.

[116] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, Egham, TW20 0EX, UK, 1998.

[117] B. Widrow and M. Hoff. Adaptive switching circuits. In *Western Electronic Show and Convention, Convention Record*, volume 4, pages 96–104. Institute of Radio Engineers (now IEEE), 1960.