# PATTERN RECOGNITION TECHNIQUES APPLIED TO RUST CLASSIFICATION IN STEEL SURFACES

Yun Zhang

Master of Philosophy

City University

Department of Computing

June 2012

## CONTENTS

# LIST OF FIGURES

*Number* *Page*

## LIST OF TABLES

## ACKNOWLEDGMENTS

I would like to thank my wife for her help, an extraordinary patience, encourage and support during the research that led to completion of this thesis.

I am most grateful to my supervisors Dr. Andrew Tuson, Dr. Peter Smith and Dr. Geoff Dowling for their support, advice and attention to details. This thesis would not exist without their help.

Thanks to for the Automated Inspection and Maintenance of Steel Structures (AIMS) project team who supports me the entire steel surface visual library.

I thank all at the Department of Computing for providing a friendly environment to work in.

Also thanks to all the other people, not mentioned here, who have supported and helped me during the research.

**University Librarian** is allowed to copy the thesis in whole or in part without further reference to the author. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgement.

## ABSTRACT

The life and performance of steel structure depends directly upon the steel surface preparation. The restoration of steel structure such as steel bridges, ships and storage tanks is due mainly to the use of manual surface inspection methods accompanied by surface preparation technologies. It requires a long project duration, high costs and hazardous practices for both worker and environment to complete surface restoration.

The developments of surface preparation technologies make it essential to develop technologies that allows patch restore of corrode steel structure in practice.

This thesis addresses the problem of classification of rust steel surfaces. Various Pattern recognition methods are studied for classifying less subjective steel surfaces from a time corrosion perspective. Our primary contribution is: with appropriate features from the steel surfaces, artificial neural network pattern recognition methods have the abilities to classify the less subjective rust steel surfaces reliably and be suitable for automation. The results provide important information about the classification methods for rust steel surface analysis.

**Keywords:** pattern recognition, rust, classification, steel images.

## ABBREVIATIONS AND TERMS

| | |
|---|---|
| ADALINE | Adaptive Linear Neuron network |
| ALC | Adaptive Lineal Combiner |
| BP | Back-Propagation network |
| ER | Electrical Resistance |
| GLCM | Gray Level Co-occurrence Matrices |
| KNN | K-Nearest Neighbour method |
| LMS | Least Mean Squares algorithm |
| LPR | Linear Polarization techniques |
| LVQ | Learning Vector Quantization |
| NDT | Non Destructive Testing |
| MANOVA | Multivariate analysis of variance |
| MLP | Multilayer Perceptron network |

# Chapter 1 - Introduction and Overview

## 1.1 Industry Background

Steel is the most commonly used metal material in the modern life. The reason to choose steel among all metals is the strength and ease of production of steel. However, steel corrode in many media including most outdoor atmospheres. The serious consequences of the steel corrosion process have become a problem of worldwide significance [Roberge 06]. Protective coatings are probably the most widely used products for steel corrosion control. They are used to provide long-term protection of steel structure under a broad range of corrosive condition, extending from atmospheric exposure to the most demanding condition.

### 1.1.1 Steel Surface Preparation Definition

The life of a coating of steel structure depends as much on the degree of surface preparation as on the subsequent coating system. Surface preparation is defined in IS0 12944-4 as 'any method of preparing a surface for coating'. It is an important part of any steel corrosion protection strategy [Momber 03].

The primary functions of steel surface preparation are to clean the steel surface of material that will induce premature failure of the coating system and provide a surface that can be easily wetted for good coating adhesion.

The ISO 8502 states that the performance of protective coatings of paint and related products applied to steel is significantly affected by the state of the steel surface immediately prior to painting. The principal factors that are known to influence the performance are:

- The presence of rust and mill scale;
- The presence of surface contaminants, including salts, dust, oil and greases;
- The surface profile, which includes primary, roughness and waviness profiles.

The major factors for the selection of a corrosion protection system [Pietsch, Kaiser 02] are shown in the Figure 1.1.

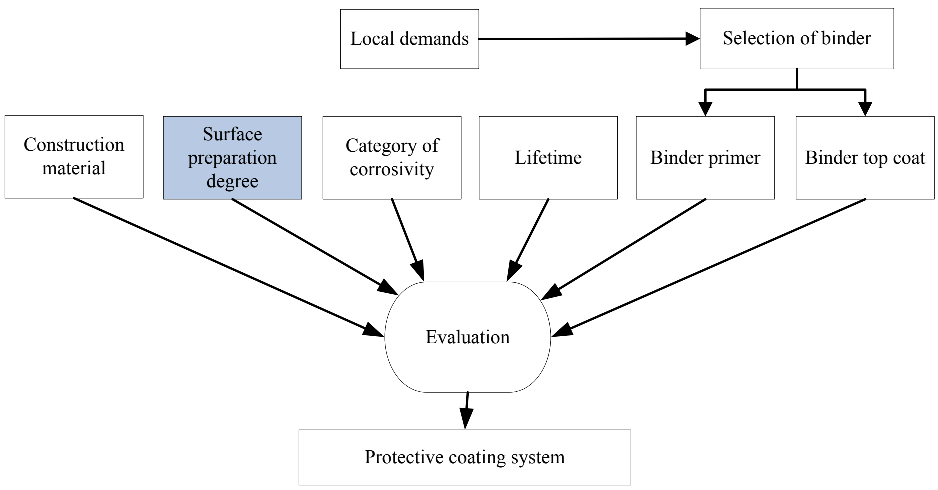


Figure 1.1 Evaluation process for a protective coating system [Pietsch, Kaiser 02]

### 1.1.2 Steel Surface Preparation Standards

Steel surface cleaning methods are specified by several standards, being the most extended the Swedish SIS 0559900, transformed into ISO 8501-1:1988 international standard on rust. It defines rust grades and preparation grades of uncoated steel substrates and steel substrates after overall removal of pervious coatings

The rust grades of uncoated steel surface are classified as A, B, C and D from minimum to maximum by a human expert. The descriptions about them are shown below and the images of rust grades show in the Figure 1.2.

- **Rust grade A**: Steel surface largely covered with adhering mill scale but litter, if any, rust – This would correspond to a hot-rolled steel surface newly made.
- **Rust grade B**: Steel surface which has begun to rust and from which the mill scale has begun to flake – This would correspond to a hot-rolled steel surface exposed to wind and weather without protection into a moderately corrosive atmosphere, for two or three months.
- **Rust grade C**: Steel surface on which the mill scale has rusted away or from which it can be scraped, but with slight pitting visible under normal vision – This would correspond to a steel surface exposed to wind and weather without protection, into a moderately corrosive atmosphere, for about a year.
- **Rust grade D**: Steel surface on which the mill scale has rusted away and on which general pitting is visible under normal vision – This would correspond to a steel surface that was exposed to wind and weather, without protection, into a moderately corrosive atmosphere, for about three years).

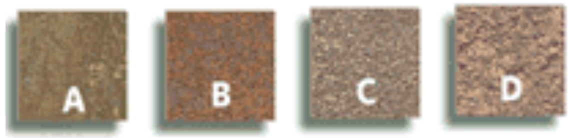


Figure 1.2 International standard of rust grades A, B, C and D

The Figure 1.3 shows a series of 1.5 inch squares with black dots representing various area percentages. These diagrams are not proposed to reproduce the appearance of actual rust patterns but simply to serve as a guide for judging the percentage of surface covered by rust or rust blisters.

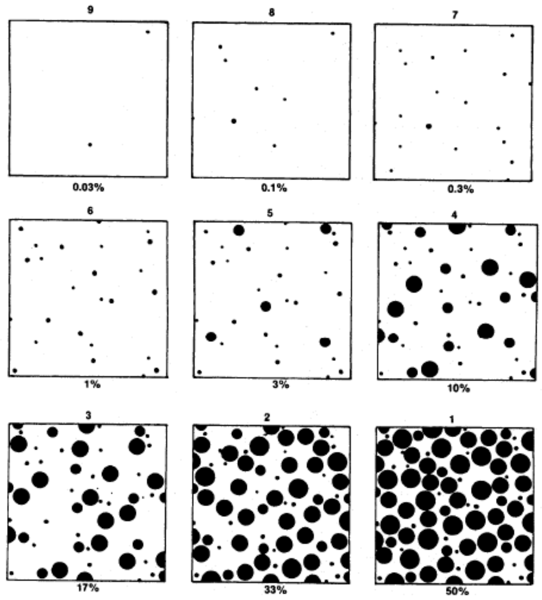


Figure 1.3 Diagrams representing rust grades and the correspond area percentage

### 1.1.3 Procedure for Visual Assessment of Steel Substrates

A substrate is a prepared or treated steel surface. Currently, the human inspection is commonly used to assess the steel substrates. This procedure requires either in good diffuse daylight or in equivalent artificial illumination to examine the steel surface and compare it with each of the photographs of the standard rust grades, using normal vision. Place the appropriate photograph close to, and in the plane of, the steel surface to be assessed. For rust grades, record the assessment as the worst grade that is evident.

### 1.1.4 Steel Surface Preparation Methods

Definition and subdivision of steel preparation methods are listed in ISO 12944-4 (1998). Basically, the following three principal surface preparation methods can be distinguished:

- Water, solvent and chemical cleaning that includes water cleaning, steam cleaning, emulsion cleaning etc.
- Mechanical cleaning method that includes hand-held tool cleaning, power-tool cleaning, blast cleaning and water blast cleaning.
- Flame cleaning

Usually, the first step in the surface preparation process is to mechanically remove rust or debris from the substrate. The methods of mechanical cleaning are given in Figure 1.4.

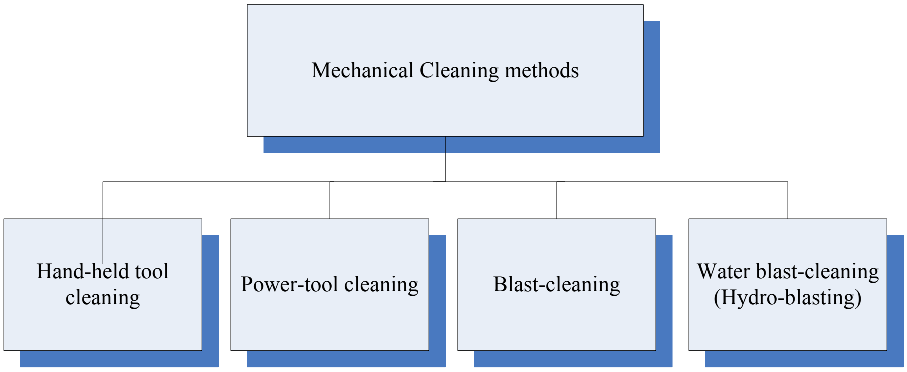


Figure 1.4 Mechanical cleaning method [Momber 03]

The hydro-blasting method is the most effective mechanical cleaning method by far. Hydro-blasting is a technique for cleaning surfaces, which relies entirely on the energy of water striking a surface to achieve its cleaning effect. The tool of any hydro-blasting application is a high-speed water jet. Although the speed of the jet is its fundamental physical property, the pressure generated by the pump unit that produces the jet is the most important evaluation parameter in practice. Water jet applications can be distinguished according to the level of the applied operational pressure as follows [Momber 03]:

a) Pressure cleaning: the use of pressurised water, with or without the addition of other liquids or solid particles, to remove unwanted matter from various surfaces, and where the pump pressure is below 340 bar.

b) High-pressure water cleaning: the use of high-pressure water, with or without the addition of other liquids or solid particles, to remove unwanted matter from various surface, and where the pump pressure is between 340 and 2000 bar.

c) Ultra high-pressure water cleaning: the use of pressurised water, with or without the addition of other liquids or solid particles, to remove unwanted matter from various surfaces, and where the pump pressure exceeds 2000 bar.

The Figure 1.5 shows a typical working environment to apply hydro blasting method.

Figure 1.5 The working environment to apply the hydro-blasting preparation method

## 1.2 Problem Statement

### 1.2.1 Disadvantages of Visual Assessment of Steel Substrates

Firstly, the human inspection is commonly used to assess the steel substrates (see section 1.1.3). The human experts normally assess rust conditions by surveying a steel structure and taking photographs of the areas that need maintenance. The standards used in this evaluation process are often subjective in nature and are based on the experiences of the evaluator. Currently, photographic standards are used for classification of the level of corrosion of coated steel surfaces, as well as, to assess the levels of cleanliness achieved by the established surface preparation methods. The visual standard may difficult to use due to differing appearances of steel, hue and lighting effects. Because of this, the possibility exists that an error may enter the evaluation process that, due to the length of the maintenance cycle, could result in the collapse of the structure before it can be repaired.

Secondly, destructive testing methods have to be applied to support coating condition in some cases.

Final, Only spot inspection is practical.

### 1.2.2 The Hazards of Steel Surface Preparation

As the most common and effective method (see section 1.1.4), the blasting cleaning operation is an activity with significant inherent hazards. If work task are approached inappropriately, significant risks with the potential of serious injury, including fatality are possible. The ISO 12944-4 stats the following for surface preparation in general: 'All relevant health and safety regulations shall be observed.' Hydro blasting has a high injury potential: high-speed water jets can damage skin, tissue. General sources of danger to hydro-blasting operations include the following:

- Reactive forces generated by the exiting water jets.
- Cutting capability of the high-speed jets;
- Hose movements (especially during switch-on of the pump).
- Working in areas of electric devices;
- Uncontrolled escape of pressurised water.
- Damaged parts being under pressure.
- Dust and aerosol formation.
- Sound emitted from equipment and water jet.
- Impact from rebounding debris from the jet impact point.

### 1.2.3 Conclusions

To overcome the problems listed in section 1.2.1 and 1.2.2, it is essential to develop a robotic system to automatically perform the inspection and cleaning operation. The key design criterion for a robotic system is the rust detection and measurement method. Currently, several methods for rust detection and measurement have been research. There methods fall into two categories; they are non-direct (non-intrusive) and direct (intrusive) techniques. Direct techniques such as Corrosion Coupons, Electrical Resistance (ER) and the Linear-polarization techniques (LPR) are small contacting methods and thus not attractive for this case [Trujillo 03]. In contrast, as a large scale and non-contacting method, the visual inspection method seems to be more suitable for the robotic system.

