DETECTION OF MALIGNANCY ASSOCIATED CHANGES IN CERVICAL CELL NUCLEI USING FEED-FORWARD NEURAL NETWORKS

By

Roger Kemp

B. Sc. (Physics) Simon Fraser University, 1991

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
Masters of Science

in
THE FACULTY OF GRADUATE STUDIES
DEPARTMENT OF PHYSICS

We accept this thesis as conforming to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
April 1994
© Roger Kemp, 1994
In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature)

Department of PHYSICS

The University of British Columbia
Vancouver, Canada

Date 1994 April 28
Abstract

It has been recognized that normal cells in the presence of a precancerous lesion undergo subtle changes that affect the DNA distribution in their nuclei. These changes have been termed Malignancy Associated Changes (MACs). This thesis examines the design of a classifier that separates normal slides from slides containing MACs in the presence of a severely dysplastic lesion.

Classifiers were designed using discriminant functions and feed-forward neural networks with various structures. The discriminant function correctly separated MACs from normal cells with a classification rate of 61.6% for a 16904 cell test set. Neural network classifiers were able to achieve up to 72.5% separation for this cell-by-cell classification task when four hidden units were used. Using more than four hidden units led to a decline in the test set performance.

The slide-by-slide classification rates were calculated for each classifier based on the distribution of classifier values for the cells on each slide. The discriminant function scored 69.5% on the test set containing 197 slides. The neural network classifiers all scored between 74% and 77% when used for slide-by-slide classification.
Table of Contents

Abstract ii

Table of Contents iii

List of Tables v

List of Figures vii

Acknowledgements xi

1 Introduction 1

2 Classification with Discriminant Functions and Neural Networks 5
   2.1 Linear Discriminant Functions ........................................ 6
   2.2 Neural Networks ..................................................... 11
      2.2.1 Introduction .................................................. 11
      2.2.2 Feed Forward Networks with Error Back-propagation .......... 12
      2.2.3 Alternatives to the Back-Propagation Algorithm ............. 16
      2.2.4 The Quickprop Algorithm .................................... 18
      2.2.5 Decision Surfaces ............................................ 19
      2.2.6 Neural network solutions to the classification example ...... 21
      2.2.7 Overfitting Training Data .................................... 26

3 Classification of Cytological Smears 28
   3.1 Cancer ............................................................... 28
List of Tables

3.1 Descriptive differences between the cells that make up the squamous epithelium. .................................................. 31

4.1 Frequency of the number of cells per slide used in the design of the cell-by-cell classifier. .................................................. 46

4.2 The breakdown of the number of cells of each type used to train and test the cell-by-cell classifiers is shown. ...................... 46

4.3 The frequency distribution for the absolute values of the pairwise correlation between the 53 features is shown. Correlation values near 1 are of interest because they indicate variables which can be eliminated from the classifier model. .................................................. 51

4.4 The average performance of 15 neural networks with 20–2–1 structure trained with Quickprop on 1000 cells (500 MACs, 500 negatives) is shown for different weight decay levels. The test results for an independent set of 1000 cells should be closer to those of the training data for larger decay levels. .................................................. 55

4.5 The linear discriminant function attained a 62.1% classification rate on the training set. .................................................. 58

4.6 The linear discriminant function attained a 61.6% classification rate on the test set. .................................................. 58

4.7 The number of network weights and number of training iterations used are shown for networks with different numbers of hidden units. ............. 60
4.8 The average performance and standard deviations for 20 randomly initialized networks are shown when different numbers of hidden units are used. The training set performance improves as the number of hidden units is increased, but the test set performance peaks when four hiddens are used. The performance of the best network on the test data is given for each case.

4.9 The staining dates of the 41 slides that make up peak 3 (see text) are shown. The total number of slides from each date is given in the rightmost column.

4.10 The first 18 principal components are shown for the MAC training data. They are obtained by finding the eigenvalues of the covariance matrix for 53 feature training set. The contribution of each principal component to the total variation of the data and the cumulative total are also given.

4.11 The average performance and standard deviations for 20 randomly initialized networks are shown for different numbers of hidden units. The networks were trained on the first ten principal components of the training set, which accounts for 86.9% of the variation of the data.

4.12 Breakdown of the number of slides of each type used to design slide-by-slide classifiers.

4.13 Correlation values of the five slide features are shown for the five hidden unit cell-by-cell classifier. These values were calculated for the slide training set.

4.14 The performances of the slide features for classification of the test set of 197 slides are shown for the best neural networks and discriminant function.
List of Figures

2.1 Decision boundary and contours of equal probability density are shown for two populations that identical covariances of the form $S_i = \sigma^2 I$. 

2.2 A linear discriminant vector is shown along with the separating boundary (solid line) for an example of a two class classification problem. Discriminant function value contours are lines perpendicular the discriminant vector as shown by the two additional contours in the figure.

2.3 The decision boundary for a quadratic discriminant function is shown for the classification problem. This discriminant function correctly separates 85.5% of the training examples.

2.4 Structure of a feed-forward with one layer of hidden units.

2.5 A sigmoid shaped function (one that saturates for extreme input values) is the most common nonlinear activation function used with neural networks.

2.6 Decision surfaces are shown for feed-forward networks with one, two and three layers of weights. In the first case the separating surface will be a hyperplane and the network performs like discriminant function vector. When there are two layers of weights, the decision surface will be curved and can separate a single region of arbitrary shape. If more than two weight layers are used, the network can form boundaries around multimodal distributions in the feature space.

2.7 The combination of two vectors defines a decision region in the lower left corner that approximates the original distribution of the training examples.
2.8 The structure of a neural network with two hidden neurons that implements the decision boundary defined by the two vectors is shown. 23

2.9 The decision boundary is shown for a neural network before and after it was optimized using a conjugate gradient training technique. 24

2.10 Three neural network output contours are shown for a 5 hidden unit solution to the classification example. The decision boundary is shown by the .50 output value contour. The .01 and .99 contours nearly coincide with the decision boundary. Thus all points in the plane have neural network output values of nearly zero or one. 25

3.1 The types of cells that make up the squamous epithelium are shown schematically. There are usually several layers of parabasal, intermediate and superficial cells in normal epithelia. 29

3.2 The differences between the different stages of precancerous lesions are shown. The images on the left are the nuclei of typical cells of the four types scanned using the automated image cytometer described in Section 3.5. 35

3.3 The decision tree used by the automated classifier system at the B.C. Cancer Agency Imaging Department. Each node represents a decision currently made using a discriminant function or threshold. 40

4.1 Every third cell nucleus was selected for inclusion in the training data and the remainder of the data was used for testing. 47

4.2 The cell-by-cell performance of neural networks with different numbers of hidden units is shown for the training and test sets. Each result is averaged for 20 runs using randomly initialized networks. The standard deviations are shown as error bars for each result. 61
4.3 The ROC curves for the discriminant function and networks giving the best test set performance are shown. The ROC curves for the best networks using three, five, six and seven hidden units nearly coincide with those of the two and four hidden networks and are not shown.

4.4 The histogram of discriminant function values is shown for the test data. The discriminant function correctly classifies 61.6% of the test examples.

4.5 The histogram of neural network output values for the optimal network with 1 hidden unit is shown for the test data. This network correctly classifies 63.5% of the test examples.

4.6 The histogram of perceptron output values is shown for the test data. The perceptron assigns negative cells to extreme output values.

4.7 The ROC curves for the perceptron and the combination of the perceptron and threshold are shown. The ROC curves for the neural network with two hidden units is shown for comparison purposes.

4.8 The histogram of neural network output values for the optimal network with 2 hidden units is shown for the test data. This network correctly classifies 71.9% of the test examples.

4.9 The histogram of neural network output values for the optimal network with three hidden units is shown for the test data. This network correctly classifies 72.4% of the test examples.

4.10 The histogram of neural network output values for the optimal network with four hidden units is shown for the test data. This network correctly classifies 72.5% of the test examples.

4.11 The histograms of neural network output values for the optimal network with 4 hidden units is shown for the whole negative test set and for the case where 41 slides have been removed.
4.12 The outputs for every eight training example for the two perceptrons that make up the two hidden unit networks are plotted.

4.13 The two principal components are shown for an artificial data distribution in two dimensions shown by the shaded region. The first principal component lies along the direction of greatest variation of the data. The second component is constrained to be perpendicular to the first.

4.14 The slide-by-slide ROC curves for the discriminant function and neural networks with two and five hidden units are shown. The slide test set consists of 125 normal and 72 severe slides.
Acknowledgements

I would like to acknowledge several people who have helped me a great deal during my research and writing of this thesis. First, I would like to thank Dr. Branko Palcic for introducing me to the field of cytology and making this thesis possible. Dr. Calum MacAulay has provided me with good advice and suggestions on many topics relating to pattern classifiers. I would also like thank Henry Lee for his friendship and wisdom, and Peter Payne who has been my guru for all things cytopathological.
Chapter 1

Introduction

Cancer is one of the leading causes of death in society today. In 1987, the death rates for cancer in Canada were 212 per 100,000 for men and 168 per 100,000 for women [44]. This places cancer, which accounts for 26% of all fatalities, number two behind heart disease, which accounts for over 40% of all fatalities.

The incidence of cancer in British Columbia is on the rise. The incidence rate for new cancers in 1991 was approximately 300 per 100,000 [3], which is up 9% over incidence data from 10 years ago. This statistic was standardized to account for the aging of the population. For these reasons, cancer remains a research priority. Research is conducted into treatment, prevention and early detection of cancers. Early detection is important because many cancers are more successfully treated when detected early.

For some types of cancer, early detection can be accomplished by public screening programs. British Columbia has had a cervical cancer screening program in place since 1949. Over the period from 1955 to 1985 the mortality rate from cervical cancer has dropped from approximately 11 per 100,000 to 3 per 100,000 [1]. This improvement is largely due to the early diagnosis of this cancer, where it can be effectively treated.

In the case of cervical cancer, a sample smear is done by scraping the surface of the cervix to obtain a cell sample and fixing the cells on a slide. The slide is stained and the diagnosis is made by a pathologist or cytotechnologist. The diagnosis is made by searching the slide for examples of abnormal cells, which are characteristic of a malignancy. This is done by viewing different portions of the slide under a microscope and distinguishing
Chapter 1. Introduction

the cells from other materials and debris on the slide. On average, there are over 150,000 cells on each slide and rendering the diagnosis requires careful examination of most of these objects.

Under the current screening program in British Columbia, women are examined every two years between the onset of sexual activity and age 35, and every 5 years thereafter. The cytotechnologists at the B.C. Cancer Agency, therefore, must examine between 2500 and 3000 slides daily. This labor intensive task can be accomplished in British Columbia because of the amount of resources devoted to the screening program. However, most countries, including the United States, do not have such public screening programs in place.

A computerized system to screen cervical cell smears would be a valuable tool to aid cytotechnologists. Recently, researchers at several laboratories [32, 38, 39, 51] have built prototype systems to screen smears automatically. These systems attempt to detect abnormal cells in order to aid the cytotechnologist’s diagnosis of the slide.

A new approach to the diagnosis of cervical smears involves detecting Malignancy Associated Changes (MACs)[34, 36] in apparently normal cells. It has been shown that normal cells in the presence of a precancerous lesion undergo subtle changes that affect their nuclear features [4, 7, 30, 31]. These changes cannot be usually detected by eye, but can be seen when making quantitative measurements of the DNA distribution in the nucleus. A decision process which includes the detection of MACs could be used to detect cancer in cases where no tumor cells are present on the cervical smears.

A new automated image cytometer [15] developed by Xillix Technology Corporation and the B.C. Cancer Agency uses both abnormal cells and the presence of MACs to diagnose cervical smears. The device scans the nuclei of stained cervical smears and calculates up to 120 quantitative nuclear features. These features are used to classify each object on the slide and ultimately the entire slide itself. The usual decision methods
used in this process are linear discriminant functions in combination with thresholds on the nuclear features. This thesis compares the use of linear discriminant functions and neural networks for the detection of MACs.

Neural networks have become a popular pattern classification technique since the publication of the back-propagation algorithm [42] nearly ten years ago. They are useful for decision problems where a nonlinear decision surface separates different classes of data. For this reason they can be successful in problems where standard statistical techniques such as the discriminant function fail.

Chapter two of this thesis derives the linear discriminant function and the back-propagation algorithm for training neural networks. The relationship between the two decision methods is shown using a two-class classification problem. This problem is solved using discriminant functions and neural networks with different structures. It is also used to demonstrate how overfitting of the training data can occur when classifiers with too many free parameters are used. Alternative neural network training methods are introduced and discussed.

Chapter three gives background regarding cancer and its effects on cells. The structure of the squamous epithelium, which lines the uterine cervix, is described. A brief description of the nature of MACs and what is known about them is given in Section 3.4. Cytological screening and issues regarding the design of an automated cytological classifier are discussed in this chapter.

Chapter four contains the results of a series of experiments performed on a dataset consisting of MAC cells and normal cells. A detailed description of the data used in the studies is given in Section 4.1. The results are given for experiments to:

1. calculate the correlations between nuclear features,

2. compare three neural network training methods,
Chapter 1. Introduction

3. study the effects of varying neural network weight decay parameters,

4. design classifiers that separate cells from different staining batches,

5. design cell-by-cell discriminant function and neural network classifiers,

6. use principal component analysis to reduce the dimensionality of the data, and

7. perform slide-by-slide classification using a discriminant function and optimal neural networks of various structures.

Chapter five summarizes the results of this thesis and suggests further worked based on these results.
Chapter 2

Classification with Discriminant Functions and Neural Networks

A pattern classifier is a rule or procedure for assigning a label to an object. An object can be represented by a vector of features

\[ x = (x_1, x_2, ..., x_d) \]  

where each \( x_i \) is some measurement of a feature of \( x \) and \( d \) is the number of features used to describe the object. A classifier transforms objects in the \( d \) dimensional feature space into scalar values which may then be interpreted as probabilities or measures of the degree to which objects belong to a particular class. The features can be physical measurements or enumerations of some quality that distinguishes the classes, such as colour, etc.

The design of a classifier begins with choosing a set of prototypes from the desired classes and selecting a classification model such as a discriminant function, neural network or Bayes network [45]. The classifier is trained with the prototypes and parameters are found for the selected model that minimize the error function appropriate to the model. If the model is sufficiently complex, i.e. if a large number of parameters are used, it is possible to form a rule that correctly classifies most or all of the training examples. However, the performance of the classifier will generally be worse on any mutually exclusive validation (test) set. Consequently, the use of a separate test set is important to confirm how well the classifier can be expected to classify new data. Its performance on the training data can be misleading, a point which will be demonstrated
further in this chapter. The performance of the classifier on a test set is commonly referred to as its ability to generalize [10].