## 1.3 Previous Research

### 1.3.1 Material Surface Classification

There are some relative researches related with material surface classification have been done. Some typical researches are listed below:

- Reniers [[Reniers 08] has present a method to be sued to detect convex ridges on voxel surface extracted from 3D scans.
- Lepistö [Lepistö 03] has introduced a rock texture classification method, which is based upon textural and spectral features of the rock and the textural features are calculated from the co-occurrence matrix. Two types of rock textures are tested, and the experimental results show that the proposed features are able to classify rock textures.
- Sharma [Sharma 00] has worked on Meastex images (A number of image sets which contain examples of artificial and natural textures. Each image has a size of 512×512 pixels and is distributed in raw PGM format) to evaluate the textures methods for image analysis, in which four group images of asphalt, concrete, grass and rock are tested.
- Bruno [Bruno 99] has applied the different statistical and spectral method to characterize and classify ornamental stone samples.

- Don and Fu [Don 84] have inspected metal surface by the roughness of the surface.

### 1.3.2 Steel Surface Classification

The fundamental research of steel surface classification has been done by Ünsalan and Ercil [Ünsalan 95, Ünsalan 98 and Ünsalan 99]. They have considered the problem of studying pattern recognition techniques for analysing textured surface and applied the results to the classification of steel surface. In their research, various texture analysis methods had been applied to extract features from steel surfaces. All the features were optimized by feature selection and extraction algorithms and then fed into a classifier.

They have used a K-Nearest Neighbour classifier (KNN) to classify six different steel surface types (grades A, B, C of both rust grades and sandblasted forms, see section 1.1.2), and a Combining classifier for the binary classification. The system structures are given in Figure 1.6 and Figure 1.7. Figure 1.8 shows the classification results for GLCM.

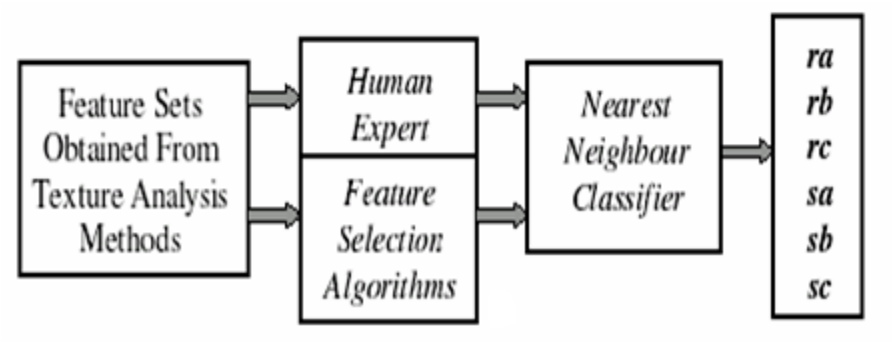


Figure 1.6 Previous Research - System structure to classify six classes [Ünsalan 95]

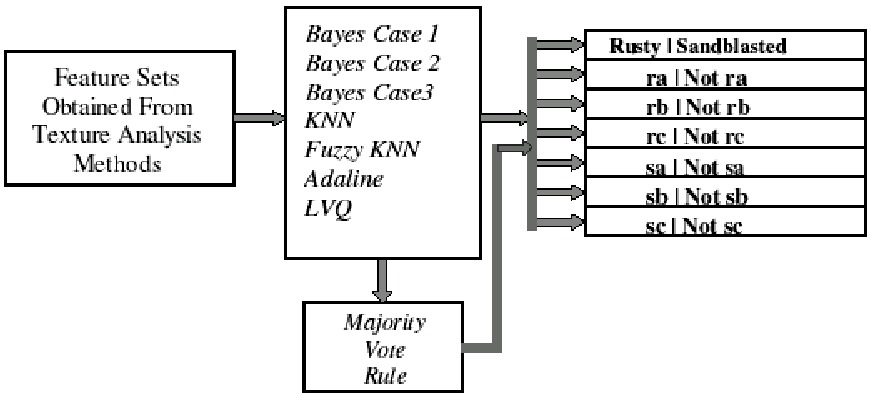


Figure 1.7 Previous Research - System structure of the binary classification [Ünsalan 95]

| | | Classified as | | | | | |
|---|---|---|---|---|---|---|---|
| | | Ra | Sa | Rb | Sb | Rc | Sc |
| Sample from Class | Ra | 87.45 | 0.00 | 8.91 | 0.00 | 3.64 | 0.00 |
| | Sa | 0.09 | 90.55 | 0.00 | 6.55 | 0.91 | 1.91 |
| | Rb | 7.09 | 0.00 | 91.36 | 0.00 | 1.55 | 0.00 |
| | Sb | 0.00 | 3.27 | 0.00 | 79.36 | 0.00 | 17.36 |
| | Rc | 1.91 | 0.00 | 1.09 | 0.00 | 97.00 | 0.00 |
| | Sc | 0.00 | 1.45 | 0.00 | 16.64 | 0.00 | 81.91 |

Figure 1.8 Previous Research - Classification result for GLCM [Ünsalan 95]

The result obtained from the previous research indicates that the features obtained from GLCM (Gray Level Co-occurrence Matrices Method) and MRF (Markov Random Fields Method) can be used by a KNN classifier to discriminate the six types (uncoated and prepared rust grades A, B and C) of steel surface with very high accuracy.

## 1.4 Objectives

The previous research has made some results, but there are still some areas need to be researched further:

- The statistical pattern recognition method (KNN classifier) is the only approach applied in the previous research to classify six types of steel surface. Two kind of artificial neural networks, Kohonen's learning vector quantization (LVQ) and ADALINE have only been applied to make the binary decision. The artificial neural networks have been commonly used in pattern recognition as a natural pattern classifier and cluster. The artificial neural network can approximate every classification function as closely as required in theory. A question has been raised is "can an artificial neural network solve the same problem of non-destructive steel surface classification or even perform better?"

- All previous researches are focused on the uncoated and prepared rust grade A, B and C. Is there any possibility to discriminate the continuous and less subjective rust grades

In this thesis, the problem of classification of steel surfaces is considered. This research therefore will focus on the above two problems to use artificial neural network and statistical pattern recognition techniques to classify continuous and less subjective grades of rust rather than simply A to C grades.

## 1.5 Outline of the Thesis

Chapter 2 provides an overview of texture analysis methods and pattern recognition methods which have been applied for this research.

Chapter 3 explains how to obtain features from the steel surface. A visual library which contains 500 pictures representing rust between 0 to 8000 hours has been adopted. Six different steel surface types are selected form this visual library and the Co-occurrence Matrices method is applied as the feature extraction algorithms. The classification approaches are also provided in this chapter. Various artificial neural network and statistic pattern recognition methods are examined.

The results are stated and discussed in the Chapter 4.

## Chapter 2 - Literature Review

This chapter explains some key concepts for pattern recognition techniques as well as and texture analysis methods

This research will implement some of the pattern recognition techniques and texture analysis methods in order to solve the rust image classification problem.

Texture analysis is one possible method to detect features in images. As a well-know statistical method it measures second-order texture characteristics of an image, the Co-occurrence matrix method is introduced as the main method for feature extraction.

There are three major approaches to design a pattern recognition system. However, we only focus on two approaches – statistical pattern recognition and artificial neural network pattern recognition.

This chapter also explains various pattern recognition methods practised within this research along with their advantages and disadvantages, which are:

1) The Discriminant analysis and K-Nearest Neighbour analysis are introduced as statistical pattern recognition methods.

2) ADALINE neural network, LVQ (learning and vector quantization) network and Multilayer feed-forward neural network are introduced as artificial pattern recognition methods.

## 2.1 Introduction of Pattern Recognition

The definition of pattern is - “A pattern is essentially an arrangement or an ordering in which some organization of underlying structure can be said to exist” [Pandya 96]. Pattern recognition is the scientific discipline that has a goal to classify objects into a number of categories or classes. These objects can be images or signal waveforms or any type of measurements that need to be classified by different applications.

Pattern recognition has a long history, but it was mostly the output of theoretical research in the area of statistics before the 1960s. However, the demands for practical applications of pattern recognition are increased by the advent of computers, and pattern recognition becomes an important field of computer science and electrical engineering that studies the operation and design of systems to recognize patterns in data. Important tasks that have been tackled are image analysis, character recognition, speech analysis, man and machine diagnostics, person identification and industrial inspection etc. Pattern recognition also is an integral part in most machine intelligence systems which built for decision making. Pattern recognition is very important in the machine vision area. A machine vision system captures images and analyses them to obtain descriptions of what is imaged [Theodoridis 06].

There are two paradigms for pattern recognition, classification and cluster analysis. Classification pattern recognition is also called supervised pattern recognition – there is a predefined set of classes of patterns and the tasks are to classify a future pattern as one of these classes. The classifier is designed by exploiting this a priori known information.  In cluster analysis (also known as unsupervised pattern recognition), the aim is to seek groups of patterns, where a pattern needs to be assigned to a so far unknown class of patterns.

Pattern recognition aims to map input feature space to the output class space, and is concerned with making decision from complex patterns of information [Ripley 96] such that:

a) Each input vector is mapped to one and only one class. This is to say each sample vector in the input space will be assigned to one class only.

b) Many input vectors map to one class. The classification algorithm categorizes the input space.

c) The class space is constructed by the information that extracted from the training sample set. The main differences between many classification algorithms are the constructions of the class space. Each classifier has its own characteristic rules to construct the class space.

d) All the vectors from the input space that have not been used in training are also mapped to one class in the output class space. This is the generalization principle of the classification algorithm. The performances of the classifiers are measured by their correct classification rates of test sample set [Ünsalan 98].

To give an example for the classification: Figure 2.1 shows two images, each having a distinct region inside it. To distinct these two regions, the measurable quantities need to be identified. Figure 2.2 shows a plot of the mean value of the intensity in each region of interest versus the corresponding standard deviation around this mean.

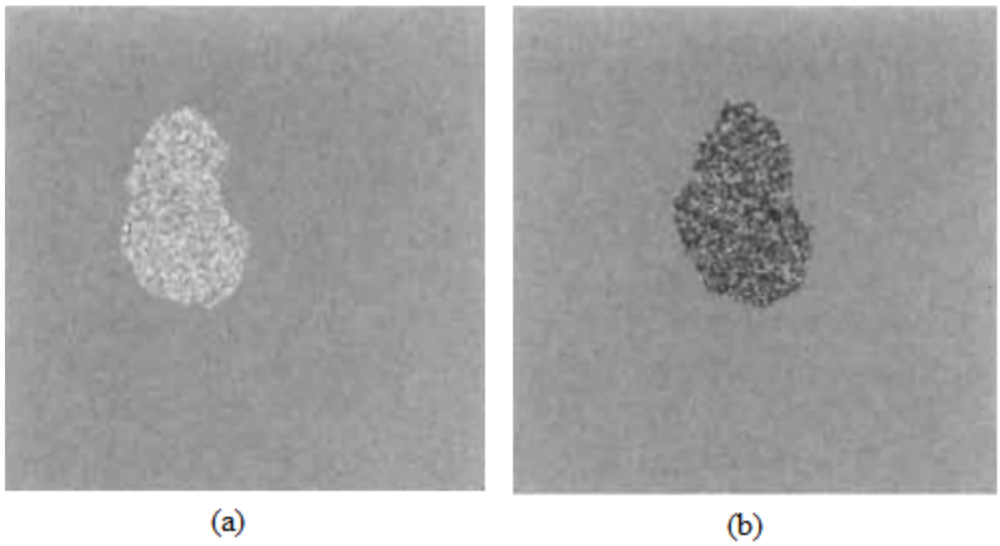

Figure 2.1 Example of image regions corresponding to (a) class A and (b) class B.

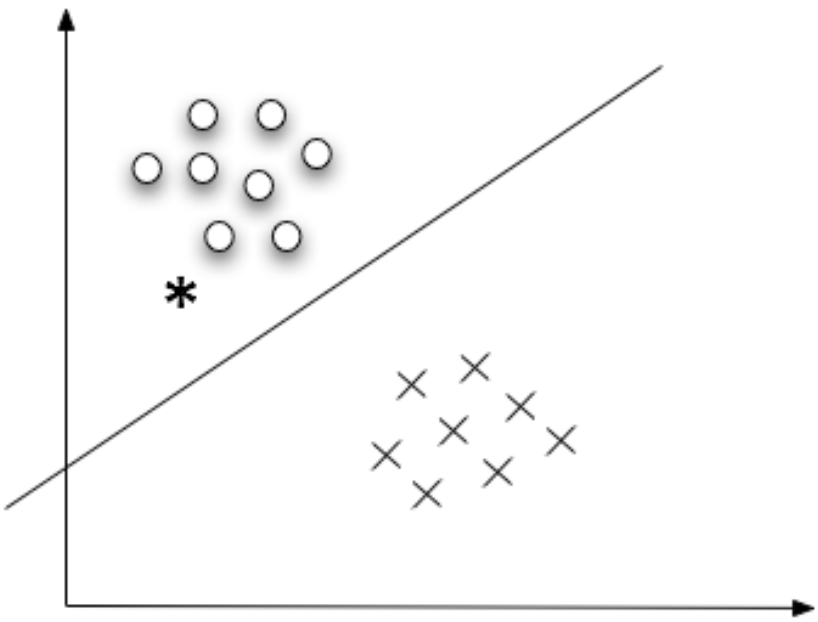

Figure 2.2 Plot of the mean value versus the standard deviation for a number of

different images originating from class A (o) and class B (+)

Each point corresponds to a different image from the available database. It shows that class A patterns tend to spread in a different area from class B patterns. The straight line seems to separate the two classes. If a new image which is not identified to neither class is provided, we can measure the mean intensity and standard deviation in the region of interest of the new image and plot the corresponding point. This is shown by the asterisk (*) in Figure 2.2. Then it is more reasonable to assume that the unknown pattern is more likely to belong to class A than class B. The mean value and the standard deviation in this case, are known as features. In the more general case $l$ features $x_i$ $i = 1,2,...,l$ , are used and they form the feature vector

$$x = [x_1, x_2, ..., x_l]^T$$

Where T denotes transposition and each of the feature vectors identifies uniquely a single pattern (object).

The straight line in Figure 2.2 is called the decision line, and it establishes the classifier whose role is to divide the feature space into regions that correspond to either class A or class B.

Figure 2.3 shows the different stages followed for the design of a classification system. These stages are interrelated, and depending on the results, one may go back to redesign earlier stages to improve the overall performance.

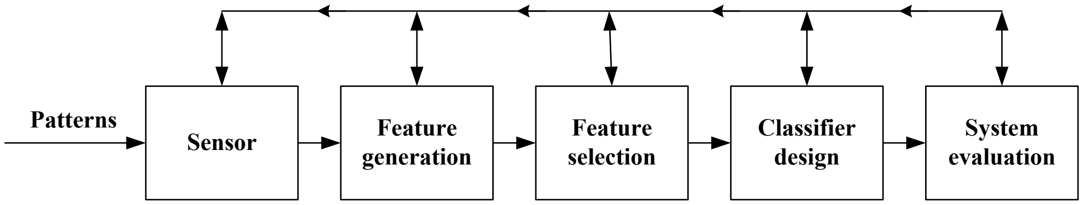


Figure 2.3 The basic stages involved in the design of a classification system

The major approaches to design a pattern recognition system are:

- Statistical pattern recognition.
- Syntactic or structural pattern recognition.
- Artificial neural network (ANN) pattern recognition.

In statistical pattern recognition, the problem of pattern classification is described as a statistical decision problem and it learns all information from examples. Statistical pattern recognition is a mature theories and a number of commercial recognition systems have been designed.

In structural pattern recognition, the properties of information about the class are used to structure the problem. In syntactic pattern recognition, the information is provided by the grammar of a formal language. The patterns are represented in a hierarchical fashion, with pattern built of sub-patterns. These sub-patterns may also be built of other sub-patterns or they can be primitives [Pandya 96].

In artificial neural network pattern recognition, neural networks are implemented as a class of mathematical algorithms and provide solutions to a number of specific problems. Artificial neural network techniques are strongly related to corresponding statistical method. More details about artificial neural network will be introduced later.

Figure 2.4 shows the steps involved in the design of a typical pattern recognition system.

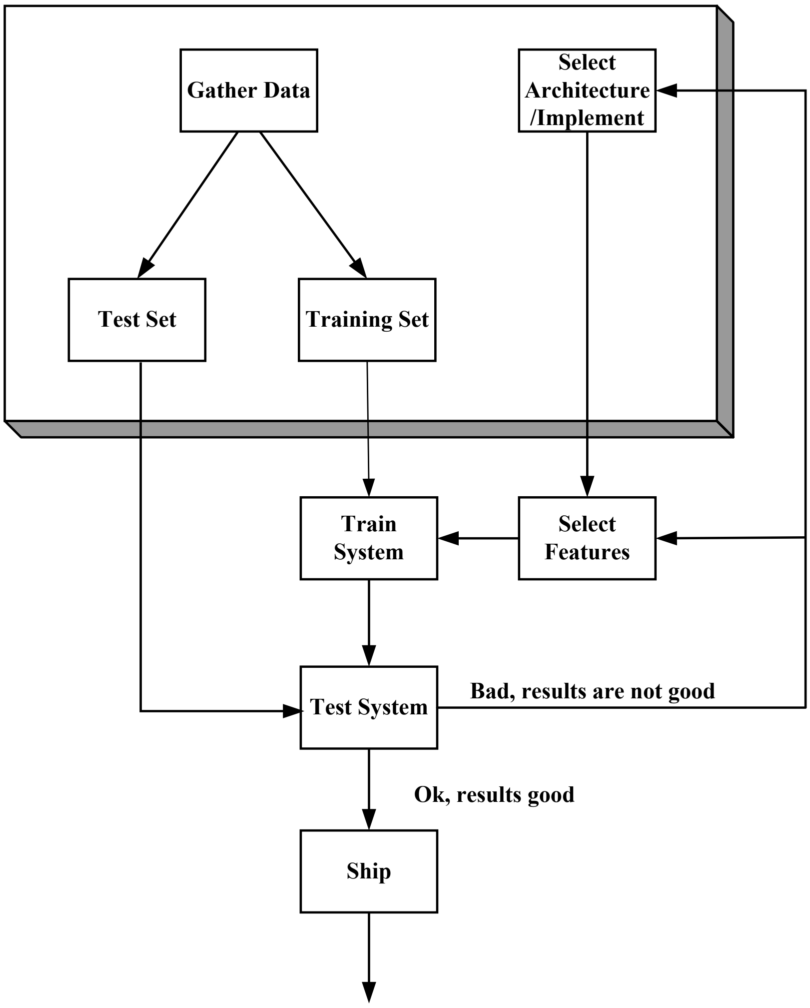


Figure 2.4 The flow chart of the process of designing a learning machine for pattern recognition

## 2.2 Statistical pattern recognition methods

### 2.2.1 Discriminant Analysis

There are many different ways to represent pattern classifiers. One of the most useful is in term of a set of discriminant functions. Theory behind the discriminant function can be found in [Huberty 06] and [Field 00], but a brief overview is given below.

A discriminant is a multivariate test to detect relationships between several dependent variables. It is applied to determine which features discriminate between a numbers of groups. A discriminant function is a linear combination of dependent variables and can be described as a set of multiple regression equations. The standard expression of multiple regression is shown below.

$$\gamma = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n + \varepsilon_i$$

Where, $\gamma$ is the outcome variable, $\beta_m (m \in [1,n])$ is the coefficient of $m^{th}$ predictor. $\varepsilon_i$ is the difference between the predicted and the observed value of $\gamma$ for the $i^{th}$ subject.

The $\beta$ values in the discriminant function are weights which describe the contribution of each dependent variable to the variate and are obtained from the eigenvectors of the product of the model sum of squares and cross-product (SSCP) matrix (H) and the inverse of residual SSCP matrix ($E^{-1}$). Matrix H contains the model sums of squares for each dependent variable and the model cross-product between the two dependent variables. Similarly, matrix $E^{-1}$ contains the residual sums of squares for each dependent variable and the residual cross-product between the two dependent variables [Field 00].

Before giving an introduction to the product of the model SSCP matrix (H) and the residual SSCP matrix (E), some prior knowledge needs to be understood.

a) Model Sum of Squares ($SS_M$): shows how much of the total variation can be explained by the fact that different data points come from different groups, it is obtained by $SS_M = \sum n_i(\bar{x}_i - \bar{x}_{grand})^2$.

Where $\bar{x}_i$ is the mean of each group. $\bar{x}_{grand}$ is the grand mean and $n_i$ is the number of scores of each group.

b) Residual Sum of Squares ($SS_R$): shows how much of the variation cannot be explained, $SS_R = \sum (x_i - \bar{x}_i)^2$.

Where $x_i$ is each score in a group and $\bar{x}_i$ is the mean of the group from which $x_i$ came.

c) Cross-product (CP): The difference between the scores and the mean in one group multiplied by the difference between the scores and the mean in the other groups. Two cross-products should be calculated, they are:

The model cross-product ($CP_M$) – show how the relationship between the dependent variable is influenced by experimental manipulation:

$$CP_M = \sum n\left[(\bar{x}_{group(group_1)} - \bar{X}_{grand(group_1)})\ldots(\bar{x}_{group(group_m)} - \bar{X}_{grand(group_m)})\right]$$

where $\bar{x}_{group(group_m)}$ is the mean of each group, $\bar{X}_{grand(group_m)}$ is the grand mean.

The residual cross-product ($CP_R$) – show how the relationship between the dependent variables is affected by individual differences.

$$CP_R = \sum (x_{i(group_1)} - \overline{X}_{group(group_1)})...(x_{i(group_m)} - \overline{X}_{group(group_m)})$$

where $x_{i(group_m)}$ is each score in group m, $\overline{X}_{group(group_m)}$ is the group mean.

Now, the definitions of the model SSCP matrix (H) and the residual SSCP matrix (E) can be defined as:

$$H = \begin{bmatrix} SS_{M(group_1)} & CP_M & ... & CP_M \\ CP_M & SS_{M(group_2)} & ... & CP_M \\ ... & ... & ... & ... \\ CP_M & CP_M & ... & SS_{M(group_m)} \end{bmatrix}$$

$H$ represents both the unsystematic variation that exists for each dependent variable and the co-dependence between the dependent variables that is duo to chance factor alone.

$$E = \begin{bmatrix} SS_{R(group_1)} & CP_R & ... & CP_R \\ CP_R & SS_{R(group_2)} & ... & CP_R \\ ... & ... & ... & ... \\ CP_R & CP_R & ... & SS_{R(group_m)} \end{bmatrix}$$

$E$ represents both the systematic variation that exists for each dependent variable and the co-dependence between the dependent variables that is due to the model.

The product of $HE^{-1}$ represents the ratio of systematic variance to the unsystematic variance in the model.

It will have one regression equation for each group to calculate the discriminate score, and then "the final step is to assess how large these values are compared to what we would expect by chance alone" [Field 00]. There are four ways for us to access the values.

- Pillai-Bartlett Trace (V), the sum of the proportion of explained variance on the discriminant function: $V = \sum_{i=1}^{s} \frac{\lambda_i}{1+\lambda_i}$.
- Hotelling's $T^2$, the sum of the eigenvalues for each variate: $T = \sum_{i=1}^{s} \lambda_i$
- Wilks's Lambda($\Lambda$), the product of the unexplained variance on each of the variates: $\Lambda = \prod_{i=1}^{s} \frac{1}{1+\lambda_i}$
- Roy's Largest Root, the eigenvalue for the first variate: largest root = $\lambda_{largest}$

where, $\lambda_i$ $(i = 1,2 \ldots s)$ is the eigenvalue for each of the discriminant variates, and $s$ is the number of variates.

Discriminant function analysis is MANOVA reversed. In MANOVA, the independent variables are groups (features obtained from the rust image in this research, see section 2.4.1.2 and section 3.4) and the dependent variables are the predictors (rust classes in this research). Discriminant analysis usually used to predict membership in naturally occurring groups. It answers the question: can a combination of variables be used to predict group membership? Usually, several variables are included in a study to see which ones contribute to the discrimination between groups.

Discriminant function analysis is broken into a two-step process:

1) Testing significance of a set of discriminant function. This step is computationally identical to MANOVA. There is a matrix of total variances and covariances; likewise, there is a matrix of pooled within-group variances in covariances. The two matrices are compared via multivariate tests in order to determine whether or not there are any significant differences (with regard to all variable) between groups. One first performs the multivariate test, and, if statistically significant, proceeds to see which of the variable have significantly different means across the groups.

2) Classification. Once group means are found to be statistically significant, classification of variables is undertaken. Discriminant analysis automatically determines some optimal combination of variables so that the first function provides the most overall discrimination between groups; the second provides second most, and so on. Moreover, the function will be independent or

orthogonal, that is, their contributions to the discrimination between groups will not overlap. The first function picks up the most variation; the second function picks up the greatest part of the unexplained variation, etc. Computationally, a canonical correlation analysis is performed that will determine the successive functions and canonical roots. Classification is then possible from the canonical function. Subjects are classified in the groups in which they had the highest classification scores. The maximum number of discriminant functions will be equal to the degrees for freedom, or the number of variables in the analysis, whichever is smaller.

### 2.2.2 K-Nearest Neighbour (KNN) classifiers

K nearest neighbour, KNN is a non-parametric discriminant technique very useful for classification purpose [Ďurčeková 09], which does not need any assumption to the destitution of errors. The intuition underlying Nearest Neighbour Classification is very straightforward, examples are classified based on the class of their nearest neighbours. The main variant of KNN is based the majority vote rule, which means that K neighbour objects, nearest to the classification object, are search and then the classification of the given object is made according to which class the neighbour object are predominantly classified. It is often useful to take more than one neighbour into account where k nearest neighbours is used in determining the class [Cunningham 07].

K nearest neighbour algorithm is very simple; it works based on minimum distance from the query instance to the training samples to determine the K nearest neighbours.

In Parzen Windows Approach bin size s taken constant, sample size is assumed variable to improve approximation of likelihood probability distribution function (PDF). The approximation for the likelihood PDF can be improved by taking constant sample size and variable bin size also. It is called K nearest neighbour estimator. Variable bin size can be given as $dk(X)$.

The estimated PDF becomes:

$$f(X) = \frac{1}{n \times dk(x)} \sum_{i=1}^{n} \left( \frac{x - X_i}{dk(x)} \right)$$

The kernel function is taken as:

$$K(u) = \begin{cases} 1/2 & if |u| < 1 \\ 0 & otherwise \end{cases}$$

In KNN classifier, the volume of the bin is taken variable, while the number of samples in the bin is taken as constant. The discriminant function KNN classifier becomes form the above estimated probability distribution function:

$$g_i(x) = \frac{k_i}{k}$$

where k is the neighbourhood size taken and $k_i$ is the number of samples belonging to class i in the k neighbourhood.

There is a special case of KNN such that the neighbourhood size is taken as one. This classifier is called Nearest Neighbour classifier. It simply assigns the test sample to the class having a training sample closest to it.

As explained above, the KNN classification has two stages; the first stage is the determination of nearest neighbours and the second is the determination of the class using those neighbours.

KNN should be considered in seeking a solution to any classification problem as it very easy to understand and implement. Some advantages of KNN are as follows [Cunningham 07]:

- KNN is easy to implement and debug.
- KNN can be very effective if an analysis of the neighbor is useful as explanation.
- There are some noise reduction techniques that work only for k-NN that can be effective in improving the accuracy of the classifier.

These advantages of KNN, particularly those that derive from its interpretability, should not be underestimated. However, some significant disadvantages are as follows:

- Because all the work is done at run-time, k-NN can have poor run-time performance if the training set is large.
- KNN is very sensitive to irrelevant or redundant features because all features contribute to the similarity and thus to the classification. This can be ameliorated by careful feature selection or feature weighting.
- On very difficult classification tasks, KNN may be outperformed by more exotic techniques such as Support Vector Machines or Neural Networks.

## 2.3 Artificial Neural Networks

### 2.3.1 ADALINE Neural Networks

Widorw and Hoff introduced the ADALINE (Adaptive Linear Neuron) network and a learning rule which is called Least Mean Squares (LMS) algorithm in 1960. The ADALINE network is very similar to the perceptron, except that its transfer function is linear, but the LMS algorithm is more powerful than the perceptron learning rule. The perceptron rule is guaranteed to converge to a solution that correctly categorizes the training patterns, but the network can be very sensitive to noise, since the patterns are often placed close to the decision boundaries. The LMS algorithm minimizes mean square error, and tries to move the decision boundaries as far from the training patterns as possible.