2.1 Linear Discriminant Functions

The linear discriminant function [11] is one of the most common methods used for pattern classification. It was introduced by Fisher [13] as a technique for reducing multidimensional data to scalar “discriminants” such that the means of populations of different classes are maximally separated. The linear discriminant vector described by Fisher can also be derived from Bayesian principles [20]. If the populations of the two classes are normally distributed and have identical covariance matrices, the maximum-likelihood Bayesian classifier turns out to be the linear discriminant vector.

Let \( x_1, \ldots, x_n \) be \( n \) samples with \( n_0 \) of class 0 from set \( X_0 \) and \( n_1 \) of class 1 from set \( X_1 \). Let \( w \) be the discriminant vector whose components we wish to find. Then, let \( y_i = w^T x_i \) and define \( Y_0 \) and \( Y_1 \) to contain each \( y_i \) for which the corresponding \( x_i \) is contained in \( X_0 \) or \( X_1 \). The sample means are given by

\[
\mathbf{m}_i = \frac{1}{n_i} \sum_{x \in X_i} x
\]

and

\[
\bar{m}_i = \frac{1}{n_i} \sum_{y \in Y_i} y = \frac{1}{n_i} \sum_{x \in X_i} w^T x = w^T \mathbf{m}_i
\]

Let the covariance matrices for the populations be

\[
\mathbf{S}_i = \sum_{x \in X_i} (x - \mathbf{m}_i)(x - \mathbf{m}_i)^T
\]

and define scatter of the \( Y_i \) populations to be

\[
\bar{s}_i^2 = \sum_{y \in Y_i} (y - \bar{m}_i)^2
\]
Then $s_i^2$ is proportional to the variance within the class and can be written as

$$s_i^2 = \sum_{x \in X_i} (w^T x - w^T m_i)^2 = \sum_{x \in X_i} w^T (x - m_i)(x - m_i)^T w = w^T S_i w$$  \hspace{1cm} (2.6)$$

Then the Fisher linear discriminant is the vector $w$ which maximizes the expression

$$J(w) = \frac{|\tilde{m}_0 - \tilde{m}_1|^2}{\tilde{s}_0^2 + \tilde{s}_1^2} = \frac{|w^T (m_0 - m_1)|}{w^T (S_0 + S_1) w}$$  \hspace{1cm} (2.7)$$

The numerator of this expression is the separation between the means of the two data classes while the denominator is proportional to the variances within the classes. Thus, maximizing $J(w)$ seeks to maximize the separation between the classes while minimizing the spread within the classes. If the populations have identical covariance matrices, i.e. $S_0 = S_1$ then there is a solution for $w$ which maximizes $J(w)$, namely

$$w = S_0^{-1} \cdot (m_0 - m_1)$$  \hspace{1cm} (2.8)$$

For each new object $x$ we form the product $y = w^T x$ and assign $x$ to class 1 if $y$ exceeds some threshold, otherwise to class 0. The discriminant vector $w$ measures the association of objects to the two classes. When the expression for $y$ is written using the components of $w$ as $y = \sum_j w_j x_j = d(x)$, $d(\ldots)$ is referred to as a discriminant function. The terms discriminant function and discriminant vector will both be used where appropriate.

The second term in the product in Equation 2.8 is the vector that connects the means of the two populations. If the covariance $S_i = \sigma^2 I$ where $I$ is the identity matrix, and each feature has identical variance $\sigma^2$, then the objects fall into identical hyperspherical clusters centered around the respective class means.

This situation is shown in Figure 2.1 for a two dimensional classification task. In this case the discriminant vector is, to within a multiplicative constant, $(m_0 - m_1)$ and the decision surface will be a hyperplane perpendicular to the vector at its midpoint.
When $S$ has some other form, the vector $w$ is adjusted from the direction $(m_0 - m_1)$ by the matrix $S_0^{-1}$ (Equation 2.8). This means that if a feature has a large variance with respect to the separation between the class means, it makes a smaller contribution to the discriminant vector.

Figure 2.1: Decision boundary and contours of equal probability density are shown for two populations that identical covariances of the form $S_i = \sigma^2 I$.

Figure 2.2 shows a two dimensional classification problem which will be used to demonstrate the behavior of discriminant functions and neural networks. The two groups of objects were generated by taking uniformly distributed points from slightly overlapping annular regions in the first quadrant and stretching them along the vector $(1,1)$. Each group consists of 100 points, and these points were used to train the neural networks and discriminant functions. The entire training set of 200 points is shown in Figure 2.2. A larger validation set of 1000 points was constructed to test the performance of the classifiers. It consisted of 500 objects from each class drawn from the same probability
density distributions.

Figure 2.2: A linear discriminant vector is shown along with the separating boundary (solid line) for an example of a two class classification problem. Discriminant function value contours are lines perpendicular the discriminant vector as shown by the two additional contours in the figure.

Figure 2.2 also shows the linear discriminant vector which best separates the two groups. Lines perpendicular to the vector represent contours of equal discriminant value. The decision boundary is shown by the solid line between the two groups. Adjusting the position of decision threshold is equivalent to adding a constant to each of the \( y_i \) values. This shifts the location of the decision boundary along the direction of the discriminant vector. The decision boundary shown in the figure correctly separates 81.5% of the 200 training examples and 83.7% of the 1000 test examples.
In practice, the condition that the covariances be equal is rarely observed. If the populations are both multivariate normally distributed but the covariances are different, then the optimal discriminant function will be quadratic. The optimal discriminant function \( g(x) \) will have the form

\[
g(x) = w_0 + \sum_{j=1}^{d} w_j x_j + \sum_{j=1}^{d} \sum_{k \neq j}^{d} w_{jk} x_j x_k = \sum_{j=1}^{d^*} a_j z_j(x)
\]  

(2.9)

where \( d^* \) is the dimension of a feature space containing all the original features plus all the terms of products of pairs of features. The value of \( d^* \) is given by \( d^* = \frac{1}{2}(d+1)(d+2) \) and \( a \) and \( z \) are \( d^* \)-dimensional vectors.

Points in the \( d \)-dimensional space are mapped into points in the \( d^* \)-dimensional space. In \( z \)-space the decision surfaces will be hyperplanes, and the linear discriminant vector can be found as described above. In \( x \)-space this discriminant function will contain quadratic terms of the \( x_i \) variables so the decision surfaces will be hyperparaboloids. This is commonly referred to as a quadratic discriminant function.

This highlights a problem with the terminology for discriminant functions. The discriminant functions are linear because they result from the inversion of the covariance matrix \( S_i \); not because of the form of the decision surface in the original feature space. Duda [11] uses the term generalized linear discriminant to refer to the discriminants which can be expressed as \( g(x) = a^T y \).

The curse of dimensionality can make it difficult to use generalized discriminants. For a problem with ten free features, a quadratic discriminant will have 66 components while a third order (cubic) discriminant would have 286 components. Thus it becomes impractical to use discriminants of higher orders unless there are very few features or very many training examples.

Figure 2.3 shows the decision surface for a generalized linear discriminant with quadratic terms of the two features. It does a better job of approximating the curvature of
the decision boundary between the two classes. The boundary correctly separates 85.5% of the training examples and 86.9% of the test examples.

Figure 2.3: The decision boundary for a quadratic discriminant function is shown for the classification problem. This discriminant function correctly separates 85.5% of the training examples.

2.2 Neural Networks

2.2.1 Introduction

There has been a great deal of interest in using neural networks for pattern classification during the last ten years. Pattern classification is done by the brain all the time. Images are recognized by the brain in a fraction of a second (say, \( \sim 0.1 - 0.2 \) seconds). Yet, the
firing rate of neurons in the brain is of the order .005 seconds. This means that the brain can recognize an image in roughly 30 neuron cycles.

Today's computers operate approximately a million times faster than the brain. Yet, the fastest computers running sophisticated pattern recognition algorithms cannot come close to what the brain does routinely. It is the distributed nature of the brain that gives it its ability to quickly process images and categorize them. This distributed processing feature is the motivation behind developing neural network systems for classification.

It is known that there are \( \sim 10^{11} \) neurons in the brain each with \( \sim 10^4 \) interconnections. Each neuron performs a non-linear transformation on a weighted combination of its inputs and passes the result to others via synaptic junctions. The term neural networks is used to describe a variety of systems that have this property, i.e. interconnected elements each performing simple computations. The term multilayer perceptron, after the terminology introduced by Rosenblatt [41], is used by many authors to maintain the distinction between the implementation of neuron-like algorithms on computers and the true neural networks of the brain.

2.2.2 Feed Forward Networks with Error Back-propagation

The most popular neural network paradigm is the feed forward network trained by the back propagation of errors [42]. The back propagation algorithm describes how the weights of a neural network are changed in order to train it to give a particular output when presented with a set of input/output pairs \((x^\mu, y^\mu)\).

Figure 2.4 shows a schematic of a feed forward neural network with two layers of connection weights. The \( d \) inputs are shown by \( x_i \) and the \( m \) outputs are shown by \( O_k \). Between the inputs and outputs are a layer of \( n \) hidden units denoted \( V_j \). The inputs are connected to the hidden units by connection weights \( w_{ij} \), and the hidden units are connected to the outputs by connection weights \( W_{jk} \). Subscript \( i \) refers to only input
Let the training examples \((x^\mu, y^\mu)\) come from a set of \(p\) training pairs. For each training pattern, hidden unit \(j\) receives the sum

\[
h_j^\mu = \sum_i w_{ij} x_i^\mu
\]

and produces the output

\[
V_j^\mu = g(h_j^\mu) = g(\sum_i w_{ij} x_i^\mu)
\]

where \(g(\ldots)\) is a nonlinear function. Output unit \(k\) receives the sum

\[
h_k^\mu = \sum_j W_{jk} V_j^\mu = \sum_j W_{jk} g(\sum_i w_{ij} x_i^\mu)
\]

and generates the output

\[
O_k^\mu = g(h_k^\mu) = g(\sum_j W_{jk} g(\sum_i w_{ij} x_i^\mu))
\]

This process is repeated for as many layers of weights as exist in the network.
The standard error function is the sum-of-squares error measure

\[ E(w) = \frac{1}{2} \sum_{\mu k} (y_k^\mu - O_k^\mu)^2 \]  

(2.14)

\( E \) is differentiable with respect to the weights provided \( g(\ldots) \) is also differentiable. The derivative of \( E \) with respect to the hidden-to-output units is

\[ \frac{\partial E}{\partial W_{jk}} = - \sum_{\mu} (y_k^\mu - O_k^\mu) g'(h_k^\mu) V_j^\mu \]

\[ = - \sum_{\mu} \delta_k^\mu V_j^\mu \]  

(2.15)

where

\[ \delta_k^\mu = g'(h_k^\mu)(y_k^\mu - O_k^\mu) \]  

(2.16)

The derivative \( E \) with respect to the input-to-hidden units is calculated using the chain rule

\[ \frac{\partial E}{\partial w_{ij}} = \sum_{\mu} \frac{\partial E}{\partial V_j^\mu} \frac{\partial V_j^\mu}{\partial w_{ij}} \]

\[ = - \sum_{\mu k} (y_k^\mu - O_k^\mu) g'(h_k^\mu) W_{jk} g'(h_j^\mu) x_i^\mu \]

\[ = - \sum_{\mu} \delta_j^\mu x_i^\mu \]  

(2.17)

where

\[ \delta_j^\mu = \sum_k (y_k^\mu - O_k^\mu) g'(h_k^\mu) W_{jk} g'(h_j^\mu) = g'(h_j^\mu) \sum_k W_{jk} \delta_k^\mu \]  

(2.18)

For multi-layer networks the derivatives are found by further applying the chain rule to each layer of weights.

With the back-propagation algorithm, network weights are updated using

\[ \Delta W_{jk} = -\eta \frac{\partial E}{\partial W_{jk}} \quad \text{and} \quad \Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} \]  

(2.19)

in order to minimize the error function \( E \). The parameter \( \eta \) is referred to as the learning rate and controls the size of the steps taken in the direction of the error gradient. The
task of training a neural network becomes a multivariate optimization problem with the network weights as the optimization parameters. Training patterns \( x^\mu \) are applied to the network inputs and the results are fed forward through the network until an output pattern \( O^\mu \) is obtained. The output is compared to the desired output patterns \( y^\mu \), and the error function \( E \) is computed. From this, the error contributions of the individual units (the \( \delta \)'s) can be calculated. These errors are propagated back through all the network weights towards the inputs. The prescription for updating each weight (Equation 2.19) can then be applied.

The function \( g(...) \) is referred to as the transfer or activation function. As the equations above show, it must be differentiable. It must also be nonlinear, otherwise the neural network as whole will be linear and all non-input units can be replaced by a single unit. The standard activation function is sigmoidal in shape (Figure 2.5). It saturates for small and large inputs. This means that for inputs with large magnitude (positive or negative), the function returns a value that asymptotically approaches the minimum or maximum output values. This nonlinear behavior is the key feature which gives neural networks the ability to perform decisions that cannot be made with discriminant functions.

The two most common activation functions are both exponential in form:

\[
g(h) = \frac{1}{1 + e^{-2\beta h}}
\]

is used when using units that have a \([0,1]\) output range and

\[
g(h) = \tanh(\beta h)
\]

when using units that have a \([-1,1]\) output range. Both functions have a central range where their behavior is almost linear. The parameter \( \beta \) controls the steepness at which the activation function approaches its extreme values.
Sigmoid activation function used by many feed-forward networks

\[ g(h) = \frac{1}{1 + e^{-h}} \]

Figure 2.5: A sigmoid shaped function (one that saturates for extreme input values) is the most common nonlinear activation function used with neural networks.

It is usual for each non-input unit in the network to have a bias or threshold. Standard practice is to add an extra connection weight between each unit and an artificial input unit clamped at ±1. These connection weights are treated like regular weights when calculating derivatives and performing weight updates. Thus, the number of weights in a one hidden layer feed-forward network with full connections between layers will be

\[ n_{weights} = (n_{inputs} + 1) \times n_{hiddens} + (n_{hiddens} + 1) \times n_{outputs} \quad (2.22) \]

2.2.3 Alternatives to the Back-Propagation Algorithm

The back-propagation algorithm is a non-linear optimization algorithm that takes steps of a given length \( \eta \) along the error gradient. If \( \eta \) is very small, the convergence of the network to an acceptable solution will be very slow. If \( \eta \) is too large, the training error will oscillate wildly as many of the update steps will overshoot the minima of local valleys of the error value in the weight-space.
Hundreds of papers have been published in which the authors describe some improvement of the back-propagation algorithm. The most common improvement is the introduction of a momentum parameter, \( \alpha \). The weight update equation (Equation 2.19) is modified so that each step includes a contribution of the previous weight update:

\[
\Delta w_{ij}(t + 1) = -\eta \frac{\partial E}{\partial w_{ij}} + \alpha \Delta w_{ij}(t)
\]  

(2.23)

The momentum, \( \alpha \), has a value between zero and one. The best value of \( \alpha \) and \( \eta \) are usually found by trial and error. Several authors describe adaptive techniques to automatically adjust these parameters [48, 19] to improve the rate of convergence.