The structure of the ADALINE network includes an element which is denominated Adaptive Lineal Combiner (ALC) that obtains a linear response which can be applied to other element of bipolar commutation. So if the output of the ALC is positive, the response of the ADALINE network is + 1, if ALC is negative, then the result of the ADALINE network is – 1. The linear output that the ALC generates is given by:

$$y(t+1) = \begin{cases} +1 & s > 0 \\ y(t) & s = 0 \\ -1 & s < 0 \end{cases}$$

The binary answer corresponding to the ADALINE network is:

$$s = \sum_{j=0}^{N} w_j x_j = \mathbf{w}^{\mathrm{T}}\mathbf{X}$$

ADALINE can be used to classify objects into two categories. However, it can do so only if the objects are linearly separable [Hagan 95].

### 2.3.2 Multilayer Feed-forward Neural Networks

The history of research into artificial neural network has already spans more than half a century. Artificial neural networks are based on the neural structure of the human brain. The first artificial neural network (Perceptron) was designed by F. Rosenblatt at 1957 [Yuan 99], which was the first time that the theoretical research was transferred into a practical experiment. Unfortunately, the basic perceptron network is only able to solve a limited class of problems (which was proved by Minsky and Papert [MiPa 69]), the researches therefore, were led to concentrate on the mathematical or computer science aspect of pattern-formatted information processing [Pandya 96], for example, statistical pattern recognition and classification of patterns with syntactic structure. However, during the 1980s, the problems of lack of new ideas and powerful computers with which to experiment were overcome. Since then the research of artificial neural network has been greatly improved and brought us that much closer to the goal of creating human-like behaviour systems [Pandya 96].

A neural network is a parallel, distributed information processing structure consisting of processing elements. Each processing element has many collateral connections to other processing elements – inputs and outputs. The actual output depends on the current value of the input signals and transfer function of this element [Hecht-Nielsen 90]. A neural network resembles the brain in two respects [Haykin 94]:

- Knowledge is obtained by the network through a learning process.
- Interneuron connection strengths (synaptic weight) are used to store the knowledge.

The procedure to perform the learning process is called a learning algorithm [Haykin 94]. Actually, the traditional methods for the design of neural networks are the modification of synaptic weights.

The artificial neural networks have some important characteristics [Masters 93]:

(1) **Distribution storing and fault-tolerance:** Information is not stored in any one place. Instead, it is stored in the whole network. Every part of a network does not store a single piece of information exactly, but stores combinations of many items of information. The network is equipotential for information storing. For neural networks we use an associative method to retrieve the information which we originally stored in net. The advantage is that if some of the information is missing (e.g. error or lost), the system still can work properly.

(2) **Parallel processing:** The structure of ANN is parallel, and every unit does similar processing at the same time. By adopting this approach, artificial neural networks have a very high speed, and they are much quicker than a normal serial computer.

(3) **Adaptiveness:** A neural network is able to self-adapt as per the requirement in a continually changing environment by powerful learning algorithms and self-organizing rules.

(4) **Nonlinear Processing:** The ability to deal with nonlinear relationships and noise-immunity, make artificial neural networks a good candidate for difficult classification and prediction problems. Figure 2.5 shows a general model of a multiple-input neuron.

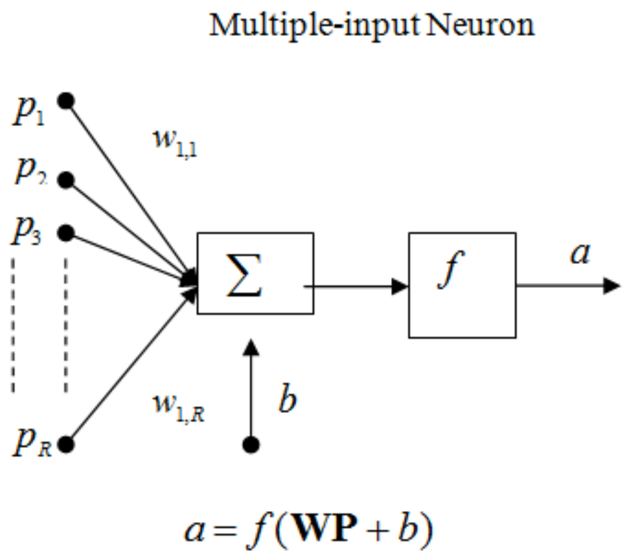


Figure 2.5 Multiple-input Neuron

Where, $p_1, p_2, ..., p_R$ are individual inputs; $w_{1,1}, w_{1,2}, .., w_{1,R}$ are corresponding weights of $p_1, p_2, ..., p_R$; b is a bias and $f()$ is the transfer function. The neuron output can be written as $a = f(\mathbf{WP} + b)$.

Although there are many different neural networks for classification, clustering, and modelling, the most popular and versatile form of neural network classifiers is by far the multilayer perceptron (MLP) network trained by back propagation. “It has been shown that multilayer perceptron networks with a single hidden layer and a nonlinear activation function are universal classifiers. That is, such networks can approximate decision boundaries of arbitrary complexity.” [Pandya 96].

A multilayer feed-forward network is structured by a set of neurons, which are arranged into two or more layers. The basic structure of a multilayer feed-forward network is made of three layers – input layer, output layer and one hidden layer.

#### 2.3.2.1 Back-propagation algorithm

Back-propagation is a specific technique for implementing gradient descent in weight space for a multilayer feed-forward network [Haykin 94]. The learning in back-propagation has been separated to two distinct phases – the forward phase and the backward phase. In the forward phase, all input signals propagate through the whole network, layer by layer, to calculate the actual outputs. In the backward phase, the network compares the actual outputs with expected outputs to generate error signals, which are then propagated in the backward direction through whole network, and the weights of the network are updated by minimising the sum of squared error. Those two phases are repeated until the error has been accepted as being tolerable small.

The back-propagation algorithm has appeared as the most popular algorithm for the supervised training of multilayer perceptrons, and has two distinct properties:

- It is simple to compute locally.
- It performs stochastic gradient descent in weight space.

With all the algorithms that spring from the gradient descent methods, the convergence speed of the back-propagation scheme depends on the value of the learning constant. The value of the learning constant must be sufficiently small to guarantee convergence but not too small, because the convergence speed becomes very slow.

The cost function minimization for a multilayer perceptron is a nonlinear minimization task. Thus, the existence of local minimal in the corresponding cost function surface is an expected reality. Hence, the back-propagation algorithm runs the risk of being trapped in a local minimum. If the local minimum is deep enough, this may still be a good solution. However, in cases in which this is not true, getting stuck in such minimum is an undesirable situation and the algorithm should b reinitialised from a different set of initial conditions.

To overcome some of the drawbacks of back-propagation (gradient descent), There are many ways for improving the algorithm. The major variations of back-propagation are heuristic modifications, which arise out of a study of the distinctive performance of the standard back-propagation algorithm. Some heuristic modifications of back-propagations are discussed with next two sections (see section 2.3.2.2 and 2.3.2.3).

### 2.3.2.2 Back-Propagation with momentum

This is a modification that intends to smooth out the oscillations in the trajectory to improve a convergence. A low-pass filter is applied to do this. When the momentum filter is added to the parameter changes, the following equations for the momentum modification to back-propagation are obtained [Hagan 95].

$$\Delta W^{m}(k) = \gamma \Delta W^{m}(k-1) - (1-\gamma)\alpha S^{m}(a^{m-1})^{T}$$

$$\Delta b^{m}(k) = \gamma \Delta b^{m}(k-1) - (1-\gamma)\alpha S^{m}$$

Where, $m = o,1,...M-1$, $M$ is the number of layers in the network; $\Delta W$ is weight change matrix; $a$ is output; $\Delta b$ is bias matrix; $\alpha$ is learning rate; $\gamma$ momentum coefficient and $S^{m}$ is sensitivity.

Figure 2.6 shows the different with a training sample with (left) or without (right) momentum by using same learning rate. The momentum tends to make the trajectory continue in the same direction and to accelerate convergence when the trajectory is moving in a consistent direction.

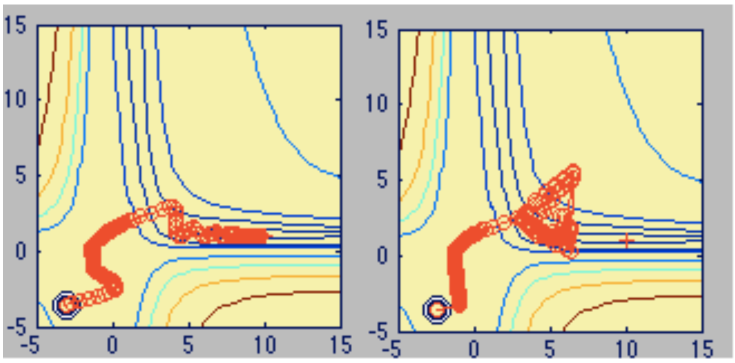


Figure 2.6 Trajectory with and without Momentum

#### 2.3.2.3 Variable learning rate

Convergence will be speed up if the learning rate is increased on flat surfaces and then decreased when the slope increases. There are several heuristics need to be followed [Haykin 94]:

- Every adjustable network parameter of the cost function should have its own individual learning rate parameter.
- Every learning rate parameter should be allowed to vary from one iteration to the next.

- When the derivative of the cost function with respect to a synaptic weight has the same algebraic sign for several consecutive iterations of the algorithm, the learning rate parameter for that particular weight should be increased.
- When the algebraic sign of the derivative of the cost function with respect to a particular synaptic weight alternates for several consecutive iterations of the algorithm, the learning rate parameter for that weight should be decreased.

There are many different approaches for varying the learning rate. A very straightforward batching procedure is introduced, where the learning rate is varied based on the performance of the algorithm. The rules of the variable learning rate back-propagation algorithm are:

a) If the squared error (over the entire training set) increases by more than some set percentage $\xi$ (typically one to five percent) a weight update is discarded after the weight update, the learning rate is multiplied by some factor $0 < \rho < 1$, and the momentum coefficient $\gamma$ is set to zero if it is used.

b) If the squared error decreases after a weight update, then the weight update is accepted and the learning rate is multiplied by some factor $\eta > 1$. If $\gamma$ has been previously set to zero, it is reset to its original value.

c) If the squared error increases by less than $\xi$, then the weight update is accepted and the learning rate is remain same. If $\gamma$ has been previously set to zero, it is reset to its original value.

#### 2.3.2.4 Designing feed-forward network architectures

The difficult problem to design a feed-forward network is to choose the size of the network. For multiple-layer feed-forward networks, the number of hidden can mean the difference between success and failure. While there are no hard and fast rules for defining the network parameters, the following three guidelines should be followed [Masters 93]:

- Use one hidden layer.
- Use very few hidden neurons
- Train until you can't stand it anymore.

It has been proved that there is no theoretical reason ever to use more than two hidden layers. It has also been see that for the vast majority of practical problem, there is no reason to use more than one hidden layer.

Beside the associated computational complexity problems, there is a major reason why the size of the network should be kept as small as possible. This is imposed by the generalization capabilities that the network must possess. The term generalization refers to the capability of the multilayer neural network to classify correctly feature vector that were not presented to it during the training phase.

### 2.3.3 Learning and Vector Quantization (LVQ) networks

The learning vector quantization network is a hybrid network and it can be used in both of unsupervised and supervised learning extension of the Kohonen network methods to form classifications. In the LVQ network neurons in the first layer are assigned to a class, each class is then assigned to one neuron in the second layer. The number of neurons in the first layer will always be at least as large as the number of neurons in the second layer. The LVQ algorithm may be expressed in the following steps [Pandya 96].

As with the competitive network, each neuron in the first layer of the LVQ networks learns a prototype vector, which allows it to classify a region of the input space. However, instead of calculating the proximity of the input and weight vector by using the inner product, the LVQ network will calculate the distance directly. On advantage of calculating the distance directly is that the vector need not be normalized.

Step1. Initialization:

Initialize the weight vectors. The weight vectors may be initialized randomly and initialized the learning rate as well.

Step2. For each vector $\mathbf{x}^{(p)}$ in the training set follow steps 2a and 2b.

Step 2a. Find the winning neuron $k$ such that:

$$i(\mathbf{x}^{(p)}) = k \text{, where } \left\|W_k - \mathbf{x}^{(p)}\right\| < \left\|W_j - \mathbf{x}^{(p)}\right\| \qquad j = 1,2,\ldots,n$$

Step 2b. Update the weight $\mathbf{w}_k$ as follows:

$$W_k^{new} = \begin{cases} W_k^{old} + \eta(\mathbf{x}^{(p)} - W_k^{old}) & if \qquad T = C_j \\ W_k^{old} - \eta(\mathbf{x}^{(p)} - W_k^{old}) & if \qquad T \neq C_j \end{cases}$$

Where $C_j$ is the correct class of feature j, T is the decision made at time t and $\eta$ is the learning rate.

Step 3. Adjust the learning rate:

The learning rate is reduced as a function of iteration.

Step 4. Check for termination:

Exit if termination conditions are met. Otherwise go to step 2.

The LVQ network described above works well for many problems, but it does suffer from a couple of limitations:

First, as with competitive layers, occasionally a hidden neuron in an LVQ network can have initial weight values that stop it from ever winning the completion. The results is a dead neural which never does anything useful.

Secondly, depending on how the initial weight vectors are arranged, a neuron's weight vector may have to travel through a region of a class that it does not represent.

## 2.4 Texture Analysis

Texture analysis and classification are common tasks in pattern recognition. The main aim of texture analysis is texture recognition and texture-based shape analysis. The aim of texture description is to derive some measurements that can be used to classify a particular texture [Nixon 02]. Texture refers to properties which represent the surface or structure of an object [Sonka 98], it could be defined as a structure composed of a large number of more or less ordered similar elements or patterns without one of these drawing special attention [VanGool 85]. "The notion of texture appears to depend upon three ingredients: (i) some local 'order' is repeated over a region which is large in comparison to the order's size, (ii) the order consists in the nonrandom arrangement of elementary parts, and (iii) the parts are roughly uniform entities having approximately the same dimensions everywhere within the textured region." [Hawkins 69]. Texture comprises texture primitives (also called texels) and is highly dependent on the number considered (the texture scale). "A Texture primitive is a contiguous set of pixels with some tonal and/or regional properties, and can be depicted by its average intensity, maximum or minimum intensity size, shape, etc." [Sonka 98]. Image texture is represented by the number and types of primitives and their spatial relationship. In a simple way, texture is the feature which can help to segment images into regions of interest and to classify those regions. It also can be quantified and used to identify the object classes they represent.

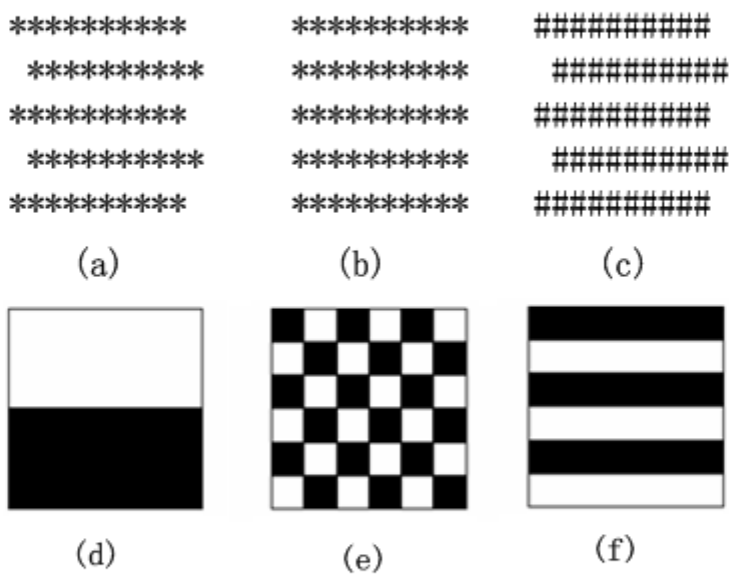


Figure 2.7 Artificial Texture

Figure 2.7 (a) and (b) show that same number and same type of primitives give different textures. Similarly, (a), (b) and (c) show the same spatial relationship of primitives does not guarantee texture uniqueness. (d), (e) and (f) have different texture patterns, although each of them contains 50% black and 50% white pixels.

Texture can be classified according to their strength, which is highly affected by the choices of texture description method. Weak textures have small spatial interactions between primitives and can be described by frequencies of primitive types appearing in some neighbourhood. Strong textures show the spatial interactions between primitives are somewhat regular and the frequency of occurrence of primitive pairs in some spatial relationship may be adequate. Therefore, many statistical texture properties are evaluated to describe the weak textures, and the strong texture recognition is normally accompanied by an exact definition of texture primitives and their spatial relationships [Sonka 98].

Statistical and syntactic considerations are two main approaches to texture description. The statistical methods compute different properties when texture primitive sizes are comparable with the size of pixels. Each texture of this approach is described by a feature vector of properties of a point in a multi-dimensional feature space. The aim is to find a decision rule classifying a texture to some specific class. The syntactic approach is suitable when primitives can be described using a larger variety of properties. Syntactic texture description is found on an analogy between the texture primitive spatial relations and the structure of a formal language [Sonka 98].

Various texture extraction methods of statistical texture description are described in briefs below [Sonka 98]:

Texture analysis has been used in a range for recognising synthetic and natural textures. Previous researches have evaluated the texture methods for image analysis. Each study has used a different combination of textures methods which can be categorised as: statistical, geometrical, structural, model based and signal processing features [Sharma 00].

#### 2.4.1.1 Autocorrelation texture description

The textural character is related to the spatial size of the texture primitives. Small primitives give fine texture (e.g. dog fur), the other way round, and large primitives present coarse texture (e.g. river pebbles). Higher spatial frequencies characterize fine textures and low spatial frequencies portray coarse texture. In this model, texture spatial organization evaluates the linear spatial relationships between primitives. The autocorrelation function value decreases slowly with increasing distance if the primitives are large, whereas it decreases rapidly if texture consists of small primitives. Nevertheless, the autocorrelation increases and decreases periodically with distance if the primitives are periodic. The autocorrelation coefficients can be described below.

$$C_{ff}(p,q) = \frac{MN}{(M-p)(N-q)} \frac{\sum_{i=1}^{M-p} \sum_{j=1}^{N-q} f(i,j) f(i+p, j+q)}{\sum_{i=1}^{M} \sum_{j=1}^{N} f^2(i,j)}$$

where $p, q$ is the position difference in the $i, j$ direction, and $M, N$ are the image dimensions.

#### 2.4.1.2 Co-occurrence matrices

As a statistical approach, the Co-occurrence Matrices method is a powerful and popular feature extraction algorithm [Mridula 11]. Texture features based on co-occurrence matrix are an efficient means to study the texture of an image. This was the result of the first approach to describe, and then classify, image texture. The Co-occurrence method describes second-order image statistics and works well for a large variety of textures. It describes the spatial relations between tonal pixels, and invariance to monotonic gray-level transformations. However, it does not consider primitive shapes and is not suitable for the texture consists of large primitives. It can be obtained by calculating the number of times each pair of coloured pixels is found at distance $d$ in direction $\Theta$ in a digital image. Frequencies of co-occurrence as functions of angle and distance can be represented formally as [Sonka 98]:

$$\begin{aligned} P_{0^o,d}(a,b) =& |\{[(k,l),(m,n)] \in D : \\ & (k-m=0, l-n=-d), f(k,l)=a, f(m,n)=b\}| \end{aligned}$$

$$\begin{aligned} P_{45^o,d}(a,b) =& |\{[(k,l),(m,n)] \in D : \\ & (k-m=-d, l-n=d), f(k,l)=a, f(m,n)=b\}| \end{aligned}$$

$$\begin{aligned} P_{90^o,d}(a,b) =& |\{[(k,l),(m,n)] \in D : \\ & (k-m=d, l-n=0), f(k,l)=a, f(m,n)=b\}| \end{aligned}$$

$$\begin{aligned} P_{135^o,d}(a,b) =& |\{[(k,l),(m,n)] \in D : \\ & (k-m=d, l-n=d), f(k,l)=a, f(m,n)=b\}| \end{aligned}$$

where $|\{\ldots\}|$ refers to set cardinality and $D = (M \times M) \times (M \times N)$.

If $d = 1, \Theta = 0$, we consider pixels are unit apart horizontally to the east. A simple example source image and its Co-occurrence Matrix $P_{0^o,1}$ are given in Figure 2.8

| | | | |
|---|---|---|---|
| 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 3 | 2 | 2 |
| 3 | 3 | 2 | 1 |

Source Image

| | **0** | **1** | **2** | **3** |
|---|---|---|---|---|
| **0** | 2 | 2 | 0 | 1 |
| **1** | 0 | 2 | 0 | 0 |
| **2** | 0 | 1 | 1 | 0 |
| **3** | 0 | 0 | 0 | 1 |

Co occurrence Matrix for source image.

Figure 2.8 - Source Image and Co-occurrence Matrix of the source image

Four features are derived from the Co-occurrence Matrix $f(r,c)$. They are ***Energy***, ***Contrast***, ***Entropy*** and ***Homogeneity***. The definitions are shown below [Hall-Beyer 00].

| Feature | Formula | Description |
|---|---|---|
| **Energy:** | $\sum_{r,c} f(r,c)^2$ | A measure of homogeneity, the opposite of entropy. Also known as Uniformity or Angular Second Moment. |

| | | |
|---|---|---|
| **Contrast:** | $\sum_{r,c}(r-c)^2 * f(r,c)$ | Measure variability of value differences and hence coarseness of texture.<br>A large value of contrast indicates large local variation. |
| **Entropy:** | $\sum_{r,c} f(r,c) * \log(f(r,c))$ | Indicates degree of disorder or non-homogeneity.<br>This is largest when all entries of $f(r,c)$ are equal. It is low when the elements are close to either 0 or 1. |
| **Homogeneity:** | $\sum_{r,c}\frac{f(r,c)}{1+\lvert r-c\rvert}$ | This is also known as Inverse Difference Moment.<br>It is high when the co-occurrence matrix concentrates along the diagonal. |

**Edge frequency**

Edges can be detected by using edge operator masks to yield an edge image from an original image. The distance-dependant texture description function $g(d)$ can be computed for any sub-image $f$ defined in a neighbourhood $N$ for variable distance

$d$ :

$$g(d) = |f(i,j) - f(i+d,j)| + |f(i,j) - f(i-d,j)| \\ + |f(i,j) - f(i,j+d)| + |f(i,j] - f(i,j-d)|$$

The function $g(d)$ is inversely related to the autocorrelation function, its minimum corresponds to the maximum of the autocorrelation function whereas its maximum corresponds to the autocorrelation minimum.

**Primitive length (run length)**

Coarse textures are presented by a large number of neighbouring pixels of the same gray-level, while a small number represents fine texture. A primitive is a maximum contiguous set of pixels in the same direction and have the same gray-level and can be defined by its gray-level, length and direction. If $B(a,r)$ is assumed be the number of primitives of all directions having length $r$ and gray-level $a$, let $M$, $N$ be image dimensions, $L$ the number of gray-levels, $N_r$ the maximum primitive length in the images and the total number of runs - $K$ given by $\sum_{a=1}^{L}\sum_{r=1}^{N_r} B(a,r)$, then five features can be defined as below:

- Short primitives emphasis: $\frac{1}{K}\sum_{a=1}^{L}\sum_{r=1}^{N_r}\frac{B(a,r)}{r^2}$

- Long primitives emphasis: $\frac{1}{K}\sum_{a=1}^{L}\sum_{r=1}^{N_r} B(a,r)r^2$

- Gray-level uniformity: $\frac{1}{K}\sum_{a=1}^{L}\left[\sum_{r=1}^{N_r}\frac{B(a,r)}{r^2}\right]^2$

- Primitive length uniformity: $\frac{1}{K}\sum_{r=1}^{N_r}\left[\sum_{a=1}^{L} B(a,r)\right]^2$

- Primitive percentage: $\frac{K}{\sum_{a=1}^{L}\sum_{r=1}^{N_r} r \quad B(a,r)} = \frac{K}{MN}$

**Laws' texture energy measures**

The texture properties of laws' texture energy measures are determined by assessing average gray-level, edges, spots, ripple and waves in texture [Laws 79]. Certain gradient operators such as Laplacian and Sobel operators are observed to accentuate the underlying microstructure of texture within an image. The measures are derived from three simple vectors: $L_3 = (1,2,1)$ which represent averaging; $E_3 = (-1,0,1)$ calculating first difference (edges); and $S_3 = (-1,2,-1)$, corresponding to the second difference (spot). After convolution of these vectors with themselves and each other, five vector result:

| | | | |
|---|---|---|---|
| Level | L5 | = | [1 4 6 4 1 ] |
| Edge | E5 | = | [-1 -2 0 2 1 ] |
| Spot | S5 | = | [-1 0 2 0 -1 ] |

Wave W5 = [-1 2 0 -2 1 ]

Ripple R5 = [ 1 -4 6 -4 1 ]

The arrays are convolved with other arrays in a combination manner to generate a total of 25 masks. For example, the L5S5 mask can be derived

$$L_5^T \times S_5 = \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \\ -4 & 0 & 8 & 0 & -4 \\ -6 & 0 & 12 & 0 & -6 \\ -4 & 0 & 8 & 0 & -4 \\ -1 & 0 & 2 & 0 & -1 \end{bmatrix}$$

Though the gray level co-occurrence matrices computes the frequency of spatial pixel pair in a texture image instead of single pixel, the retrieval performance of grey level co-occurrence matrix is not quite satisfactory because of the simplicity of gray pixels in gray level co-occurrence matrices [Li 10].

Apart from the above statistical methods, there are still many other texture analysis methods can be implemented. For example, (a) the model based methods – Markov random fields; (b) image processing methods – Histogram of an image (HIS) and Radon transform; (c) spectral methods – 2-D Fourier transform and Wavelet transform; (d) Human visual processing – Gabor transform and Surface density approach etc.(see [Ünsalan 95] for more details).

Co-occurrence matrix is one of the commonly used texture analysis methods and has been successfully applied to many applications. Weszka et al [Weszka 76] found the co-occurrence were the best after compared the Fourier spectrum, second order gray-level statistics, co-occurrence statistics and gray-level primitive length. Dubes and Ohanian [Dubes 92] have compared Markov Random Filed parameters, multi-channel filtering features, fractal based features and co-occurrence matrices features, and found the co-occurrence method performed the best. Sharma and Singh [Sharma 00] have used five different statistic texture features extraction methods for image understanding studies by using a set of K nearest neighbour classifiers and the results show that the co-occurrence matrix method and the Law's method have the best result. Zacharias [Zacharias 10] has combined singular points and gray level co-occurrence matrix to solve the fingerprint classification problem.

Most importantly, the previous research done by Ünsalan [Ünsalan 95] has used eight different texture analysis methods to extract features from steel surfaces and the results show that features which obtained from co-occurrence matrices method and Markov random fields method can be classified as six types of steel surface with very high accuracy (see section 1.3.2).

According to the above results, the co-occurrence matrices method will be applied as the main feature extraction method. At same time to consider the particular colour distributions of the steel surface, the co-occurrence matrices will also be applied to red (R), green (G) and blue (B) colour scheme along with gray level.

# Chapter 3 - Visual Library and Classification Approaches

## 3.1 Introduction

The previous experiments and some publications show that it is possible to extract some features and to classify them using pictures representing real metallic plates rusted and filmed in regular time for some reasonable for industry period of time.

The purpose of the research is to obtain a system that it will be able to achieve with a high probability and a little time of processing, the time of rush of some metallic surface images.

A collection of images of different time of corrosion and with some different treatments of steel materials will be used to achieve this purpose.

The co-occurrence matrices method will be used to extract features from these images, from which we can classify rust grade based on the time scale and produce some conclusions for depth of the rust and future treatment of the surface.

This chapter will be separate to several tasks: visual library producing, visual library optimization, visual library automation data support system, models for date representing, system and the approaches for rust classification.

## 3.2 Visual Library

### 3.2.1 Visual Library images requirements

The visual library was produced from 500 pictures representing rust between 0 and 8000 hours time (333 days). The initial conditions for pictures were applied to be able to reduce the errors during the calculations. Some of the conditions involved difficult requirements such as right and constant light source and the same camera allocation.