The realm of operations research provides many alternatives to gradient descent which are much better than the back-prop algorithm. Some of these methods rely on calculating the second derivative of \( E \) with respect to the network weights. The Taylor expansion of \( E \) about the current set of weights \( w_0 \) is

\[
E(w) = E_0 + (w - w_0) \cdot \nabla E(w_0) + \frac{1}{2}(w - w_0)^T \cdot H \cdot (w - w_0) + \cdots
\]  

(2.24)

where \( H \) is the Hessian matrix of second derivatives

\[
H_{ij} = \frac{\partial^2 E}{\partial w_i \partial w_j}
\]  

(2.25)

Ignoring higher order terms, the derivative of Equation 2.24 with respect to \( w \)

\[
\nabla E(w) = \nabla E(w_0) + H \cdot (w - w_0)
\]  

(2.26)

To minimize \( E \) we set \( \nabla E(w) = 0 \) and rearrange (Equation 2.26) in terms of \( w \) to obtain

\[
w = w_0 - H^{-1} \cdot \nabla E(w_0)
\]  

(2.27)

This technique is Newton's method for function minimization [40]. It requires that we compute and invert a \( d \times d \) matrix of second derivatives at each iteration. This can
be costly in terms of both memory and CPU time when implemented in an algorithm. Further, unless the error function is quadratic with respect to the weights it can perform in an unstable manner. As such, it is not practical to implement in this form.

It is possible to obtain an optimization algorithm with second order convergence without explicitly calculating $H$ [40]. This fact is used by conjugate gradient methods such as the Polak-Ribiere method, which are the basis of several neural network training techniques [23, 28]. At each iteration of the algorithm, a line search is made along a direction $d_{\text{new}}$ that is selected to contain contributions from both the gradient and the previous search direction:

$$d_{\text{new}} = -\nabla E_{\text{new}} + \gamma d_{\text{old}}$$

(2.28)

The factor $\gamma$ is calculated at each step to optimize the contribution of the previous search direction. These algorithms can provide an order of magnitude increase in the rate of convergence over the standard back-propagation algorithm for some problems.

2.2.4 The Quickprop Algorithm

Several neural network training procedures have been developed which use approximations of the second derivative matrix. These methods perform repeated line searches based on an approximation to $H^{-1}$ and are known as quasi-Newton or variable metric methods. Watrous [49] has studied their effectiveness for training neural networks and found them to be an order of magnitude more efficient than standard back-propagation.

An alternative to using the full second derivative matrix is to ignore the off-diagonal elements of $H$. Fahlmann's Quickprop algorithm [12] treats the change in the slope of the error function $E(w)$ as a function of each weight independently. In this case it is possible to use the pseudo-Newton rule

$$\Delta w_{ij} = -\frac{\partial E}{\partial w_{ij}} \frac{\partial^2 E}{\partial^2 w_{ij}}$$

(2.29)
to update each weight. Further he assumes that the error function is a parabolic function opening upwards. In this case Equation 2.29 reduces to

$$\Delta w(t) = \frac{\partial E}{\partial w(t)} - \frac{\partial E}{\partial w(t-1)} \delta w(t-1)$$

where $t$ refers to the current iteration and $t - 1$ to the previous. Since the update rule reduces to using the current and previous derivatives of the error it is efficient to implement it in a training algorithm. This compensates for the simplification that is made by assuming that the error surface is quadratic.

There are situations when following this update rule can lead to difficulties. If the current and previous error slopes are nearly equal then the denominator of Equation 2.30 will be near zero and the rule will prescribe an enormous update to the weights. To prevent this, Fahlman introduces a parameter $\mu$ which he calls the “maximum growth factor”. No step is allowed to exceed a previous step by a factor greater than $\mu$. After experimenting with a variety of values he recommends using $\mu = 1.75$ for most problems.

To prevent the algorithm from oscillating around the local error minimum a gradient descent term is added to the weight update. This term adds $\epsilon$ times the current slope to each $w_{ij}$ if the slope of $\frac{\partial E}{\partial w_{ij}(t)}$ has the same sign as $\frac{\partial E}{\partial w_{ij}(t-1)}$. If the two slopes have different signs then the minimum has been overshot and only the quadratic update rule is used.

### 2.2.5 Decision Surfaces

As was shown in Figure 2.2, a discriminant vector forms a hyperplane separating boundary between two populations. This is also the case for a neural network that contains a single layer of weights (Figure 2.6). The vector of network weights, $w$, can be interpreted as a discriminant vector because the non-linear activation function, $g(w)$, is invertible. This type of neural network structure is called a perceptron [41].

When two layers of weights (one layer of hidden units) are used, each hidden unit in
Figure 2.6: Decision surfaces are shown for feed-forward networks with one, two and three layers of weights. In the first case the separating surface will be a hyperplane and the network performs like discriminant function vector. When there are two layers of weights, the decision surface will be curved and can separate a single region of arbitrary shape. If more than two weight layers are used, the network can form boundaries around multimodal distributions in the feature space.

the network acts as a discriminant vector in the feature space. Since these discriminant values pass through the activation function (Equation 2.11) before reaching the output units, the resulting decision boundary will be a single curved surface. The corners shown in the decision surfaces in Figure 2.6 will be rounded because hidden unit outputs are summed at the output unit and contributions from two hidden units will adjust the position of the switching threshold [24]. If the weights between the inputs and each hidden unit are scaled by some large value it is possible to approach arbitrarily close to the boundaries shown in the figures. It has been shown that a neural network with a
single layer of hidden units can represent any continuous function to an arbitrary accuracy given enough hidden units [9].

Note that if only a single hidden unit is used in the hidden layer, the decision surface will be a hyperplane, as with the perceptron. The neural network requires at least two hidden units to form a nonlinear decision surface.

When three or more layers of weights (two or more layers of hidden units) are used it is possible to approximate multimodal population distributions. Given enough hidden units it is possible to approximate any function with arbitrary accuracy with exactly two hidden layers [8].

2.2.6 Neural network solutions to the classification example

The relationship between the decision surface of a feed-forward neural network and a discriminant function vector can be demonstrated with the classification example from the previous section. Figure 2.2 showed the optimal boundary when a single separating plane is used. The curved boundary could be approximated better if two separating lines were used in combination as shown in Figure 2.7. Neither boundary alone provides good separation for all the data, but the combination of two, shown by the region in the lower left, provides a better fit to the training examples. The vectors to which these boundaries correspond are also shown in the figure. The first points along the direction (0.4, 1.0) while the second points along the direction (1.0, 0.4).

It is possible to construct a simple neural network which contains these two vectors among its weights. Figure 2.8 shows a network with two hidden units connected to the inputs. The weights leading to the first hidden unit \( w_1 \) are set to a constant \( k_1 \) times the first discriminant vector \( (1.0, 0.4) \), and similarly \( w_2 = k_2(0.4, 1.0) \). Then, making a few decisions regarding how fast the output should change from zero to one, the values for \( k_1, k_2 \) and the remaining weights can be found.
Chapter 2. Classification with Discriminant Functions and Neural Networks

The combination of two vectors defines a decision region in the lower left corner that approximates the original distribution of the training examples.

The decision surface for this neural network is shown by the dotted line in Figure 2.9. Although only a rough approximation to the boundary in Figure 2.7, it shows how the weights connecting the inputs to each hidden unit can be interpreted as discriminant vectors in the feature space. This network was then optimized using the Polak-Ribiere conjugate gradient optimization routine. The training set was presented to the network for 200 iterations of the algorithm, resulting in the optimized design shown in the figure. The optimized network correctly classifies 89% of the training examples and 88.1% of the test set.

There are two ways to improve the performance of a neural network solution to this...
Figure 2.8: The structure of a neural network with two hidden neurons that implements the decision boundary defined by the two vectors is shown.

The performance of the previous solution, obtained after 200 iterations of the conjugate gradient method, could be improved slightly by optimizing the network for a longer time. In this case the improvement is marginal, and there is no detectable difference between the decision boundaries for the two networks.

The second way is to use a larger number of hidden units. Adding extra hidden units to the network adds parameters to the system, thereby increasing its ability to fit the training examples. Figure 2.10 shows three contours of neural network output for a neural network with 5 hidden units.

The activation function used for the neural network classifiers is Equation 2.20, with $\beta = 1.0$. Since the range of the neural network output (Equation 2.13) is the range of $g(\ldots)$, each point in the plane of Figure 2.10 is mapped to an output value between zero and one. The contours of output value equal to 0.01, 0.50, and 0.99 are shown. The network was obtained by performing 10,000 iterations of the conjugate gradient training method using the 200 training examples.

Because of the large number of presentations of the training patterns the training procedure produced a network that is precisely tuned to the training examples. The
Figure 2.9: The decision boundary is shown for a neural network before and after it was optimized using a conjugate gradient training technique.

decision boundary (solid line) appears to be a series of five segments, each due to the influence of one of the hidden units. The contours of neural network output equal to 0.01 and 0.99 are both shown as dotted lines in the figure. They almost coincide the decision boundary in many areas. This means that there are almost no regions where the training examples have intermediate values (between .01 and .99). All points in the plane are mapped to either nearly zero or one. This demonstrates how the neural network output, although continuous, can come arbitrarily close to a discontinuous mapping.

There is almost no rounding of the corners where the decision surfaces for these vectors meet. This occurs because of the large network weights that can result from training the
network for many iterations. Given that the original distribution being approximated had a smooth, nearly circular boundary this result is not desirable.

The network correctly identifies 92.5% of the training examples but only 86% of the test examples. Thus, the performance of the five hidden unit neural network on the test set is poorer than that of the two hidden unit neural network. This is an example of what is referred to as overfitting. The large number of parameters in the model allows the network to give undue attention to individual training patterns. Its training set performance is better than that of the simpler network but no longer reflects how well it

Figure 2.10: Three neural network output contours are shown for a 5 hidden unit solution to the classification example. The decision boundary is shown by the .50 output value contour. The .01 and .99 contours nearly coincide with the decision boundary. Thus all points in the plane have neural network output values of nearly zero or one.
will perform on a separate test set.

2.2.7 Overfitting Training Data

The performance of the discriminant vector (Figure 2.2) and the five hidden unit network (Figure 2.10) show the difficulty of selecting the right number of parameters to fit the training data. If too few parameters are used the decision rule will not be able to adequately separate the classes, while if too many parameters are used the generalization will be poor.

Most of the time, it is difficult to know how many parameters to use in a classifier in advance. It is possible that a problem may be linearly separable, which means that a plane can be used to separate the two distributions. In this case a neural network with hidden units is not even needed. The optimal neural network will have only one layer of weights (i.e. a perceptron) and is functionally equivalent to a discriminant vector.

It is likely that there are errors in the training and test sets used in most real world applications. There may be misclassifications of examples or measurement errors in their features. Consequently, it is not desirable to fit the data beyond a certain level of accuracy because we may be attempting to fit the noise in the system.

To avoid overfitting, several methods have been introduced to reduce the number of parameters in the system. The most widely used approach is weight decay. A penalty term is added to each weight during each iteration to reduce its size [23]. This can lead to some weights having nearly zero magnitude, whereupon they can be removed from the network either during the training [18] or afterwards. The penalty term causes weights that are not constantly being reinforced to decrease in value.

One way to introduce the penalty term is to multiply each weight by a factor \((1 - \epsilon)\)
after each iteration, so that
\[ w_{ij}^{\text{new}} = (1 - \epsilon)w_{ij}^{\text{old}} \]  \hspace{1cm} (2.31)

where \( \epsilon \) is the weight decay term. This method is the same as modifying the error function to be
\[ E^{\text{new}} = E^{\text{old}} + \frac{\epsilon}{2\eta} \sum_{ij} w_{ij}^2 \]  \hspace{1cm} (2.32)

The value of \( \epsilon \) used will depend on the amount of noise in the training set, although using a value of \( 10^{-4} \) or \( 10^{-3} \) appears to be common [23].

Reducing all weights by a constant fraction may excessively penalize the use of large weights causing the network to use many smaller weights. Although this method may solve the problem of weights becoming exceedingly large it also introduces a decay term in the derivative [16] which can lead to error terms decreasing with each iteration. Further, it forces the network to perform the classification task using small contributions from every feature rather than allowing important features to predominate.
Chapter 3

Classification of Cytological Smears

3.1 Cancer

Cancer is the general name applied to a group of over 100 disease-states [43] characterized by uncontrolled cell growth and the ability of cells to invade other tissues. Normal cells divide through mitosis, producing two cells genetically identical to the original. The blueprint for cell division is contained within the DNA in the nucleus of the cell. The DNA can become damaged due to the presence of certain chemicals, viruses, high frequency radiation or other environmental stresses. When this happens, the normal cell growth and division behavior can be altered. If the portion of the DNA responsible for growth regulation is changed, the cell may divide more rapidly than the surrounding tissue. The next generation of cells produced by the cell will tend to have unstable DNA. Through mitotic errors and further environmental stresses, the errors in the DNA of the succeeding generations of the original cell may multiply.

The overall behavior of this population is difficult to predict. While still localized around the site of the original cell, the population is referred to as a premalignant (pre-cancerous) lesion. The lesion may progress to some stage and then halt. If the growth halts, it may then remain static for the life of the bearer or actually regress, as normal cells surrounding the lesion outgrow those that make up the lesion. The lesion may also grow and spread to other areas of the body. This type of lesion is termed invasive cancer.
3.2 The Epithelium

Epithelia are tissues that line the surfaces of organs or glandular tissue characterized by close packed cells and a free surface. They protect the underlying tissue physically and chemically. One of the most common types of epithelial tissue is the squamous epithelium, which is the name for most epithelia in direct contact with the external environment. This includes the skin, mouth, esophagus, vagina and uterine cervix. Cells of uterine cervix are of particular interest to this thesis.

Figure 3.1 shows a schematic representation of stratified squamous epithelial tissue. There are four types of cells that make up the stratified squamous epithelium. The **basal** cells form the base of the epithelium. In healthy tissue, they are the only cells that are able to divide. Their cytoplasm tends to be smaller than the other type of cells and have a large nuclear area compared to the surrounding cytoplasm.
The basal cells divide periodically, and the new cells are pushed towards the surface. These cells enlarge to become parabasal cells. Their cytoplasmic area is approximately twice that of the basal cells and their nuclear area is around 50% larger.

The third type of cell is the intermediate cell. Intermediate cells are between three and six times larger than basal cells. Like the parabasal cells, their nuclei are 50% larger than that of the basal cells, and they may be oval in shape. They are the most common type of cells in the epithelium and are the most useful for diagnosing the condition of the tissue.

The cells at the surface are the superficial cells. As intermediate cells are pushed to the surface, they grow in size and differentiate to become the superficial layer. The nucleus condenses into a very small size in the superficial layer. Such nuclei are called pyknotic (dense) nuclei.

3.2.1 The Nucleus

The main component of the nucleus observed in stained cell samples is the nuclear chromatin. These structures are made up of DNA and associated proteins. The quantity of chromatin is the measure of the amount of DNA in the nucleus. DNA content is one of the most important methods for identifying cancerous cells.

In basal cells the chromatin has a coarse structure. Parabasal and intermediate cells have granular chromatin. As the nucleus condenses during the formation of superficial cells, the chromatin appears very dark and little structure is seen. Table 3.1 summarizes differences between the types of cells that make up the squamous epithelium.
Chapter 3. Classification of Cytological Smears

3.3 Differences Between Normal and Malignant Cells

When cells become malignant, there are usually changes in the cytoplasm and nucleus that are noticeable when viewed through a microscope. These changes depend on both the type and location of the cancer. Detecting them accurately is important for both diagnosing the state of the tissue and determining the prognosis for the patient.

Cancer cells tend to have a greater variability in their size when compared to normal cells of the same cell type. However, cells can become variable from benign disorders, non-malignant viral infections, or other causes. Therefore, using the cytoplasmic area is not a reliable method to diagnose the tissue. In squamous epithelial tissue, normal cells vary between 15–60μm in size, as is shown in Table 3.1. This makes it even more difficult to determine whether a cell is abnormally large, since the normal variability in the size of the cells making up the epithelium is already a factor of four.

A population of cancer cells will have larger size variations than a population of normal cells. When viewing a tissue section, the structure of the tissue can be determined (as in Figure 3.1). In this case it is possible to use the presence or absence of size variations among the cell population to aid the overall diagnosis.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Basal</th>
<th>Parabasal</th>
<th>Intermediate</th>
<th>Superficial</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>8–10μm</td>
<td>15–20μm</td>
<td>30–60μm</td>
<td>40-60μm</td>
</tr>
<tr>
<td>Shape</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polygonal</td>
<td>0%</td>
<td>5%</td>
<td>85%</td>
<td>75%</td>
</tr>
<tr>
<td>Oval</td>
<td>5%</td>
<td>40%</td>
<td>10%</td>
<td>20%</td>
</tr>
<tr>
<td>Round</td>
<td>95%</td>
<td>55%</td>
<td>5%</td>
<td>5%</td>
</tr>
<tr>
<td>Nuclear Diameter</td>
<td>7–9μm</td>
<td>8–13μm</td>
<td>10–12μm</td>
<td>5–7μm</td>
</tr>
<tr>
<td>Nuclear Shape</td>
<td>Round</td>
<td>Round or oval</td>
<td>Round or oval</td>
<td>Round</td>
</tr>
<tr>
<td>Chromatin Pattern</td>
<td>Coarse</td>
<td>Granular</td>
<td>Finely granular</td>
<td>Pyknotic</td>
</tr>
</tbody>
</table>

Table 3.1: Descriptive differences between the cells that make up the squamous epithelium.
Nuclear abnormalities are the most noticeable differences between cancer cells and normal cells. The amount of DNA in the nuclei usually increases with the progression of the lesion. The area of the nucleus is roughly proportional to the amount of DNA contained therein, so doubling the amount of DNA usually causes the nuclear area to double. It is not uncommon for nuclei of malignant cells to be four times the size of normal cells taken from the same site.

The nuclei of normal cells undergoing mitosis will temporarily increase in size as DNA synthesis occurs. Just before division these cells will contain twice the DNA of normal, nondividing cells. Consequently, the absolute size of the nucleus or DNA content is not an exact characteristic of a malignant cell. It is necessary to consider the DNA content of suspected malignant cells in relation to that of the population of normal cells.

Normal cells have round or oval nuclei with no bumps or indentations. Irregularities such as notches in the perimeter of the nucleus or protrusions are common indicators of cancer cells.

The texture of cancer cell nuclei tend to be coarser than that of normal cells. Stained cancer cells tend to have dark nuclei, which is termed hyperchromasia. Chromatin in these cells aggregates into clumps, giving the appearance of dark spots in the nuclei when viewed under a microscope.

3.3.1 Staging of Malignancies

All of the characteristic differences between normal and premalignant cells described above are used to assess the severity of a precancerous lesion. It is important to note that the existence of a precancerous lesion alone, no matter what its severity, does not automatically imply a poor prognosis for the patient, if left untreated. What is important is that lesion does not progress to a stage where it spreads to the surrounding tissue.

In order to succinctly characterize the stages of premalignancies, oncologists have
attempted to develop a standard nomenclature for epithelial lesions. Several different staging systems are used by cancer specialists to describe the tissue state. Staging systems group precancerous tissues into classes based on the degree of change of the cellular cytoplasm, nuclei and tissue organizations from normal appearance, called the degree of dysplasia. Such tissues are referred to as being dysplastic.

As a lesion progresses, it is thought that the cells that make up the lesion pass through three stages: mild, moderate and severe dysplasia. If a lesion progresses beyond this stage, but has not yet spread to another site it is referred to as carcinoma in situ (carcinoma=“cancer of the epithelium”, in situ=“localized”).

A cancer spreads by breaking through the base of the epithelium and sending cells through the blood or lymphatic system to other parts of the body. This condition is called metastasis. These cells may then take hold in other organs creating new cancerous populations which do not perform the regular functions of the cells native to that site. This accounts for the deadly nature of a cancer. If a cancer spreads away from the site of origin then it is called metastatic carcinoma which is a full-blown cancer.

Lesions that fall under the category of mild dysplasia are characterized by small morphological changes of the epithelial cells. They are “low grade” lesions in the sense that 60% to 70% of them will naturally regress when left untreated [22]. The cells tend to have slightly enlarged nuclei, and their nuclear chromatin, although mostly fine grained, is just beginning to form clumps. The nucleus is still round or oval in shape and any protrusions or indentations in the nucleus should be small.

Moderately dysplastic cells usually have significantly larger nuclei than normal cells. Dark spots can be seen in the nucleus as the chromatin clumps together. There may be significant irregularities in the shape of the nucleus.

Lesions that are severely dysplastic are also known as “high grade” lesions since most will progress to invasive cancer. Severely dysplastic cells have very large nuclei or a high
nuclear area to cytoplasm ratio compared to normal cells from the same site. The interior of the nucleus will usually have large clumps of chromatin and the nuclear shape is often irregular. In some cases, the nucleus appears to be in the midst of splitting into more than one lobe. These types of cells are easier to identify than cells with other grades of dysplasia.

The descriptions of the different stages of dysplasia are summarized in Figure 3.2. The figure also shows normal, mildly dysplastic, moderately dysplastic and severely dysplastic cell nuclei. These nuclei were stained and scanned using the automated image cytometer system described in Section 3.5.

In general, there is some difficulty in categorizing cells into the different classes. There is a continuum of changes that occur as a lesion progresses. These changes cannot be adequately represented using a few descriptive labels. As well, the definitions of the stages of dysplasia are vague enough that there is some overlap between the classes. It is common for two different cytopathologists to classify the same object into different classes. This can occur even if they confer on the diagnosis, as each specialist’s opinion will depend on what their past experience and training have taught them. For this reason, there is some overlap between what is called benign/mild, mild/moderate, and moderate/severe. There are fewer instances of disagreement of diagnosis where the diagnoses are two or more classes apart.

3.4 Malignancy Associated Changes

Normal cells in the presence of a malignancy undergo subtle changes which alter some of the characteristics of the nucleus. Nieburgs [34, 35, 36, 37] described changes in the nuclei of normal cells taken from blood and liver tissue when a tumor was nearby. These changes were termed Malignancy Associated Changes (MAC). The term MACs or MAC
STAGING OF PRECANCEROUS LESIONS

NORMAL INTERMEDIATE CELL
- Nucleus is round or oval in shape
- Chromatin is uniformly distributed throughout the nucleus

MILD DYSPLASIA
- Nucleus is slightly enlarged or irregular in shape
- Nuclear chromatin is just beginning to form clumps

MODERATE DYSPLASIA
- Significantly larger nucleus than normal cells; irregular shape
- Distinct clumps of chromatin are seen within the nucleus

SEVERE DYSPLASIA
- Nucleus is large, irregularly shaped and appears dark due to extra amounts of chromatin
- Chromatin tends to be seen in large clumps
- Nuclear area is very large compared to the surrounding cytoplasm (not seen in photo)

Figure 3.2: The differences between the different stages of precancerous lesions are shown. The images on the left are the nuclei of typical cells of the four types scanned using the automated image cytometer described in Section 3.5.
cells then denotes apparently normal cells which have undergone such a change.

The changes that were observed were difficult to detect by eye. Some question existed as to whether the observed changes were artifacts of the cell samples. Since then, careful cytometric measurements have been done by different groups [7, 21, 50] on normal cells and cancer cells to reveal significant changes in quantitative nuclear features. Cytometric measurements of MACs have been made in colon, lung and cervical carcinomas. They have been detected up to 10cm [33] away from the site of a carcinoma.

It is believed that MACs occur because of growth factors emitted by tumor cells. The growth factors are proteins that are absorbed by the normal cells around the lesion and enter the nuclei where they instigate changes in the nucleus. Cytometric measurements of the nuclear features have shown differences in the nuclear roundness [4], nuclear area, and DNA content [31] of MACs versus normal cells from normal-diagnosed patients. It has also been noticed that there is a difference in the chromatin distribution between normal cells and MACs taken from in situ cervical carcinomas, where the chromatin in the MACs was noticed to be closer to the nuclear periphery [30].

Since MACs are difficult to detect while abnormal cells are fairly easy to detect, it is natural to question why one would bother to search for MACs in potentially malignant tissue. When a cell smear is taken, the cells are scraped off one area of the epithelial tissue. If this area does not contain any abnormal cells, there is the possibility that a patient with a premalignant lesion or cancer will be diagnosed as normal. Since the range for MAC expression has been measured to be 10cm in some cases, it is possible that a cancer can be detected using only the apparently normal cells from the cell smear. Thus, a cytological classifier which includes the detection of MACs may be able to diagnose the existence of a lesion that would otherwise be missed.
3.5 Cervical Cytology and Cytometry

Cervical cell smears are obtained by scraping the surface of the cervix with a device similar to a spatula. The spatula is then smeared over a slide and the cells are allowed to air dry or are fixed to the slide by dipping the slide in a fixative solution such as 95% ethanol. A smear generally contains over 100,000 objects including epithelial cells, cellular debris, leukocytes (white blood cells), macrophages, microorganisms, etc.

When nuclear features are being used to diagnose the cells, the slide is stained with a stoichiometric nuclear stain, where the amount of stain absorbed by the nucleus is proportional to the amount of DNA in the nucleus. This gives the nuclei high contrast against the cytoplasm.

The number of overlapping objects on the slide depends largely how the smear has been applied to the slide. This has consequences with respect to the ability of cytopathologists or cytotechnologists to identify premalignant or malignant cells. If the nuclei of two cells overlap, then the technologist may see what appears to be a very dark nucleus, or a large nucleus with an abnormal protrusion. It may be possible for the technologist to resolve the overlap, but an automated cytometric device would likely decide that the object is an abnormal cell.

The cytologist renders the diagnosis for the slide by examining areas throughout the slide under a microscope. Several thousand objects must be examined in order to completely view the slide. The diagnosis of the slide is determined by the most severe level of dysplasia of any cell on the slide.

An automated image cytometer developed by Xillix Technologies Corporation and the Cancer Imaging Department of the B.C. Cancer Agency [15] can serve as an aid to the cytotechnologist by automatically scanning the slide collecting cell images. The cytometer consists of a digital camera which uses a scientific CCD array containing 1.4
The current automated diagnostic system uses discriminant functions calculated with over 70 available nuclear features. Fifty-three of these features are used in this thesis for the separation of MACs from true normal cells. The features consist of shape features, DNA quantity features, Markovian nuclear texture features, and discrete nuclear texture features. A complete description of these features can be found in Calum MacAulay's thesis [25].

Some of the features are those that have been demonstrated in the past to be useful in separating normal cells from abnormal cells. The integrated optical density (IOD) feature, which is the measure of the amount of chromatin in the nucleus is the most commonly used quantitative nuclear feature. It is calculated by summing the logarithm of the intensity ratio between each pixel of the segmented image and the background intensity. This raw IOD value is then normalized by dividing it by the mean IOD of the intermediate cell nuclei on the slide. Normal, non-dividing cell nuclei should have a normalized IOD near 1.0 while those undergoing division should have a normalized IOD near 2.0.

The IOD and nuclear area are powerful features for separating normal from abnormal cells. This is to be expected given the changes that occur when cells become malignant. For MACs however, the changes are difficult to distinguish by eye. It is not clear which features will give the best discrimination for this decision process.
3.6 Design of a Cytological Classifier

There are several components to consider in the design of a cytological classifier system: decision strategy, model selection, data selection, training and testing/validation. Each component is important because it affects the accuracy and usefulness of the final product.

The decision strategy addresses how the overall system will treat the information on the slide. Of the thousands of objects on a typical slide specimen, roughly 50% of the objects convey no useful information. These objects consist of cellular debris, overlapping cells, white blood cells and other tissue components that do not aid the overall diagnosis. Since the final diagnosis is made from the diagnostic cells, these cells must be efficiently separated from the other material.

The decision strategy will contain rules to separate the objects into subgroups, and can be represented as a tree-like process. The tree shown in Figure 3.3 is the decision tree process used by the automated classifier system at the Cancer Imaging Department of the B.C. Cancer Agency. The decision at each node is performed by a combination of thresholds and a discriminant function. The second decision, which splits objects into three groups based on integrated optical density, is performed by applying a threshold to the normalized IOD nuclear feature. An object may pass through as many as seven decision nodes in the process of being classified.

The diagnostic cells consist of the normal and abnormal cells on the slide. These make up only a fraction of what will be scanned by the cytometer. Further, it is necessary that the cells be in good focus and are segmented correctly by the image acquisition software. If the segmentation is incorrect, or the focus blurred, the values of the nuclear features will be affected. The orientation of the cell on the slide also affects its nuclear features. A cell that is folded or merely viewed on edge will have an apparently smaller nuclear area and irregular nuclear shape as compared to the same cell when viewed face on.
Figure 3.3: The decision tree used by the automated classifier system at the B.C. Cancer Agency Imaging Department. Each node represents a decision currently made using a discriminant function or threshold.
Since there may be hundreds or thousands of cells with diagnostic value on a slide, how many should be used in the decision process? If a large number of cells is used, then adding more cells to the decision process should not affect the slide diagnosis. The question is then how to determine what number of cells is required to adequately represent the slide distribution as a whole. It has been the experience of researchers in the Cancer Imaging Department that no significant improvement is made in a slide diagnosis when classifications using more than 1000 cells are compared with the classification made using around 1000 cells [26].