Conditions for Video library filming:

- The intensity of the illumination has to be identical for all the surfaces of the pictures;
- The distance between the samples and the CCD camera has to be fixed for all the sampling period;
- For the development of the work, the same CCD camera should be used for the image gathering; The geometrical place of the CCD camera, the light and the samples should be equal for all the pictures;
- The samples has to be (if possible) from a unique source material (metal);
- The conditions of the creation of the rust has to be equal for all the samples;
- The samples have to be taken in equal intervals of time.

### 3.2.2 Visual Library Optimization

The video library is not only a randomly stored collection of the pictures. The data inside should be organised in a proper way according to the software requirement from the support of the system.

### 3.2.3 Size Optimization Process

The data (images) has to be stored in a minimum space in the memory and using the optimum size.

This was done automatically decreasing the size of the files and excluding some background of the pictures by a support system. The size of the image in the Figure 3.1 was reduced from 680 KB to 30 KB, which correspondingly reduces the time for image processing.

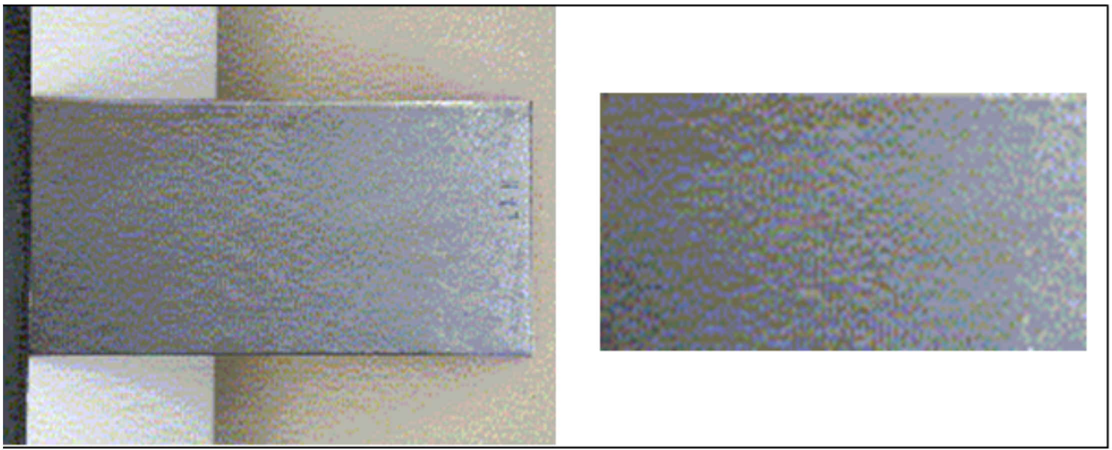

Figure 3.1 Image size optimization

### 3.2.4 Labelling (Naming) and Storing the Pictures in Database

The process of labelling and storing the pictures is very important for a proper and optimal management of the software support which grabs the pictures, calculate the feature and store the results data in stack or proper input.

To be realized the mentioned support system the pictures should be stored and named in structures (subdirectories).

Figure 3.2 shows a brief view of the structure of the file presented in Windows Explorer.

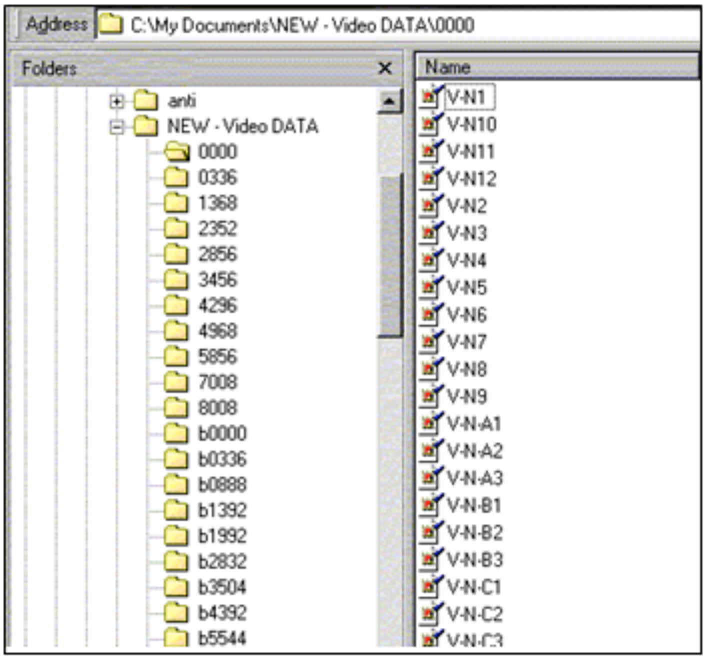


Figure 3.2 The structures in Video Library

The typical structure represents subdirectories named as:

- 0000 to 8000 - non-blasted with mill-scale intact (series from 0h to 8000h);
- b0000 to b7000 – different source to group I and blast cleaned prior to exposure (series from 0h to 7344h)
- pb0000 to pb5664 – same source as group I, but blasted cleaned prior to exposure (p group) (series from 0h to 5664h)

### 3.3 Visual Library Automation data Support System

It is necessary to develop a software system to support the calculation process and to extract the features for each image in the library. The system will be able to automatic loading and store the date from the visual library and will reduce the calculation time significantly.

The extreme advantage of this automatic support system can also be demonstrated in a software level. Having such system, the mathematical models can be replaced without disturbing the functions and communication process in the supporting system. At the same time, the supporting system can be very complicated according to the functionality of the storage of the data, sorting the results, grabbing the information from the visual library and giving some information for the user.

The software system can reduce the developing time helping to create only ones the supporting facilities for features extraction. The Figure 3.3 and Figure 3.4 show some user interfaces of the software system.

The software system will provide the following functionalities:

- It will have the ability to support multiple image file formats (i.e. JPEG, bitmap and binary image).
- It will have the ability to optimizing (preparing) rust images. It will allow the initial position and size to be specified. It will also allow multiple images to be processed at the same time.
- It will have the flexibility to allow the system to accommodate a mathematical model for feature extraction and operate with it.
- The feature data (result data) will be store in a text file. The data within the text file can be imported to other application such as Excel, SPSS and MATLAB.

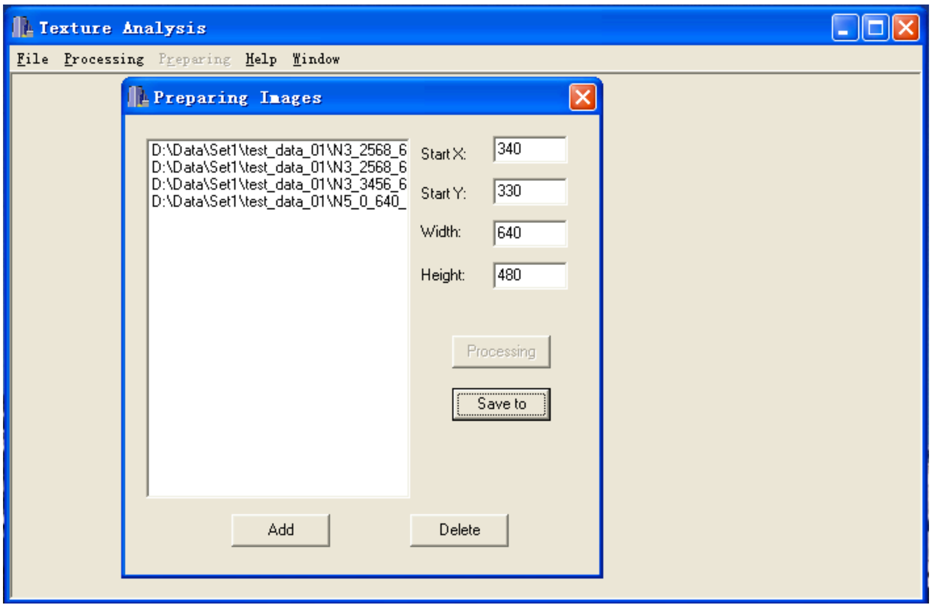


Figure 3.3 Visual library optimization user interface

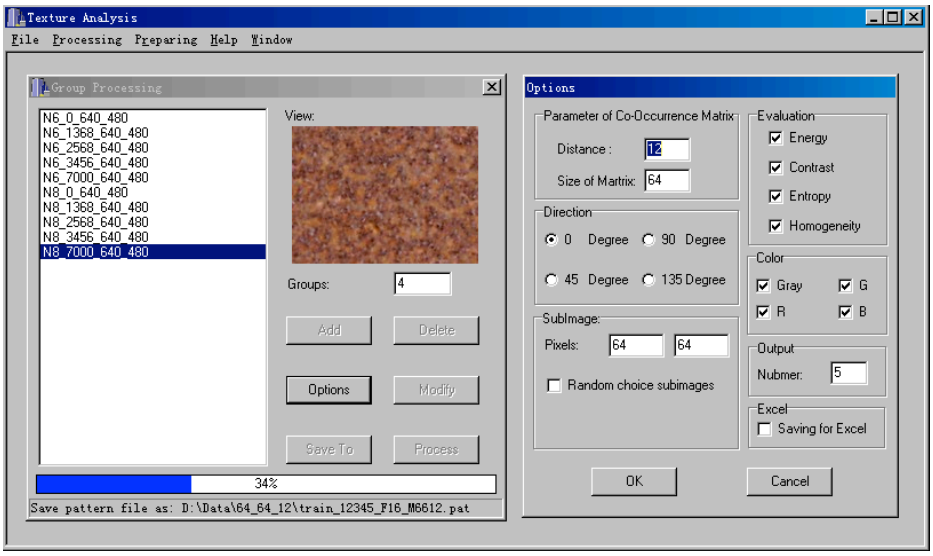


Figure 3.4 Feature extraction user interface

## 3.4 Models for data representing

As it is explained in the chapter two, the co-occurrence method presents second-order image statistics, and works well for a large variety of textures. The typical here is that we connect the relative position of the pixel with it intensity of the pixels and this represent important texture feature. However, the memory requirements are big disadvantage. As value of each colour channel lies in the range 0 to 255, it means the full size of the co-occurrence matrix will be $256 \times 256$ bits, thus it will take a considerable time to calculate the those four features. Therefore, a fast way is to reduce the size of co-occurrence matrix to $32 \times 32$ (or $64 \times 64$) by dividing the colour value by 8 (or 4). In general, co-occurrence matrices are calculated for four directions,

The whole image library contains eleven time-based groups – spanning a time period from 0 to 8808 hours of exposure. To simplify the classification, a set of image data is chosen to avoid the effects of the different lighting and distance condition. 6 groups are chosen in a certain period time (0, 1368, 2856, 4296, 5856 and 7008 - 1368 hours different for each group) and four images are random selected for each group to ensure each group contains different steel surface to generalize the rust images.

Four rust images for each class are divided into two groups. One group is the training data (containing two images) and the other group is testing data (again containing two images). One set of rust images are given in Figure 3.5.


Figure 3.5 One set of rust images which contains six classes

The rules for calculating training and test/validation data are given in the following:

- The size of each image is reduced to $640 \times 480$ (since the original images of steel plate contain a background, the image is copped to hold just rusting steel).
- Each sub-image has a dimension of $32 \times 32$ or $64 \times 64$, giving 300 or 70 sub-images from each image.
- Two different sizes of Co-occurrence Matrix are applied, they are $32 \times 32$ and $64 \times 64$. Three different distances are applied to calculate the Co-occurrence Matrices at direction 0, they are 1, 6, and 12.
- Four colour channels (R, G, B, and grey level) are used for feature extraction, therefore 16 features are used in total.

Twelve different data sets are obtained from all possible combinations, there are:

| Size of Matrix | Size of sub-image | Distance (Sample size of each class) | | |
|---|---|---|---|---|
| 32 | 32 | 1 (600) | 6 (600) | 12 (600) |
| 32 | 64 | 1 (140) | 6 (140) | 12 (140) |
| 64 | 32 | 1 (600) | 6 (600) | 12 (600) |
| 64 | 64 | 1 (140) | 6 (140) | 12 (140) |

Figure 3.6 Combinations of different feature sets

The graphic representations of the co-occurrence matrix extracted for an image set are shown in the Figure 3.7, Figure 3.8 and Figure 3.9.

Co-occurrence matrix represents a special texture analysis method. The typical here is that we connect the relative position of the pixel with it intensity of the pixels and this represent important texture feature (see section 2.4.1.2).

The graphs for the co-occurrence matrix are $64 \times 64$ pixels and obtained from the $64 \times 64$ pixels sub images.

The axes in the graphs have the following meanings:

1) Axis (from 1 to 16) presents the co-occurrence matrix features

- 1-4 represent Energy, Contrast, Entropy and Homogeneity of red colour channel.
- 5-8 represent Energy, Contrast, Entropy and Homogeneity of green colour channel.
- 9-12 represent Energy, Contrast, Entropy and Homogeneity of blue colour channel.
- 13-16 represent Energy, Contrast, Entropy and Homogeneity of grey colour channel.

2) Axis (from 1 to 1260) presents the rust classes
   - Class one (1 – 210)
   - Class two (211 – 420)
   - Class three (421 – 630)
   - Class four (631 – 840)
   - Class five (841 – 1050)
   - Class six (1051 – 1260)

3) Z axis – presents the values of the features.

The graphs only provide very simple ideas about how the features extracted from the rust image represented. In the real world, the multi-demission feature space will be far more complicated.