Model selection refers to choosing the right size model to solve the problem. The simplest decision model is the application of a threshold on a single feature, such as the clustering of objects by IOD in Figure 3.3. Choosing the right model requires some information about the classes of objects being separated. If the classes are linearly separable, multivariate regression or linear discriminant functions are appropriate. If the data being used is not linearly separable, a higher order discriminant function or neural network will provide a nonlinear separation between the two classes of objects. Generally, as more parameters are added to the model, the resulting classifier becomes less robust to variations in the feature parameters.

The problem of representability is also hidden in the model selection component of the classifier. We assume that the measured features are adequate to provide discrimination between the classes. If this assumption is incorrect, we may infer erroneous conclusions from the results about the true nature of the classes.

As an example, imagine designing a classifier to separate apples from oranges using the basic features of volume and mass. These features, obviously, are not the best ones to use to distinguish the two classes. With a little enterprise, it might be possible to design a non-linear classifier which could correctly classify 75% of the training and test data. If our approach to the model selection was systematic, we may find that using
a simpler or more complicated model only reduces the classifier performance. It would be a mistake to conclude that the classes are only 75% separable, since introducing the feature of colour to the model would certainly improve the performance of the model to nearly 100%.

This situation is analogous to the one of detecting malignancies on the slide based on MACs. Since the changes to the normal cells in the presence of malignancies are not detectable by eye, there is no way to determine whether the features we measure completely describe the differences between the populations. In the case of the apple/orange classifier, colour is an obvious addition to the feature set. With MAC classifiers, optimal feature selection is not readily apparent.

When selecting the data used to design the classifier, it is necessary to ensure that the slides providing the sample cells be representative of all the slides to which the classifier will be applied. The slides are stained with a stoichiometric DNA stain that makes the nuclei stand out from the cytoplasm. There is batch-to-batch variation of the apparent darkness of the nuclei when slides are stained due to variations in the concentrations of the stain components. The coefficient of variation in the slide-by-slide normalized IOD distribution of intermediate cells within has been recently measured to be approximately 7% [46] for cervical smears prepared with Feulgen-Thionin stain. This variation is large enough that slides should be selected from many different staining batches when designing a classifier in order to obtain robust performance.

The MAC classifier is trained using intermediate cells that have been selected by an experienced technologist. The resulting decision process is validated using an independent test set also selected by the technologist. If the training data has been representative of the data that is encountered in everyday use, one can be confident that the validation performance is an accurate representation of the classifier performance on intermediate cells in general.
However, as Figure 3.3 shows, the objects to which the MAC classifier is applied are those which have successfully passed through the five or six previous decision steps. Some objects that are not intermediate cells are likely to make it through these discriminant functions. This will reduce the true performance of the MAC classifier when used as a part of the hierarchical system. Thus it is necessary to seek the best possible decision rule at each node of the tree.
Chapter 4

Experiments and Results

4.1 Description of Data

The data used in this thesis were taken from cervical cell smears obtained from the B.C. Cancer Agency. The cell nuclei used to design the cell-by-cell classifier came from 334 slides from premenopausal patients. A total of 190 slides diagnosed "normal" and 144 slides diagnosed "severely dysplastic" were used. Each slide was individually diagnosed by three or four independent cytotechnologists. Only the slides for which three of the diagnoses were identical were candidates for use in this study.

The slides were stoichiometrically stained with a Feulgen-Thionin stain, which is a quantitative DNA stain. To prevent batch-to-batch variations in the average staining intensity of the slides from affecting the results, slides were selected from different staining batches. This ensured that the training data would be representative of data normally scanned by the automated image cytometer.

There were a total of 292 normal slides available for use in the study, 283 of which were stained in 24 different staining batches. A total of 190 normal slides were used in the study and were taken from 14 of the 24 staining batches. Similarly, there are a total of 160 severe slides available from 21 different staining batches. Data was acquired from 144 of these severe slides which spanned the 21 staining batches.

The slides were scanned by an automated image cytometer designed by Xillix Technologies Corporation and the B.C. Cancer Agency. Each slide contains hundreds of
thousands of objects, including cells, debris and connective tissue. A decision process involving thresholds and discriminant functions was applied to each scanned object to separate out the obvious junk from the collected objects, leaving possible cells. The computer program controlling the cytometer was configured to collect 900 possible cells per slide.

The discriminant function and neural network cell-by-cell classifiers were trained using intermediate cells taken from normal and severe slides. The intermediate cells from normal slides will be referred to as negatives and the intermediate cells from severe slides will be referred to as MACs. It was not practical to use all available cells to design the classifier. Hence, it was necessary to restrict the number of cells selected from each slide. A maximum of 75 negatives and 150 MACs were selected from each of the respective slides for use in the design of the cell classifier. These limits were imposed to prevent a slide that had a high intermediate cell count from being overrepresented in the design data. Intermediate cells make up a smaller fraction of the cells scanned from the severe slides than scanned from normal slides. It was necessary to accept twice as many MACs as negatives to provide adequate sized samples of each class of data.

Each object was examined five times to determine which objects were intermediate cell nuclei in good focus. Only those cell nuclei that satisfied these criteria were included in this study. Consequently, the number of available cells varied from slide to slide. Table 4.1 shows the frequency of the number of intermediate cells in good focus per slide used in the design of the cell-by-cell classifier. Most of the normal slides had more than the maximum 75 cell limit. Only 47 of 144 severe slides had 150 intermediate cells among the 1000 scanned objects.

A total of 25360 cells were used to design the cell-by-cell classifier, 12680 from each class. One third of the data was used for training each classifier and the remainder was used for testing. Table 4.2 provides a breakdown of the number of cells of each type used
Table 4.1: Frequency of the number of cells per slide used in the design of the cell-by-cell classifier in the study.

<table>
<thead>
<tr>
<th>Negative slides</th>
<th>Severe slides</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of cells</td>
<td>Number of slides</td>
</tr>
<tr>
<td>from slide</td>
<td>from slide</td>
</tr>
<tr>
<td>1–10</td>
<td>7</td>
</tr>
<tr>
<td>11–20</td>
<td>6</td>
</tr>
<tr>
<td>21–30</td>
<td>7</td>
</tr>
<tr>
<td>31–40</td>
<td>3</td>
</tr>
<tr>
<td>41–50</td>
<td>3</td>
</tr>
<tr>
<td>51–60</td>
<td>2</td>
</tr>
<tr>
<td>61–74</td>
<td>15</td>
</tr>
<tr>
<td>75</td>
<td>147</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>190</strong></td>
</tr>
</tbody>
</table>

Table 4.2: The breakdown of the number of cells of each type used to train and test the cell-by-cell classifiers is shown.

<table>
<thead>
<tr>
<th>Cell type</th>
<th>Training</th>
<th>Testing</th>
<th>Total cells of type</th>
</tr>
</thead>
<tbody>
<tr>
<td>negative</td>
<td>4228</td>
<td>8452</td>
<td>12680</td>
</tr>
<tr>
<td>MAC</td>
<td>4228</td>
<td>8452</td>
<td>12680</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>8456</strong></td>
<td><strong>16384</strong></td>
<td><strong>25360</strong></td>
</tr>
</tbody>
</table>

The data were divided into the two groups by selecting every third cell for inclusion in the training set. This is shown schematically in Figure 4.1. This method of selecting the cells prevented any slide from being overrepresented in either the training or the test dataset. Since there are batch-to-batch variations in the staining intensity it is important to train on as wide as possible a cross-section of slides. The intensity differences between staining batches can be large enough that a classifier can be trained to discern intermediate cells from different batches reliably. This point will be demonstrated in
Chapter 4. Experiments and Results

Section 4.5.

Figure 4.1: Every third cell nucleus was selected for inclusion in the training data and the remainder of the data was used for testing.

4.2 Nuclear Features

Each cell nucleus used in this thesis was represented by a feature vector. Fifty-three of the 66 available features were selected for the classification experiments. The 53 features consisted of optical density, shape, Markovian and discrete texture features. A complete description of the features can be found in the MacAulay thesis [25].
The value of each feature was normalized by converting it to a z-score using

\[ z_j^i = \frac{x_j^i - \bar{x}^i}{\sigma^i} \]  

(4.1)

where \( z_j^i \) is the normalized value of the \( i^{th} \) feature of the \( j^{th} \) cell, \( \bar{x}^i \) is the sample mean for the \( i^{th} \) feature, and \( \sigma^i \) is the standard deviation for the \( i^{th} \) feature. The means and standard deviations were calculated for the training set population of 8456 cells.

Normalization, although not strictly necessary when using neural networks, is useful because it allows an easier interpretation of the important features from the strength of their corresponding weights in the network. Some of the features such as the nuclear area (AREA) have means of the order of \( 10^3 \) while others, such as the Markovian energy feature (ENERGY) have means of the order \( 10^{-3} \). When a neural network is being trained, the connection weights corresponding to important features are usually increased. The magnitude of the weights can then be used to indicate as to which features contribute to the performance of the network. If the inputs to the network differ in magnitude, the average magnitude of the inputs must be multiplied by the corresponding connection weights when examining the weights to determine the relative importance of the input features.

It is important to note that the strength of a connection weight for a particular input does not always give a measure of its importance. If there are correlations between features, the training procedure may assign a significant weight to inputs that provide only moderate discriminating ability for the network. We imagine training a network that had among its features two with a correlation of 1.0 and providing little discrimination ability. Through the initialization of the network and the particular training data used, the weights connecting these inputs to the hidden units could assume significant values. The weights connecting the input of the second feature to the hidden neurons would just be equal and opposite in sign to those connecting the first to the hidden neurons.
In this situation, the weight strengths alone are not enough to determine the importance of the features. The procedure that many authors recommend to determine the contribution of a feature input to a particular network is to remove the input and retrain the network. If the network performance can be recaptured, the input makes no significant contribution to this particular network. This procedure is time consuming though, as the network must be retrained after each input is removed.

This procedure does not address whether a feature is important to the classification problem in general. It only helps determine whether the feature is important to the network for the current combination of weights. A feature that can apparently be removed may prove to be a significant feature if the experiment is repeated with different initialization weights.

The most important reason for normalizing the feature values relates to the initialization of the networks for training. The training of the network is a multivariate optimization procedure that requires the network to be initialized with some (usually random) starting weights. The weights are initialized to small values so that the sigmoid activation function \( g(\ldots) \) in Equation 2.11 does not saturate to either zero or one.

The derivative of the error with respect to the input-to-hidden weights is calculated using Equation 2.17. This equation includes the derivative of the activation function, \( g'(h^u_j) = \sum_i w_{ij} x^u_i \), where \( x^u_i \) are inputs and \( w_{ij} \) are input-to-hidden-unit connection weights. When \( h^u_j \) has a large magnitude \( g'(h^u_j) \) is nearly zero (Figure 2.5). In this situation \( \frac{\partial E}{\partial w_{ij}} \) will be small and the update performed by Equation 2.19 will also be small.

As a result, the weights connecting the inputs to the \( j^{th} \) hidden unit will be changed very slowly by the optimization procedure. This hidden unit will make little contribution to the performance of the network. This situation is avoided by choosing weights such that \( h^u_j \) in Equation 2.10 is of order one. When the feature values are normalized, all
the weights can be treated in the same fashion. There is no need to consider the average magnitude of the inputs when initializing the weights.

4.3 Feature Correlations

The 53 nuclear features were used to train the neural networks without regard for any correlations which exist between the features. It is known from regression theory that using too many features in a model, at best, has no effect on the model and, in many cases, adversely affects the performance of the classifier. Neural networks are subject to these effects as well. Using redundant features increases the dimensionality of the feature space. This slows the optimization procedures from convergence to a satisfactory minimum.

Discriminant function analysis (DFA) does not suffer from this problem, though. With the stepwise addition rule used in DFA, features are added to the model one at a time, starting with an empty model. Each feature is temporarily added to the current model and the model is tested. The feature which gives the best performance when added to the current model is kept in the model for the next iteration. The procedure then repeats the test with each of the remaining features until the addition features no longer improves the performance of the classifier or meets the inclusion criterion.

There are 1378 pairwise correlations between the 53 features calculated for all the 8456 cells used for training the classifiers. Table 4.3 gives the frequency distribution of the absolute values of the correlations between the 53 features. Of particular interest are the correlations between features that are close to one. There are seven feature pairs with a correlation value between .95 and 1.0. Seven features likely could have been removed from the dataset without the potential for a significant loss of performance of the classifiers.
Table 4.3: The frequency distribution for the absolute values of the pairwise correlation between the 53 features is shown. Correlation values near 1 are of interest because they indicate variables which can be eliminated from the classifier model.

Removing features whose correlations with others are slightly lower is problematic. Since the differences between negatives and MACs are subtle, one is reluctant to throw away any feature which might provide some discrimination value. Feature selection for such problems is a weighty subject on its own and should be investigated separately in a thorough manner.

### 4.4 Training Method Selection

There are several things to consider when selecting a training method. The most obvious consideration is the speed of convergence of the method when training a network. Most authors report a rate of convergence an order of magnitude higher for their methods over standard back-propagation. The benchmark problems used to conduct the tests are usually not representative of real-world problems. In most such problems there is a true global minimum, for which the network weights exactly solve the training criteria. In real applications there is noise or errors in the training data and the problem does not have an exact solution for most neural network structures.

A second consideration is the reliability of the training method. Some algorithms are sensitive to the initial network weights. The error-weight space is full of spurious local
minima. When starting with random network weights, it is easy to become trapped in the region of such a minimum.

A common occurrence is for the weights of the network to grow without bound when near a spurious minimum. The training algorithm is able to make marginal improvements in the performance of the network on the training data by making tenfold increases in some of the weights. This problem is avoided in some methods, such as Quickprop, by limiting the maximum step size.

A third consideration for selection of the training method is the sensitivity of the training procedure to its input parameters. Most training techniques have input parameters which control step sizes, acceleration factors, momentum, etc. Much time can be spent merely tuning these parameters to provide satisfactory performance for a particular problem. If the number of network weights is changed or a different number of training examples are used it is necessary to readjust the parameters. The time spent finding training parameters rivals that spent finding a good solution to the problem.

Three different training procedures were evaluated for designing a MAC classifier: Jacobs' method [19], Polak-Ribiere conjugate gradient optimization [40], and the Quickprop algorithm by Fahlmann [12]. Training was performed on a group of 500 negative cells and 500 MACs, each represented by 20 features. Each method was tested for 1000 iterations using a variety of training parameters on 20 randomly initialized neural networks.

The networks had 20 inputs, two hidden units and a single output unit. In general, the structure of a network with 20 inputs, two hiddens and one output would be denoted as 20–2–1. The network was trained to give an output of zero for MACs and one for negative cells. This training criterion was used for all experiments for separating MACs and negatives.

Jacobs' method has five parameters which must be set for each training attempt. The best values for these parameters are determined by trial and error. When the
proper training parameters were selected it performed as well as the other two methods. However, selecting poor values for the parameters caused the method to produce networks with poor performance on the test data of 500 MACs and 500 negatives. Of the three methods examined, Jacob’s method was the most susceptible to spurious local minima.