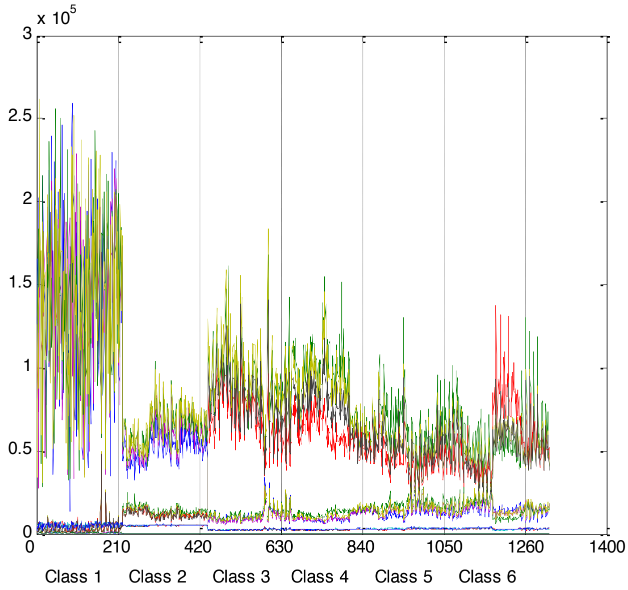


Figure 3.7 Co-Occurrence matrices of the six classes represented in all colour channels - Class View

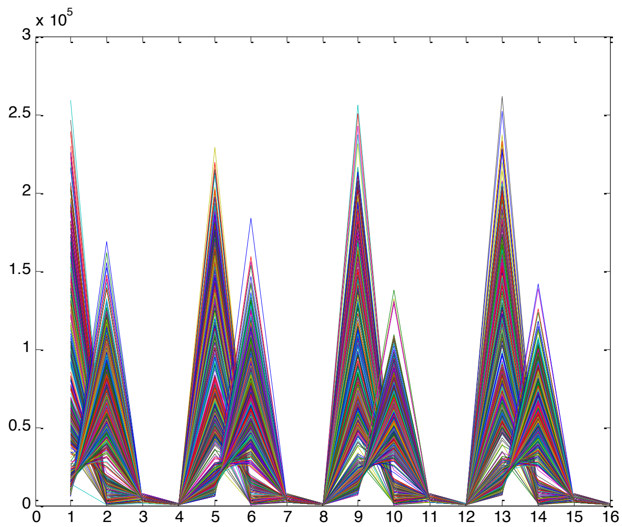


Figure 3.8 Co-Occurrence matrices of the six classes represented in all colour channels - Colour Channel View

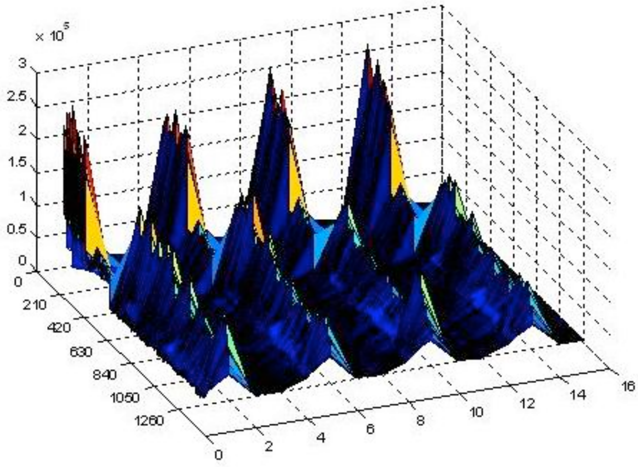


Figure 3.9 Co-Occurrence matrices of the six classes represented in all colour channels - 3D view

## 3.5 Classification Approaches

### 3.5.1 Software Packages

MATLAB is the neural network simulator which is applied in this research. Although a numeric computation and visualization software package are currently available, the MATLAB is widely available and, because of it matrix/vector notation and graphics, is a convenient environment in which to experiment with neural networks. Neural Network Toolbox of MATLAB provides tools for designing, implementing, visualizing, and simulating neural networks.

SPSS Statistics offers the full scope of statistical and analytical capabilities that this research requires. The research is benefited with its built in discriminant analysis and nearest neighbours functionalities.

However, MATLAB and SPSS are not essential for using within this research. The artificial neural network and statistic classification approaches can perform with any available programming language or other software which offer the same functionalities.

### 3.5.2 General Classification Approaches

Both artificial neural network and statistic classifications are followed the steps listed below:

a) The classification is started from the simplest case – binary classification. Training and testing data of the six classes are put into two groups (Is A and Is Not A) for all possible combinations.

b) The next step is to classify three different classes with large time scale. The class one, three and six are chosen for this approach.

c) If the results of three classes' classification are acceptable, the four classes' classification will take place. This time, the class one, two, three and six will be used.

d) The class one, two, three, five and six will be present for five classes classification

e) Finally, the classification of all six classes will be examined.

f) All feature sets (see section 3.4) will be examined by repeating the step a to b.

### 3.5.3 Artificial Neural Network Classification Approaches

The training data will be divided into three sets using random indices when training the neural networks:

- The 70% data (samples) will be presented to the neural network as training data, the network is adjusted according to its error.
- The 15% data (samples) will be presented to the neural network as validation data. The validation data are used to measure network generalization, and to halt training when generalization stops improving.
- The 15% data (samples) will be presented to the neural network as testing data. The testing date has no effect on training and so provides an independent measure of network performance during training.

After the training, the testing data will be presented to the neural network to measure the overall network performance after training. The Figure 3.10 shows the training progress of a neural network which is used to classify six classes.

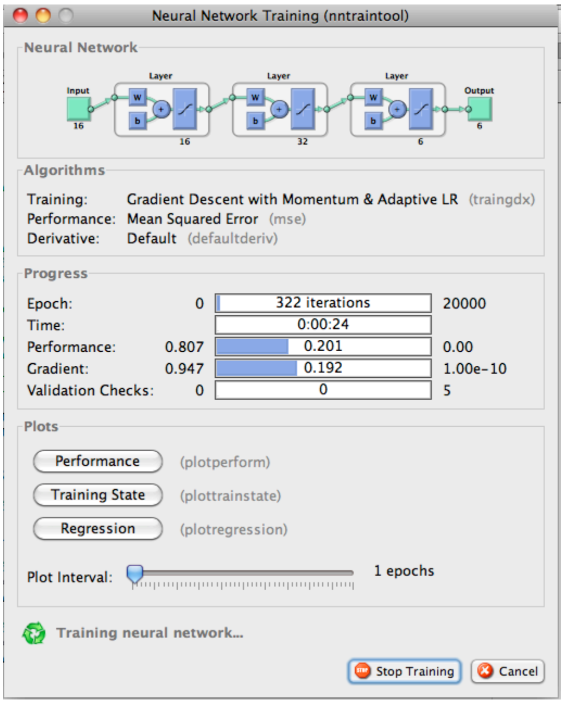


Figure 3.10 The training progress of a neural network which is used to classify six classes

### 3.5.4 The Statistic Classification Approaches

The training and testing data will be combined together as one data set for the discriminant analysis approach.

The training and testing data sets will be combined together to present to the nearest neighbours classification. However, the randomly assign cases to partitions of training is 70% and holdout is 30% during the analysis.

The classification progresses of discriminant and nearest neighbour analysis are shown in Figure 3.11 and Figure 3.12.

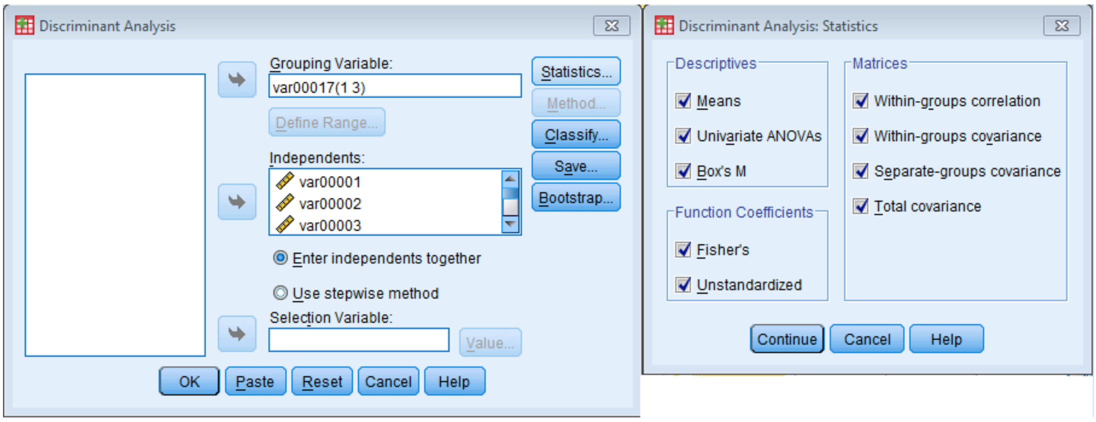


Figure 3.11 The discriminant analysis

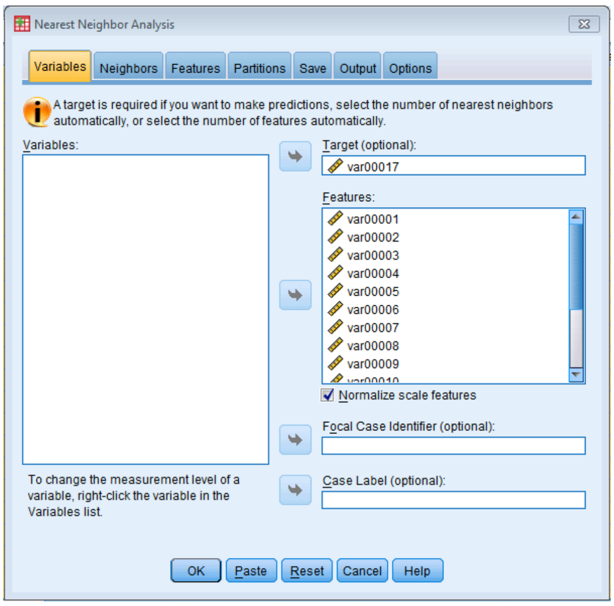


Figure 3.12 Nearest neighbour analysis

### 3.5.5 The Overall Structure of the Classification System

The structure of this research is shown in Figure 3.13. The descriptions of this system are stated below:

a) All data are obtained from rust images by feature extraction method.

b) All data have been divided into two disjoint sets - the training set and the validation set.

    a. The training and testing sets are combined together and is analysed by a discriminate function and k-nearest neighbour method to acquire statistical information.

c) The SPSS Statistics package is applied for statistic classification experiments

d) The neural networks are simulated by the MATLAB and trained with the training set.

e) The testing set is used to measure the network performance after training.

f) If the result is acceptable (can be classified), the training is successful, end of the system.

g) If the result is not acceptable, the training process is performed again by using different training parameters or different training algorithms. It is also possible to re-calculate data by using different feature extraction/selection methods.

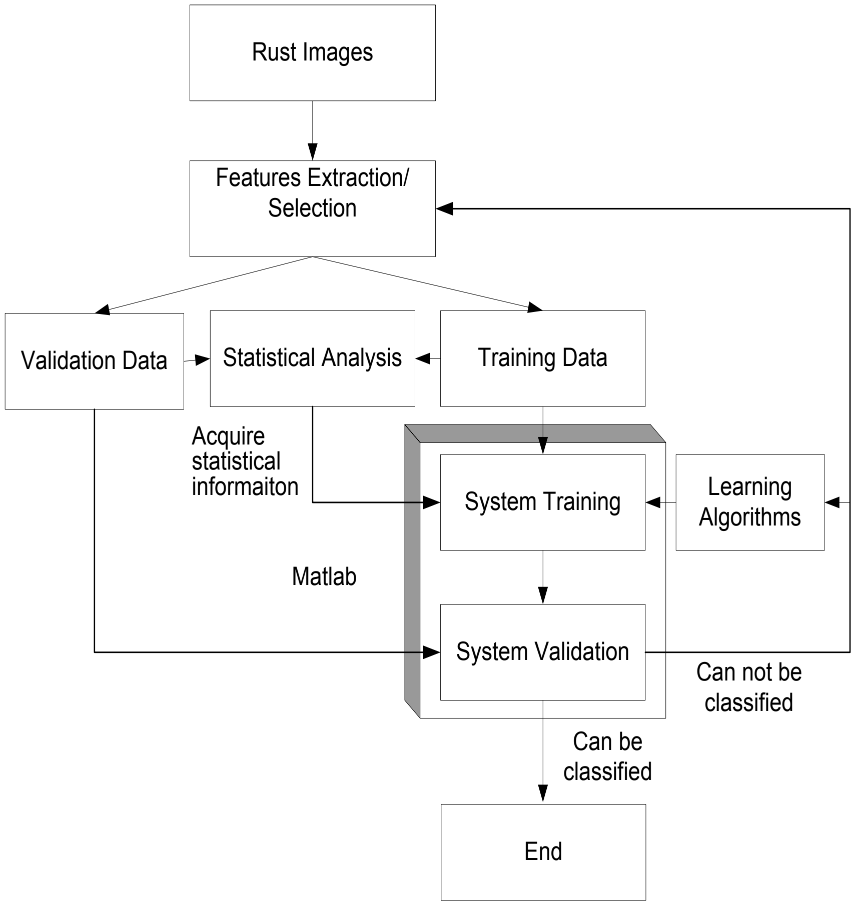


Figure 3.13 Overall structure of the classification system structure

# Chapter 4 - Results of Classification Experiments

## 4.1 Introduction

The Chapter 3 has listed the approaches of the classification experiments. Twelve different feature sets obtained from six rust groups are presented to both artificial neural networks and statistic pattern recognition methods.

In this chapter, the classification results of all cases (from binary classification to the classification of six classes) of different methods are stated. The results presented here show which method works and the performance of that method.

## 4.2 Vision Evaluation

The vision evaluation of the six rust groups will be discussed here before go through the results of pattern recognition methods.

Six different rust groups which have examined by various pattern recognition methods are chosen from a visual library representing the certain period time of rust 0, 1368, 2856, 4296, 5856, and 7008 hours. From the visual point of view, the class one (0 hour), class 2 (1368 hours) and class 6 (7008) are easy to be distinguished. However, the class 3 (2856 hours), class 4 (4296 hours) and class 5 (5856 hours) look very similar, which are begun to rust and from which the mill scale has begun to flake.

The three groups are difficult to be distinguished individually (without compare them together) by normal people (i.e. myself). The results will be used as references to measure whether or not the automated system can preference better than ordinary people.

## 4.3 The Results of Artificial Neural Networks

### 4.3.1 ADALINE neural network

ADALINE is a supervised neural network algorithm ad trained by a gradient descent rule, called LMS (Lest Mean Squares). The ADALINE network, much like perceptron network, can only solve linearly separable problems, and will only be used to examine the binary classification (see section 2.3.1).