Jacob’s method was used with several different combinations of input parameters. The magnitude of the vector containing the network weights was found for each neural network and compared to the network performance. In 14 of the 20 networks trained using Jacob’s method the magnitude of the final neural network weight vector was at least twice that of the average weight vector of the networks trained using Quickprop. The average performance of these 14 networks was 69% on the training data, while the performance of the remaining six networks was 72%. The neural networks with larger weights had a poorer performance than the others. This occurred because these networks were in the region of spurious minima. This behavior was also seen when the five input parameters were varied.

The conjugate gradient method is the simplest to use as it requires no input parameters, other than a stopping criterion, which all methods require. It starts with a set of Cartesian search directions which it adjusts to fit the local error-weight landscape. The method assumes that the error function is quadratic with respect to the variables [14]. This assumption can lead to the search directions becoming linearly dependent when the method is used for general optimization problems. This means that the algorithm ends up searching only a subspace of the weight space. The implementation of the conjugate gradient algorithm in Numerical Recipes [40] avoids this by limiting the number of iterations of the algorithm to a maximum of 200. The search directions can be reset to the original directions at this time and the procedure repeated.

Like Jacobs’ method, the conjugate gradient method was found to be susceptible to undesirable local minima when optimizing a randomly initialized network. This training
method was used in an identical experiment to the one performed with Jacob's algorithm. The network weights of eight of the 20 neural networks grew without bound. They grew to as large as 50 times the average magnitude of the weights of the optimal neural networks obtained with the Quickprop algorithm. Once in the neighborhood of a minimum, however, the algorithm performed as well as the Quickprop algorithm for convergence. This algorithm appears to be useful when used in conjunction with other training methods.

The Quickprop algorithm has two parameters ($\eta, \mu$) to set when training a network. The algorithm's author recommends using the maximum step size, $\mu=1.75$ for most problems [12], leaving just one parameter to adjust. This had the fastest rate of convergence and performed the most reliably of the three tested. When the previous experiment was repeated with this algorithm, the average network performance was 72%, which was achieved by only a few of the networks trained with the other methods. Consequently, it was used for all subsequent classifier design attempts.

One way to prevent the network weights from growing without bound is to implement the weight decay formula (Equation 2.31). After each iteration of the algorithm, new network weights are multiplied a factor $(1-c)$. This scheme is used to prevent the network weights from becoming too finely tuned to the individual training examples. Only weights which receive reinforcement from many training examples should survive the decay process.

Ultimately, weight decay is used because it should lead to better generalization by the network. This can be tested by training a network with different levels of weight decay and checking the disparities between the training and test set performances. The classification rate for the test set should be close to that of the training data if the generalization ability of the network is good.

This test was conducted for the training and test sets of 1000 cells described earlier.
The Quickprop algorithm was used with 15 randomly initialized networks. The learning rate was set to $\eta = .1$ and each network was trained for 5000 iterations of the algorithm.

Table 4.4 shows the results of this experiment for four weight decay levels: $\epsilon = 0$, $\epsilon = 10^{-4}$, $\epsilon = 10^{-3}$, and $\epsilon = 10^{-2}$. There is a slight reduction in the performance on the training data as the level of weight decay is increased. The test set classification rate does not improve significantly as the decay is increased.

<table>
<thead>
<tr>
<th>Weight decay ($\epsilon$)</th>
<th>Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Training Set</td>
</tr>
<tr>
<td>0</td>
<td>70.9%</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>70.0%</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>68.0%</td>
</tr>
<tr>
<td>$10^{-2}$</td>
<td>67.5%</td>
</tr>
</tbody>
</table>

Table 4.4: The average performance of 15 neural networks with 20–2–1 structure trained with Quickprop on 1000 cells (500 MACs, 500 negatives) is shown for different weight decay levels. The test results for an independent set of 1000 cells should be closer to those of the training data for larger decay levels.

This experiment was repeated by increasing the number of training iterations from 5000 to 20000 with similar results. There was no significant difference between the performances of those networks trained with weight decay versus those trained without weight decay. The effect of weight decay should be more apparent in larger networks, where the larger number of free parameters should lead to overfitting of the training data. When the network was enlarged to 20–3–1 for this problem, the training set performances were all around 72% while the test set performances remained at 61% for the different weight decay levels. Thus, there were no strong results in favour or against the use of weight decay.

This result may not be scalable up to situations where more features are used. It is
certain that with 53 features, some of which highly correlated, the reduction or elimination of some of the weights is possible. The use of a simple weight decay factor, however, may not be adequate to obtain the best possible neural network solutions to the problem. It may be better to limit the size of the network and improve the test set performance by judicious elimination of features through other techniques. The remainder of the neural network classifiers obtained in this thesis were trained without any weight decay being applied to the network during training.

4.5 Spurious Classifications

The importance of the data selection method can be demonstrated by performing an experiment to detect the differences between negative cells from different staining batches. A set of 4000 negative cells were drawn from 66 slides, most of which were stained on six different dates in 1991. Roughly half the slides were stained on three of these dates, and half on the other three dates. The 4000 cells were divided evenly into two groups of 2000 cells: those stained in three batches in early February, and those stained later in February and March. The two groups were split into training and test sets by selecting every other cell for inclusion in the training data, similar to that shown in Figure 4.1.

A simple neural network with 53 input features, 1 hidden unit and 1 output (53-1-1) was trained to separate the two groups of 1000 cells of each class. The network was trained by performing 200 iterations of the Quickprop training algorithm. It correctly classified 82.3% of the 2000 training examples. The network was then applied to the test set of 2000 examples, where it had a correct classification rate of 80.6%.

The features of all scanned cells are normalized to prevent staining batch variations from affecting their values. Yet, a simple classifier can consistently separate over 80% of the negative cells from three staining batches versus three others. This indicates the
potential for obtaining spurious classifiers if one is not careful in the selection of the data for training and testing. If a larger network neural network had been used the classification rate would likely have been higher.

As a control, a related experiment was performed by selecting 2000 cells uniformly from across all staining batches the same 4000 cell group for a new training set. Half of the cells were randomly selected to be class 0 and remainder were called class 1. The remaining 2000 cells were also randomly divided into two artificial groups to form the test set. In this situation, there should be no differences between the two groups and it should be impossible to obtain a classifier that gives such remarkable results.

A neural network with structure 53–1–1 was trained for 200 and then a further 2000 iterations of the Quickprop algorithm. The network only had a performance of 55% after 200 iterations, so the network was optimized for a further 2000 iterations. After the 2200 iterations the training set classification rate was 56.2% on the 2000 examples. The test set performance was 49.4%, which is what one would expect from a completely random decision, i.e. a equal likelihood of assigning objects to either class. This result highlights the significance of the 80% classification rate of the previous experiment.

4.6 Discriminant Function Analysis

Discriminant function analysis was applied to the training set (Section 4.1) of 8456 cells. The function was obtained using the 7M module of the BMDP statistical software package. The software includes a routine for stepwise addition of variables for inclusion in the discriminant function. The resulting discriminant function uses 43 of the 53 features. It correctly separated 62% of the training examples (Table 4.5) and 61.6% of the test examples (Table 4.6).

A quadratic discriminant function calculated on 53 features has 1432 variables to
Chapter 4. Experiments and Results

<table>
<thead>
<tr>
<th>Cell type</th>
<th>Classification</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;MAC&quot;</td>
<td>&quot;non-MAC&quot;</td>
<td>Total</td>
</tr>
<tr>
<td>negative</td>
<td>2615 (61.8%)</td>
<td>1613</td>
<td>4228</td>
</tr>
<tr>
<td>MAC</td>
<td>1595</td>
<td>2633 (62.3%)</td>
<td>4228</td>
</tr>
<tr>
<td>Total</td>
<td>4210</td>
<td>4246</td>
<td>8456</td>
</tr>
</tbody>
</table>

Table 4.5: The linear discriminant function attained a 62.1% classification rate on the training set.

<table>
<thead>
<tr>
<th>Cell type</th>
<th>Classification</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;MAC&quot;</td>
<td>&quot;non-MAC&quot;</td>
<td>Total</td>
</tr>
<tr>
<td>negative</td>
<td>5142 (60.8%)</td>
<td>3310</td>
<td>8452</td>
</tr>
<tr>
<td>MAC</td>
<td>3178</td>
<td>5274 (62.1%)</td>
<td>8452</td>
</tr>
<tr>
<td>Total</td>
<td>8320</td>
<td>8584</td>
<td>16904</td>
</tr>
</tbody>
</table>

Table 4.6: The linear discriminant function attained a 61.6% classification rate on the test set.

enter in the discriminant model. Since there are only 8456 training examples, there is nearly one free parameter for every five examples. This has the potential problem of overfitting the data, so no quadratic discriminants were calculated for the MAC data.

4.7 Neural Network Experiments

Experiments were performed to train feed-forward neural networks with the training set to separate MACs from negatives. To find the network with the optimum number of hidden units, networks with different numbers of hidden units were used. The networks each had 53 inputs, one output and between one and seven hidden units. They were trained using the Quickprop algorithm.

The training procedure consists of optimizing the sum of squares error measure (Equation 2.14). For training purposes MACs have been designated class 0 and negatives have
been designated class 1. Each cell, represented as a 53 component vector, is used as the inputs to the neural network. The feed-forward mechanism is applied to the network and the output unit returns a continuous value between zero and one. The error contribution for each cell is just the difference between the designated class and the output value.

It is necessary to provide a stopping criterion to end the training procedure when no progress is being made. Normally, training is stopped when the network error is no longer decreasing with each iteration or when the length of the vector derivative of the error with respect to the weights drops below a certain limit. It was decided to train each network for a certain number of iterations of the training procedure. The number of iterations was selected to be large enough such that each network had sufficient time to reach a local error minimum. In the case of simple networks, with one or two hidden units, 1000 iterations sufficed. For large networks, with seven hidden units, an upper limit of 20000 iterations was used.

This upper limit was due, in part, to the amount of time required to perform 20000 iterations with a large network. During each iteration, the entire training set of 8456 cells must be presented to the network and the error derivative calculated for each cell. The training of a seven hidden unit network required more than 50 hours of CPU time on the HP Series 700 Apollo workstation used for these experiments.

Table 4.7 gives the number of training iterations used when training networks with different numbers of hidden units. It also lists the number of weights that are present with each network structure according to Equation 2.22. The largest network used had 386 free parameters. All networks attempt to optimize the fit of 8456 equations simultaneously. Thus, the ratio of training examples to free parameters is always kept above 20:1.

Each neural network was trained 20 times using different random starting weights. This was done in order to obtain an estimate of the average performance that could be expected from networks with such a structure. Each network was then tested using the
Table 4.7: The number of network weights and number of training iterations used are shown for networks with different numbers of hidden units.

<table>
<thead>
<tr>
<th>Number of hidden units</th>
<th>Number of weights in network</th>
<th>Number of training iterations used</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>54</td>
<td>1000</td>
</tr>
<tr>
<td>1</td>
<td>56</td>
<td>1000</td>
</tr>
<tr>
<td>2</td>
<td>111</td>
<td>5000</td>
</tr>
<tr>
<td>3</td>
<td>166</td>
<td>10000</td>
</tr>
<tr>
<td>4</td>
<td>221</td>
<td>20000</td>
</tr>
<tr>
<td>5</td>
<td>276</td>
<td>20000</td>
</tr>
<tr>
<td>6</td>
<td>331</td>
<td>20000</td>
</tr>
<tr>
<td>7</td>
<td>386</td>
<td>20000</td>
</tr>
</tbody>
</table>

There is large performance improvement when two hidden units are used. The training set performance rises to 72.9% while the test set performance is 71.6%. The difference between the training and test set performances increases as more hidden units are used. The training set results improve marginally with each extra hidden unit, but the test set performance peaks when four hidden units are used. When five, six or seven hidden units are used the average test set results actually decline. These results are shown graphically in Figure 4.2.
Figure 4.2: The cell-by-cell performance of neural networks with different numbers of hidden units is shown for the training and test sets. Each result is averaged for 20 runs using randomly initialized networks. The standard deviations are shown as error bars for each result.
Table 4.8: The average performance and standard deviations for 20 randomly initialized networks are shown when different numbers of hidden units are used. The training set performance improves as the number of hidden units is increased, but the test set performance peaks when four hiddens are used. The performance of the best network on the test data is given for each case.

The disparity between training and test set results grows as more hidden units are used. The significance of the difference between the training and test results can be seen when considering the standard deviation of each result. The error bars in the figure show the standard deviations of each average and are small compared to the difference between the two curves.

This can be understood in terms of the discussion of Section 2.2.7. A network with six or seven hidden units with the 53 input features has too many free parameters for this training set. Although the number of training examples is between 20 and 30 times the number of weights, the neural networks are still overfitting the training examples. This is due, in part, to an overlap in the feature space between the two classes, as was the case in the classification example of Chapter 2. When classifying MACs, the overlap is caused by misclassified cells among training examples and noise in the feature values. In both cases using too many hidden units causes the network to attempt to fit errors and noise in the training set. Figure 2.10 shows how this occurs for the classification...
example. This behavior is also present when six or seven hiddens were used for the MAC cell-by-cell classifier.

The classification percentages for the neural network and discriminant functions were obtained by choosing a threshold and applying it to the classifier output. In the case of a neural network that is trained to call objects either class 0 or class 1 it is normal to choose 0.5 as the threshold. This threshold can be set arbitrarily, allowing one to change the levels of false positives and false negatives.

In this study, MACs are designated class 0 while negatives are designated class 1. If it is the case that the threat of calling a negative cell "MAC" (false positive) is more dangerous than calling a MAC "negative" (false negative), the decision boundary can be moved to say, 0.6. Then all objects that score in the range 0.0 to 0.6 are called "MAC" reducing the number of false negatives at the expense of more false positives.

This information can be represented visually in a graph known as a Receiver-Operator Characteristic (ROC) curve. This graph displays the false negative versus false positive rate for the classifier. The ROC curves were calculated for the neural networks that gave the best test set performance for each choice of hidden units. Figure 4.3 shows the ROC curves for the discriminant function and neural networks with one, two and four hidden units. The ROC curves for the best networks containing three, five, six and seven hidden units nearly coincide with those of the two and four hidden-unit curves and are not shown.

The diagonal line in the figure represents the performance of a classifier which gives no discrimination (i.e. randomly assigning a class to each object). For example, calling all objects "MAC" would give a false positive rate of 100%, or calling all objects "negative" would give a false negative rate of 100%. The ROC curves for the better classifiers dip more deeply below this line, than do those for the poorer classifiers. The ROC curve for an ideal classifier would trace a path along the horizontal axis and up the vertical axis.
Figure 4.3: The ROC curves for the discriminant function and networks giving the best test set performance are shown. The ROC curves for the best networks using three, five, six and seven hidden units nearly coincide with those of the two and four hidden networks and are not shown.
Although the discriminant function and the neural network with one hidden unit are both linear classifiers, their ROC curves are different. The neural network converged to a set of weights that gives it a different behavior than that of the discriminant function. It gives a lower false negative rate when the false positive rate is allowed to be large. That makes this neural network more suited to the task of identifying MACs in a situation where the penalty for accidentally classifying a MAC as "negative" must be avoided. The discriminant function, however, performs better in a situation where misclassifying negatives as "MAC" has a more severe penalty. Its ROC curve is lower in the region where the false positive rate is constrained to be small.

The difference between these two classifiers is apparent when the weights that connect the inputs to the hidden unit are treated as a vector. The angle between this perceptron vector and the discriminant function vector can be found from their inner product. The angle between the vector of perceptron weights and the linear discriminant vector is $107^\circ$, which is $17^\circ$ from orthogonal. Yet, both classifiers score 61% on the test data. Thus, there exist nearly orthogonal hyper-planes that separate significant portions of the two classes. This argues that a nonlinear decision surface should give better separation between MACs and negatives.

The difference between the one hidden unit network and discriminant function vector classifiers can also be demonstrated by examining histograms of discriminant function and neural network values. Figure 4.4 shows a histogram of the discriminant function values for the test data. The decision boundary is marked by a discriminant value of zero. The MAC population has a slightly lower discriminant value than the negative population, which is seen in the small separation between the two peaks. This can be contrasted to the histogram of the output of the neural network with 1 hidden unit seen in Figure 4.5. Here the decision boundary giving optimal performance is at neural network output equal to 0.5. This classifier assigns most test examples to a network value around
0.4, which causes many negatives to be misclassified as “MAC”.

The narrow peak of the neural network histogram is the result of the application of the activation function $g(\ldots)$ to the sum of the inputs multiplied by the first weight layer. Since $g(\ldots)$ saturates for extreme positive or negative values, many test examples are grouped into the same bins at neural network values 0.4 and 0.85. If we remove the hidden unit and the activation function from this network, what remains is the classical perceptron. The histogram of the perceptron output is shown in Figure 4.6.

The appearance of the histogram of the neural network (Figure 4.5) and that of the perceptron are markedly different. The test examples with perceptron output between -20 and 0 are squashed by the activation function into a narrow neural network output range at 0.38. As well, the examples that receive a perceptron output between 7 and 20 are grouped at the neural network peak between .825–.85.

“Unsquashing” the neural network output has revealed an interesting result. The perceptron assigns negative examples to both extremes. This is not visible in either the ROC curve or histogram for the neural network because the bins used to calculate the histogram were too wide to resolve details in the peak. The ROC curve for the perceptron is shown in Figure 4.7. When the ROC curve for the network is replotted on a finer scale it is identical to that of the perceptron. This is expected since the activation function that is applied to perceptron output is monotonic.
Figure 4.4: The histogram of discriminant function values is shown for the test data. The discriminant function correctly classifies 61.6% of the test examples.
Figure 4.5: The histogram of neural network output values for the optimal network with 1 hidden unit is shown for the test data. This network correctly classifies 63.5% of the test examples.
Figure 4.6: The histogram of perceptron output values is shown for the test data. The perceptron assigns negative cells to extreme output values.
The ROC curve shows that the perceptron actually performs worse than chance when the false positive rate is constrained to be small. This occurs because more negatives are assigned a "MAC" score than true MACs as seen in the region between perceptron output -20 and -12 in Figure 4.6. It is necessary to apply a threshold in combination with the perceptron to obtain the best performance of the perceptron classifier. This threshold should be set such that all objects that receive a perceptron score less than this amount are automatically classified as "negative". The optimal threshold, calculated from the training examples occurs at a perceptron value of approximately -11.5. Setting the threshold at this value gives the highest correct classification rate for objects that receive a perceptron score less than -11.5.

The new classifier, which applies this threshold in combination with the perceptron, was applied to the training and test examples. It correctly classifies 69.2% of the training examples and 68.3% of the test examples. The ROC curve for this new classifier is also shown in Figure 4.7.

The histogram of the best neural network (of the 20 training attempts using this structure) using two hidden units is shown in Figure 4.8. There is a significant reduction in the number of negatives being called "MAC". This is seen by comparing the height of the negative peak around network values 0.2–0.3 in Figures 4.5 and 4.8. This difference accounts for much of the performance improvement over the perceptron and discriminant function classifiers. The ROC curve (Figure 4.3) for the two hidden unit classifier is significantly lower than the other classifiers in the region where false positives are kept below 50%. It approaches the performance of the perceptron for higher false positive values.

The histogram also shows frequency peaks for the negative test examples at network values 0.85 and 0.95. It appears that some feature differences exist among the population of negative examples. This trend is also seen in the histogram of network values for the
Figure 4.7: The ROC curves for the perceptron and the combination of the perceptron and threshold are shown. The ROC curves for the neural network with two hidden units is shown for comparison purposes.
Figure 4.8: The histogram of neural network output values for the optimal network with 2 hidden units is shown for the test data. This network correctly classifies 71.9% of the test examples.
optimal network using three hidden units shown in Figure 4.9. Here, the test examples are more evenly distributed among the two frequency peaks near 1.0 than in the former. The performance of the three hidden unit network for identifying MACs is similar to that of the former network. Both have a MAC frequency distribution that assigns most MACs a network value of around 0.3 and both tail off slowly to zero as the network value approaches 1.0.

The optimal network with four hidden units provides the most interesting histogram (Figure 4.10). Four distinct frequency peaks are seen at network values 0.12, 0.32, 0.8 and 1.0. These are labelled peaks 1 through 4 in the figure. Peaks 1 and 4 are the expected result of training the network to separate MACs and negatives. The first two processing layers of the network assign each training example a value (Equation 2.12) that is mapped into the range 0—1. This causes many MAC examples to be grouped together near zero and negative examples near one. Peak 4 occurs near 1.0 but peak 1 is found at a network value around 0.12. This is most likely due to an inability of the network to satisfy both the constraint of assigning negatives exactly 1.0 and assigning MACs a value of 0.0. This condition is more conspicuous with the simpler networks (Figures 4.5, 4.8, 4.9) where neither peak occurs at zero or one.

Peaks 2 and 3 are unexpected and require interpretation. Both consist of more than 1000 cells over three histogram bins, which is roughly 12% of the 8452 cells population of either class. A possible explanation is that the peaks are an artifact of the decision surface for the neural network. Peak 3, however, is seen in the simpler neural networks that use two and three hidden neurons as well.

Most of the 1300 negative cells that make up peak 3 come from 41 of the 190 normal slides. The histograms for these slides contain frequency peaks near 0.8 and have almost no cells near 1.0. Figure 4.11 shows the histogram of neural network output for the negative cell population from 149 slides after these 41 slides have been removed. The
Figure 4.9: The histogram of neural network output values for the optimal network with three hidden units is shown for the test data. This network correctly classifies 72.4% of the test examples.
Figure 4.10: The histogram of neural network output values for the optimal network with four hidden units is shown for the test data. This network correctly classifies 72.5% of the test examples.
Chapter 4. Experiments and Results

original histogram for all the negative cells is also shown. When the 41 slides are removed, the frequency peak near 0.8 is also removed.

The first explanation examined was whether staining differences can account for the unique distribution of neural network values for the cells on the 41 slides. Table 4.9 shows the breakdown of staining dates for the 41 slides. The right hand column lists total number of slides stained on this date that were used in the study. The table shows that the slides came from a total of seven staining batches, with a majority of the slides coming from three of the batches. For each batch, however, there are a significant number of slides that do not have the same characteristics as the 41 slides. If the peak were an artifact of staining, one would expect that almost all of the slides from that batch share that characteristic.

Figure 4.11: The histograms of neural network output values for the optimal network with 4 hidden units is shown for the whole negative test set and for the case where 41 slides have been removed.

Another hypothesis was that the scanning of the 41 slides by the automated image
Table 4.9: The staining dates of the 41 slides that make up peak 3 (see text) are shown. The total number of slides from each date is given in the rightmost column.

cytometer accounts for the differences between these slides and the rest. Examination of the scanning dates reveals that 40 of the 41 slides have a scanning date of 02/10/93. However, 133 of the 190 normal slides also have this scanning date. It was not possible for the automated cytometer to scan all 133 slides on one day. The scanning of these slides most likely occurred over the course of a week and all slides were given the same scanning date. It is unlikely that the configuration of the apparatus was changed during this time. The scanning dates and the ages of the patients (which were uniformly distributed) do not appear to provide a reasonable explanation for the phenomenon.

The two frequency peaks for negative cells were also present in the histogram of network output for the neural network using two hidden units. For this network it is possible to interpret the data visually. The two hidden units in this network act as a perceptron which are combined in a nonlinear expression to obtain the final neural network output. Each perceptron can be treated as a vector in the feature space. A scatter plot of the perceptron outputs for this network is shown in Figure 4.12. For clarity, only every eighth training example is plotted. The appearance of the graph would be similar if any other portion of the training data were plotted.
Figure 4.12: The outputs for every eight training example for the two perceptrons that make up the two hidden unit networks are plotted.
The figure shows that the negative examples are clustered at the extremes of the distribution. The examples that receive a large output value from perceptron 1 (rightmost distribution) make up the peak at 0.80 in Figure 4.8. The examples that receive a large output value from perceptron 2 (leftmost distribution) make up the peak at 0.95 in the figure. Although the figure demonstrates the existence of the two clusters, the reason for their existence remains unknown.

Peak 2 in Figure 4.10 is also difficult to explain. It occurs at a network value of around 0.32 for both the MACs and negatives. There does not appear to be a preferred staining date among the MACs that make up this peak. Roughly 1000 of 1300 cells in the region 0.3–0.4 come from 44 of the 144 slides. These 44 slides were stained on at least 11 different staining dates. Staining does not appear to be the common thread between these slides. Further, the slides which contain the cells that make up peak 2 also contain cells that make up peak 1.

Since there is a smaller peak of negatives at a neural network value of 0.3, it is possible that this peak is an artifact of the neural network. The weights for this particular network may have been optimized such that they reached an error minimum in a region of the weight space where more of the negatives were given a network value of 1.0 at the expense of increasing the average MAC value. This peak is not present in either of the histograms for the 2 and 3 hidden unit networks, making this explanation more likely.

A second explanation of this peak can be made by suggesting that there are two subpopulations of the MAC training data or that the MAC data is non-normally distributed. If the second subpopulation has features which more closely represent those of the negative cells the network will be unable to fulfill the constraints of assigning the negative population 1.0 and also assigning the “negative-like” MACs 0.0. This would also explain why there is a small negative peak at 0.3. These negatives could have features values very close to the peak 2 subpopulation and would therefore be misclassified into
the region near peak 2.

Such a subpopulation could exist if there were misclassified cells in the data. If the MAC expression in cells near a premalignant lesion is not assured (i.e. if it is possible to have a lesion where the intermediate cells around it do not display MAC characteristics), then much of the data which has been called MAC could be in error. Such a biological interpretation would require more analysis of MACs themselves and cannot be inferred from this study of statistical classifiers.

Another interpretation could be made by measuring to see whether the cells found in this peak are located further from the lesion than the cells of the same slide that were correctly assigned to peak 1. It has been mentioned that MACs have been detected 10cm from the tumor site in some organs. If this property varies or is subject to other limiting factors then some intermediate cells further from the lesion may not exhibit MAC expression. Nieburgs has observed two different types of MAC expression [6]. He has noted that MAC expression in intermediate cells that are in close proximity to a tumor differs from MAC expression in those cells that are distant from the tumor.

This study cannot be done for the cytological smears used in this thesis because when the physician scrapes cells from the cervical epithelium and smears it on a slide, all spatial relationships between cells are lost. This study could be done on biopsies, where the tissue is removed in a single piece, sliced and fixed on the slide. The distribution of the cells is maintained in such slides.

4.8 Principal Component Analysis

Principal component analysis (PCA) is a statistical technique for data analysis. It is a data reduction method that finds a ranked set of orthogonal vectors that accounts for as much of the variance of the data as possible. The goal is to find the smallest possible
set of vectors that spans the subspace occupied by the data. This is useful to make large multi-dimensional data sets more manageable for analysis. It is often the case that most of the variance of the data can be accounted for by the first few principal components.

The first principal component is designated to be the vector along which the variance of the data is greatest. The second principal component is then constrained to be the vector in the subspace perpendicular to the first that accounts for the greatest portion of the remaining data variance. This definition is applied in a similar fashion to all the remaining principal components until the entire space is spanned by the mutually orthogonal vectors. This is shown graphically in Figure 4.13 for a simple data distribution in two dimensions. The density distribution is shown as the shaded region and the two principal component vectors are shown beneath. The first principal component points along the direction of greatest spread of the data. In two dimensions, the second principal component is then completely determined and lies perpendicular to the first.

The principal components of a data distribution are obtained by calculating the covariance matrix for the distribution and finding the eigenvectors of this matrix. It turns out that the $k^{th}$ principal component is the eigenvector corresponding to the $k^{th}$ largest eigenvalue of the matrix. The magnitudes of the eigenvalues also give the relative contributions of each principal component vector to account for the variation of the data.

The principal components were calculated for the training set of 8456 cells. Table 4.10 shows the eigenvalues for the first 18 principal components. The contribution of each principal component to the total variation of the data is shown in the third column and the cumulative total in the fourth. The results reveal that the first 13 components account for 90% of the data variation and the first 28 components account for 99%. This demonstrates that the underlying structure of the data can be explained with 28 features made up of components for all 53 original features. This result is expected since there are large pairwise correlations between some of the 53 features. However, the result also
Figure 4.13: The two principal components are shown for an artificial data distribution in two dimensions shown by the shaded region. The first principal component lies along the direction of greatest variation of the data. The second component is constrained to be perpendicular to the first.

tells exactly how many orthogonal directions to use to represent the data to a specified accuracy. In this case we can represent each data point as a 28 component vector and only lose 1% of the expected variation of the data.

The first ten principal components account for 86.9% of the variation of the training data. The training and test data were transformed into ten component vectors using these ten principal components. The new data sets were used to train and test neural network classifiers to separate MACs and negatives. The neural networks were trained with the Quickprop algorithm as before using different numbers of hidden units. Each training run was performed 20 times using different randomly initialized networks and the results averaged. Table 4.11 shows the performance of the neural networks trained on the first ten principal components of the MAC dataset.
<table>
<thead>
<tr>
<th>Number</th>
<th>Eigenvalue</th>
<th>Percentage of total variation</th>
<th>Cumulative percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>161058.8</td>
<td>36.5</td>
<td>36.5</td>
</tr>
<tr>
<td>2</td>
<td>60906.4</td>
<td>13.8</td>
<td>50.3</td>
</tr>
<tr>
<td>3</td>
<td>34315.3</td>
<td>7.8</td>
<td>58.1</td>
</tr>
<tr>
<td>4</td>
<td>30629.8</td>
<td>6.9</td>
<td>65.0</td>
</tr>
<tr>
<td>5</td>
<td>25316.6</td>
<td>5.7</td>
<td>70.8</td>
</tr>
<tr>
<td>6</td>
<td>20038.1</td>
<td>4.5</td>
<td>75.3</td>
</tr>
<tr>
<td>7</td>
<td>17602.1</td>
<td>4.0</td>
<td>79.3</td>
</tr>
<tr>
<td>8</td>
<td>13403.9</td>
<td>3.0</td>
<td>82.4</td>
</tr>
<tr>
<td>9</td>
<td>11307.7</td>
<td>2.6</td>
<td>84.9</td>
</tr>
<tr>
<td>10</td>
<td>8644.4</td>
<td>2.0</td>
<td>86.9</td>
</tr>
<tr>
<td>11</td>
<td>6094.7</td>
<td>1.4</td>
<td>88.3</td>
</tr>
<tr>
<td>12</td>
<td>5801.5</td>
<td>1.3</td>
<td>89.6</td>
</tr>
<tr>
<td>13</td>
<td>5233.1</td>
<td>1.2</td>
<td>90.8</td>
</tr>
<tr>
<td>14</td>
<td>4959.3</td>
<td>1.1</td>
<td>91.9</td>
</tr>
<tr>
<td>15</td>
<td>4721.8</td>
<td>1.1</td>
<td>93.0</td>
</tr>
<tr>
<td>16</td>
<td>4398.5</td>
<td>1.0</td>
<td>94.0</td>
</tr>
<tr>
<td>17</td>
<td>3409.2</td>
<td>.8</td>
<td>94.7</td>
</tr>
<tr>
<td>18</td>
<td>2702.5</td>
<td>.6</td>
<td>95.3</td>
</tr>
</tbody>
</table>

Table 4.10: The first 18 principal components are shown for the MAC training data. They are obtained by finding the eigenvalues of the covariance matrix for 53 feature training set. The contribution of each principal component to the total variation of the data and the cumulative total are also given.
The results show that on average, the training set performance does not exceed 67% and the test set results peak near 64%. The reduced classification rate is expected since the classifier is only training on a subset of the full feature space. In this situation, the training rate is more reflective of the expected classification rate on independent data sets. The training rate is within 2% of the test rate in each case, which was the case for only the smaller networks trained on all 53 features (Table 4.8). This suggests that the networks being trained on all 53 features are being overfitted by the use of too many features.

Although, PCA is useful for reducing the dimensionality of the data it does not suggest which features are important for discrimination. The principal components point along the directions of greatest variance, but the features with the greatest variance are not necessarily those which differ between the MACs and the negatives. A better strategy for reducing the dimensionality of the data would be to examine the features individually for inclusion in the model and perform this in conjunction with PCA.
4.9 Slide-by-Slide Classification

It is the slide-by-slide classification rate of a classifier that is of interest to cancer researchers. This rate is more meaningful than the cell-by-cell classifier since it addresses the reliability of the classifier's diagnosis of the patient. In general, the cell-by-cell classifier is used as a decision step in the larger classification system, as shown in Figure 3.3. The slide-by-slide rate would normally be calculated for the whole system.

For comparison purposes, the slide-by-slide classification rates were obtained for the MAC cell-by-cell classifiers. The data used to train and test the slide-by-slide classifiers consisted of 93494 cells taken from a total of 395 slides. All normal and severe slides with at least 20 cells were included. There were 65645 negative cells taken from 251 normal slides and 27985 MAC cells taken from 144 severe slides.

The data used to train and test the cell-by-cell classifiers were a subset of the data used in the slide-by-slide classification. This introduces the possibility of a biasing error in the design of the slide-by-slide classifier. The cells used to design the best neural networks were used again in the slide-by-slide classification. This potential error is mitigated by the fact that the cells used to train the cell-by-cell classifiers comprised less than 10% of the total data. The statistical features used by slide classifiers are calculated based on all the available cells on the slide. Most slide datasets contain images of between 200 and 600 intermediate cells. The nuclear features of all of these cells are used to calculate the slide features. For some of these slides, up to 50 cells have been used in the cell-by-cell classifier study. Their influence on the value of each of the slide features is small and their effect on the results should be negligible.

The slides were split into two groups by selecting every second slide for inclusion in the training set. The training set consisted of 198 slides and the test consisted of 197 slides. Table 4.12 provides a breakdown of the number of slides of each type used to for
slide-by-slide classification.

<table>
<thead>
<tr>
<th>Slide type</th>
<th>Training</th>
<th>Testing</th>
<th>Total slides of type</th>
</tr>
</thead>
<tbody>
<tr>
<td>negative</td>
<td>126</td>
<td>125</td>
<td>251</td>
</tr>
<tr>
<td>MAC</td>
<td>72</td>
<td>72</td>
<td>144</td>
</tr>
<tr>
<td>Total</td>
<td>198</td>
<td>197</td>
<td>395</td>
</tr>
</tbody>
</table>

Table 4.12: Breakdown of the number of slides of each type used to design slide-by-slide classifiers.

When selecting features for the slide classifier, an obvious choice is the average classifier value assigned to the MAC and negative cells. This statistic (called AVGVAL) was selected as one of the features. A total of five features were calculated for the intermediate cells on each slide:

1. MAC: the fraction of the slide population with neural network value less than 0.5 or discriminant function value less than 0,
2. VMAC: the fraction of the slide population with neural network value less than 0.25 (not used for discriminant function),
3. AVGVAL: the average value of neural network output or discriminant function value,
4. STDDEV: the standard deviation of neural network values, and
5. SKEW: the skewness of the distribution of neural network values.

There is significant overlap in some of the slide feature definitions. For example, any slide which receives a large VMAC ("Very MAC") score will necessarily have a high MAC score, so these features will be highly correlated. As well, any of the slides described above
will have a low AVGVAL score, indicating a strong negative correlation of the AVGVAL feature to these other features.

Table 4.13 shows the correlation between the five slide features for the slide training set calculated using the five hidden unit network. The features MAC, VMAC, AVGVAL and SKEW have large pairwise correlations. The STDDEV feature has a small correlation to these variables. This behavior was observed for the slide features calculated using the other neural networks as well.

<table>
<thead>
<tr>
<th>Feature</th>
<th>MAC</th>
<th>VMAC</th>
<th>AVGVAL</th>
<th>STDDEV</th>
<th>SKEW</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>1.0</td>
<td>.92</td>
<td>-.98</td>
<td>.14</td>
<td>.84</td>
</tr>
<tr>
<td>VMAC</td>
<td>—</td>
<td>1.0</td>
<td>-.91</td>
<td>.06</td>
<td>.78</td>
</tr>
<tr>
<td>AVGVAL</td>
<td>—</td>
<td>—</td>
<td>1.0</td>
<td>-.20</td>
<td>-.90</td>
</tr>
<tr>
<td>STDDEV</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>1.0</td>
<td>.35</td>
</tr>
</tbody>
</table>

Table 4.13: Correlation values of the five slide features are shown for the five hidden unit cell-by-cell classifier. These values were calculated for the slide training set.

Table 4.14 shows the results of performing slide-by-slide classification using thresholds calculated for individual features. The results were obtained by setting a threshold using the training slide set and recording the classification rate of this threshold when applied to the test slide set. The STDDEV feature provided almost no discrimination ability when used alone or in conjunction with the other features and was not included in the table.

All the neural networks with two or more hidden units scored between 74% and 77% for correct slide classification for at least one of the slide features. This is a significant improvement over the perceptron and discriminant which both gave approximately a 69% correct classification rate. The performance of the thresholds on the training slides was always 1–2% higher than the test results recorded in Table 4.14. Each score is sensitive to the threshold position, varying by up to 2% when it is shifted by a small amount. This
Chapter 4. Experiments and Results

Table 4.14: The performances of the slide features for classification of the test set of 197 slides are shown for the best neural networks and discriminant function.

<table>
<thead>
<tr>
<th>Number of Hiddens</th>
<th>Discriminant</th>
<th>MAC</th>
<th>VMAC</th>
<th>AVGVAL</th>
<th>SKEW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>69.0%</td>
<td>—</td>
<td>69.5%</td>
<td>66.0%</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>68.6</td>
<td>—</td>
<td>69.0</td>
<td></td>
<td>63.6</td>
</tr>
<tr>
<td>3</td>
<td>75.6</td>
<td>76.6%</td>
<td>76.6</td>
<td>76.0</td>
<td>75.1</td>
</tr>
<tr>
<td>4</td>
<td>76.1</td>
<td>75.1</td>
<td>75.6</td>
<td>72.1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>74.1</td>
<td>74.6</td>
<td>73.6</td>
<td>74.1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>77.0</td>
<td>76.1</td>
<td>76.6</td>
<td>74.6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>75.6</td>
<td>76.6</td>
<td>76.0</td>
<td>75.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75.1</td>
<td>75.1</td>
<td>74.1</td>
<td>72.3</td>
<td></td>
</tr>
</tbody>
</table>

occurs because there are only 197 test examples.

For each classifier the four slide features produced nearly the same results. This is expected given the strength of the correlation between the features. It is interesting to note that the neural network with four hidden units, which gave the highest cell-by-cell classification rate, scored lower than the other networks for the slide classification. Figure 4.14 shows the ROC curves for slide-by-slide classification using AVGVAL for three classifiers. The ROC curves for the neural networks with three, four, six and seven hidden units are not significantly different than those shown for the two and five hidden networks.

Figure 4.14 displays the performance advantage of neural networks over the linear discriminant vector for classifying MAC slides. The ROC curve for any of the neural networks with two or more hidden units shows significant improvement over that of the discriminant. There is at least a 5–8% increase in the slide-by-slide classification rate when using neural networks. The figure also shows that the networks with more than two hidden units offered no significant improvement over the network with two hidden units for slide-by-slide classification.
Figure 4.14: The slide-by-slide ROC curves for the discriminant function and neural networks with two and five hidden units are shown. The slide test set consists of 125 normal and 72 severe slides.
Pattern classifiers based on feed-forward neural networks are a relatively recent development. Their use has spread since the publication of the back-propagation algorithm in 1985. They have recently been applied by several researchers to the diagnosis of different types of cancer including skin cancer [5], liver cancer [27] and breast cancer [2]. They have also been used to diagnose cervical smear images by McKenna [29], who trained multilayer networks using 80 features to separate normal cells from dysplastic cells with a 93.7% correct classification rate.

This thesis has examined the use of neural networks in the design of a cervical smear classifier based on detection of MACs. The performance advantages of using neural networks in place of discriminant functions have been demonstrated for this classification problem. The success of neural networks in this situation is due to the subtle differences between MACs and negative cells. The author believes that neural networks should be used in other cytological screening decision problems (Figure 3.3) where standard statistical techniques have not performed adequately.

A linear discriminant function calculated for the training set correctly separated 61.6% of the 16904 test examples (Table 4.6). Twenty randomly initialized neural networks with two hidden units were trained to perform this classification task. On average, they correctly classified 71.6% of the test examples (Table 4.8). This classification rate improved as the number of hidden units was increased. Using four hidden units gave the highest average performance (71.9%). Using more than four hidden units overfitted the
training examples leading to a slight decrease in the classification rate for the test cells.

The optimal four hidden unit neural network attained a classification rate of 72.5%. It separated the negative examples into two subpopulations (Figure 4.10). This separation may have been an artifact of the classifier, but it was also present in the histograms of the neural networks using two and three hidden units. Most of the cells that made up one of the subpopulations came from 41 of the slides. These 41 slides contained almost only cells that were given a neural network value of 0.8 by the classifier. Neither staining nor scanning dates could account for the difference between these slides and the rest.

For slide-by-slide classification, the neural networks giving the best cell-by-cell classification rates were compared to the linear discriminant function. Five slide features were defined for the distributions of neural network and discriminant function values. Four of the five features provided nearly the same results for each classifier. The average discriminant function value of the intermediate cells on the slide could correctly classify 69.5% of the 197 test slides. The neural networks using between two and seven hidden units all had classification rates between 74% and 77% for slide-by-slide classification.

The best performance by a neural network for slide-by-slide classification was 77.0% by a network with five hidden units. This was only marginally better than a network using two hidden units which scored 76.6%. When deciding which classifier to use in practice, robustness is an important issue. Very little performance is gained by using the larger network. With this in mind, one would choose the network with two hidden units, which uses 111 free parameters, over the five hidden unit network, which uses 276 parameters.

The data used to train the discriminant function and neural network classifiers consisted of vectors of 53 features calculated for the images of the cell nuclei. No analysis was performed on the usefulness of the features for inclusion in the classifier models.
Seven of the features had strong pairwise correlations (.95 magnitude or larger) to others and could have been removed at the outset. Since the differences between negatives and MACs are subtle, any features which may provide some discrimination value were included in the study.

The problem of feature selection merits further study. Using too many features in the training of the neural networks at best, slows the convergence of the training methods. This can also lead to an overall reduction in the performance of the classifiers. The neural networks can be overfitted on the training examples using features with marginal discriminating ability. The performance of these networks on the training set will not reflect their expected performance in practice.

Discriminant function analysis does not suffer from this problem because features are added to the discriminant model one at a time. Features are only added to the model if they provide significant discriminating ability with respect to the features already included in the model.

Principal component analysis on the training set of 8456 cells showed that the MAC training data could be represented by 28 mutually orthogonal vectors with 0.1% loss of accuracy. The uncertainty of the nuclear features calculated from the segmented images possibly matches or exceeds this value. Thus, the training of the classifier could be shortened by converting each 53 component feature vector into a 28 component vector by using the first 28 principal components. Given that the neural network cell-by-cell classifiers performed between one and five percent less on the test set as compared to the training set when using all 53 features (Table 4.8), the loss of accuracy from using these 28 component vectors would not be significant.

Although principal components could be used to reduce the dimensionality of the data, it does not lead to an improvement of the overall classifier performance. The principal component vectors point in the directions of the greatest variance of the data.
These are not necessarily the directions along which the data classes show the greatest separation.

The variability of the feature values due to staining batch differences appeared to pose a problem in designing a robust MAC classifier. A set of 1000 cells sampled from slides taken from three staining batches were designated class 0. Another set of 1000 cells from slides from three other staining batches were designated class 1. A simple perceptron classifier was trained using these 2000 cells. It was trained to separate cells of slides from three batches from cells of slides from the other three batches. It achieved a classification rate of 82.3% on the training set. It correctly classified 80.6% of a 2000 cell test set of cells drawn from the same slides used for training.

A control experiment was performed using the same 2000 training cells by randomly assigning them either of two classes and training neural networks to separate the classes. In this case, it was impossible to obtain a classifier which performed better than chance at separating the two classes. These two results show that staining batch variations among the features are large enough that they can eclipse the feature differences between the MAC and negative populations. This problem was lessened by choosing training data from slides that spanned the staining batches. However, further work should be conducted to find a way to normalize features so that staining variations do not affect feature values.
Bibliography