#### 4.3.1.1 Binary classification

1) Binary classification of the class one

| | | Class 1 | Not Class 1 | Total |
|---|---|---|---|---|
| Count | Class 1 | 179 | 31 | 210 |
| | Not Class 1 | 147 | 903 | 1050 |
| % | Class 1 | **85.2%** | 14.8% | 100% |
| | Not Class 1 | 14.0% | **86.0%** | 100% |

Table 4.1 ADALINE - Evaluate results: the binary classification of the class one

2) Binary classification of the class two

| | | Class 2 | Not Class 2 | Total |
|---|---|---|---|---|
| Count | Class 2 | 169 | 41 | 210 |
| | Not Class 2 | 108 | 942 | 1050 |
| % | Class 2 | **80.5%** | 19.5% | 100% |
| | Not Class 2 | 10.3% | **89.7%** | 100% |

Table 4.2 ADALINE - Evaluate results: the binary classification of the class two

3) Binary classification of the class three

| | | Class 3 | Not Class 3 | Total |
|---|---|---|---|---|
| Count | Class 3 | 132 | 78 | 210 |
| | Not Class3 | 429 | 621 | 1050 |
| % | Class 3 | **62.9%** | 37.1% | 100% |
| | Not Class 3 | 40.9% | **59.1%** | 100% |

Table 4.3 ADALINE - Evaluate results: the binary classification of the class three

4) Binary classification of the class four

| | | Class 4 | Not Class 4 | Total |
|---|---|---|---|---|
| Count | Class 4 | 116 | 94 | 210 |
| | Not Class 4 | 502 | 548 | 1050 |
| % | Class 4 | **55.2%** | 44.8% | 100% |
| | Not Class 4 | 47.8% | **52.2%** | 100% |

Table 4.4 ADALINE - Evaluate results: the binary classification of the class four

5) Binary classification of the class five

| | | Class 5 | Not Class 5 | Total |
|---|---|---|---|---|
| Count | Class 5 | 111 | 99 | 210 |
| | Not Class 5 | 511 | 539 | 1050 |
| % | Class 5 | **52.9%** | 47.1% | 100% |
| | Not Class 5 | 48.7% | **51.3%** | 100% |

Table 4.5 ADALINE - Evaluate results: the binary classification of the class five

6) Binary classification of the class six

| | | Class6 | Not Class6 | Total |
|---|---|---|---|---|
| Count | Class6 | 149 | 61 | 210 |
| | Not Class6 | 241 | 809 | 1050 |
| % | Class6 | **71.0%** | 29.0% | 100% |

| | | | | |
|---|---|---|---|---|
| | Not Class6 | 23.0% | **77.0%** | 100% |

Table 4.6 ADALINE - Evaluate results: the binary classification of the class six

#### 4.3.1.2 Conclusions

After the training the network (see section 3.5.2 and 3.5.3), the test set has been examined to test the network performance. The percentages of correctly classified samples are listed. The results show that the ADALINE neural network is only able to performance the binary classification for class one, class two and class six. It fails to distinguish class four and five from other classes. It proves that the feature spaces which contains class four or five are not linearly separable. The limitation of ALAINE network fails its performances.

### 4.3.2 Learning and Vector Quantization (LVQ) Neural Network

#### 4.3.2.1 Binary classifications

1) Binary classification of the class one

| | | Class 1 | Not Class 1 | Total |
|---|---|---|---|---|
| Count | Class 1 | 201 | 9 | 210 |
| | Not Class 1 | 16 | 1034 | 1050 |
| % | Class 1 | **95.7%** | 4.3% | 100% |

| | | | | |
|---|---|---|---|---|
| | Not Class 1 | 1.5% | **98.5%** | 100% |

Table 4.7 LVQ - Evaluate results: the binary classification of the class one

2) Binary classification of the class two

| | | Class 2 | Not Class 2 | Total |
|---|---|---|---|---|
| Count | Class 2 | 198 | 12 | 210 |
| | Not Class 2 | 13 | 1037 | 1050 |
| % | Class 2 | **94.3%** | 5.7% | 100% |
| | Not Class 2 | 1.2% | **98.8%** | 100% |

Table 4.8 LVQ - Evaluate results: the binary classification of the class two

3) Binary classification of the class three

| | | Class 3 | Not Class 3 | Total |
|---|---|---|---|---|
| Count | Class 3 | 192 | 18 | 210 |
| | Not Class3 | 26 | 1024 | 1050 |
| % | Class 3 | **91.4%** | 8.6% | 100% |
| | Not Class 3 | 2.5% | **97.5%** | 100% |

Table 4.9 LVQ - Evaluate results: the binary classification of the class three

4) Binary classification of the class four

| | | Class 4 | Not Class 4 | Total |
|---|---|---|---|---|
| Count | Class 4 | 189 | 21 | 210 |
| | Not Class 4 | 38 | 1012 | 1050 |
| % | Class 4 | **90.0%** | 10.0% | 100% |
| | Not Class 4 | 3.6% | **96.4%** | 100% |

Table 4.10 LVQ - Evaluate results: the binary classification of the class four

5) Binary classification of the class five

| | | Class 5 | Not Class 5 | Total |
|---|---|---|---|---|
| Count | Class 5 | 191 | 19 | 210 |
| | Not Class 5 | 40 | 1010 | 1050 |
| % | Class 5 | **91.0%** | 9.0% | 100% |
| | Not Class 5 | 3.8% | **96.2%** | 100% |

Table 4.11 LVQ - Evaluate results: the binary classification of the class five

6) Binary classification of the class six

| | | Class6 | Not Class6 | Total |
|---|---|---|---|---|
| Count | Class6 | 197 | 13 | 210 |
| | Not Class6 | 31 | 1019 | 1050 |

| | | | | |
|---|---|---|---|---|
| % | Class6 | **93.8%** | 6.2% | 100% |
| | Not Class6 | 3.0% | **97.0%** | 100% |

Table 4.12 LVQ - Evaluate results: the binary classification of the class six

#### 4.3.2.2 The classification of three classes

| | | Class1 | Class3 | Class6 | Total |
|---|---|---|---|---|---|
| Count | Class 1 | 189 | 12 | 9 | 210 |
| | Class 2 | 8 | 194 | 8 | 210 |
| | Class 6 | 9 | 10 | 191 | 210 |
| % | Class 1 | **90.0%** | 5.7% | 4.3% | 100% |
| | Class 2 | 3.8% | **92.4%** | 3.8% | 100% |
| | Class 3 | 4.2% | 4.8% | **91.0%** | 100% |

Table 4.13 LVQ - Evaluate results: the classification of three classes

#### 4.3.2.3 The classification of four classes

| | | Class 1 | Class 2 | Class 3 | Class 6 | Total |
|---|---|---|---|---|---|---|
| Count | Class 1 | 191 | 11 | 7 | 1 | 210 |
| | Class 2 | 13 | 191 | 6 | | 210 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | Class 3 | 0 | 3 | 177 | 30 | 210 |
| | Class 6 | 0 | 5 | 15 | 190 | 210 |
| % | Class 1 | **91.0%** | 5.2% | 3.3% | 0.5% | 100% |
| | Class 2 | 6.2% | **91.0%** | 2.9% | 0.0% | 100% |
| | Class 3 | 0.0% | 1.4% | **84.3%** | 14.3% | 100% |
| | Class 6 | 0.0% | 2.4% | 7.1% | **90.5%** | 100% |

Table 4.14 LVQ - Evaluate results: the classification of four classes

#### 4.3.2.4 The classification of five classes

| Count | | Class 1 | Class 2 | Class 3 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|
| | Class 1 | 192 | 15 | 0 | 1 | 2 | 210 |
| | Class 2 | 0 | 207 | 0 | 3 | 0 | 210 |
| | Class 3 | 0 | 0 | 180 | 19 | 11 | 210 |
| | Class 5 | 0 | 3 | 14 | 179 | 14 | 210 |
| | Class 6 | 0 | 2 | 3 | 28 | 177 | 210 |
| % | Class 1 | **91.4%** | 7.1% | 0.0% | 0.5% | 1.0% | 100% |
| | Class 2 | 0.0% | **98.6%** | 0.0% | 1.4% | 0.0% | 100% |
| | Class 3 | 0.0% | 0.0% | **85.7%** | 9.0% | 5.3% | 100% |
| | Class 5 | 0.0% | 1.4% | 6.7% | **85.2%** | 6.7% | 100% |
| | Class 6 | 0.0% | 1.0% | 1.4% | 13.3% | **84.3%** | 100% |

Table 4.15 LVQ - Evaluate results: the classification of five classes

#### 4.3.2.5 The classification of six classes

| Count | | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|---|
| | Class 1 | 191 | 12 | 7 | 0 | 0 | 0 | 210 |
| | Class 2 | 23 | 187 | 0 | 0 | 0 | 0 | 210 |
| | Class 3 | 0 | 13 | 167 | 21 | 9 | 0 | 210 |
| | Class 4 | 0 | 0 | 11 | 162 | 28 | 9 | 210 |
| | Class 5 | 0 | 0 | 6 | 6 | 154 | 44 | 210 |
| | Class 6 | 0 | 0 | 5 | 12 | 37 | 156 | 210 |
| % | Class 1 | **91.0%** | 5.7% | 3.3% | 0.0% | 0.0% | 0.0% | 100% |
| | Class 2 | 11.0% | **89.0%** | 0.0% | 0.0% | 0.0% | 0.0% | 100% |
| | Class 3 | 0.0% | 6.2% | **79.5%** | 10.0% | 4.3% | 0.0% | 100% |
| | Class 4 | 0.0% | 0.0% | 5.2% | **77.1%** | 13.3% | 4.3% | 100% |
| | Class 5 | 0.0% | 0.0% | 2.9% | 2.9% | **73.3%** | 21.0% | 100% |
| | Class 6 | 0.0% | 0.0% | 2.4% | 5.7% | 17.6% | **74.3%** | 100% |

Table 4.16 LVQ - Evaluate results: the classification of six classes

#### 4.3.2.6 Conclusions

After the training the network (see section 3.5.2 and 3.5.3), the test set has been examined to test the network performance. The percentages of correctly classified samples are listed. The results have proved that the LVQ network has the ability to classify the rust classes up to six classes. However, the accuracies of the classification are not very high in the situation where six classes are classified. The more classes added, the less accuracy of classification is produced. The LVQ is method for training completive layers in a supervised manner (with target outputs). Input vectors are automatically learned by a competitive layer. The classes that the competitive layer found are dependent only on the distance between input vectors. The results provide the input vectors of class three, class four and class five are very close. The competitive layer confused with the input and has wrong decision sometime.

### 4.3.3 Back-Propagation with Momentum and Adaptive Learning Rate

#### 4.3.3.1 Binary classifications

1) Binary classification of the class one

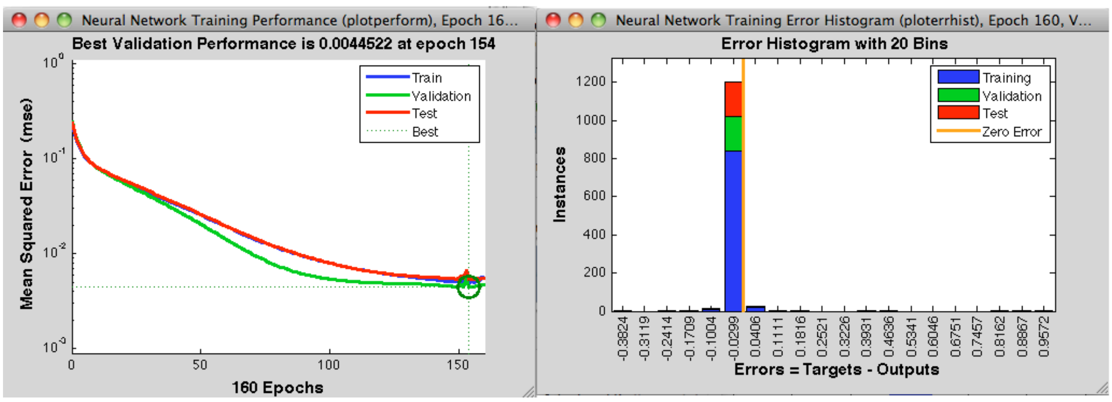


Figure 4.1 Training performance and error histogram: the binary classification of the

class one

| | | Class 1 | Not Class 1 | Total |
|---|---|---|---|---|
| Count | Class 1 | 206 | 4 | 210 |
| | Not Class 1 | 0 | 1050 | 1050 |
| % | Class 1 | **98.1%** | 1.9% | 100% |
| | Not Class 1 | 0.0% | **100.0%** | 100% |

Table 4.17 BP - Evaluate results: the binary classification of the class one

2) Binary classification of the class two

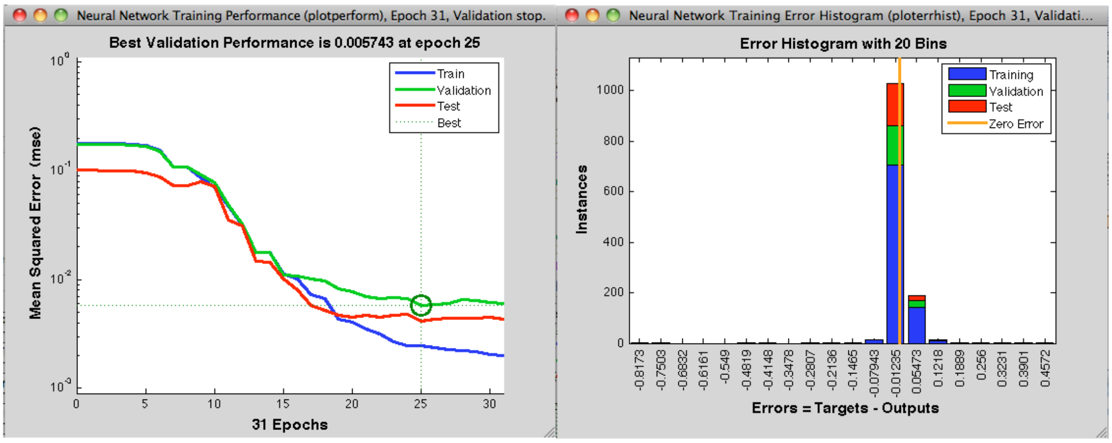


Figure 4.2 Training performance and error histogram: the binary classification of the class two

| | | Class 2 | Not Class 2 | Total |
|---|---|---|---|---|
| Count | Class 2 | 203 | 7 | 210 |

| | | | | |
|---|---|---|---|---|
| | Not Class 2 | 7 | 1043 | 1050 |
| % | Class 2 | **96.7%** | 3.3% | 100% |
| | Not Class 2 | 0.7% | **99.3%** | 100% |

Table 4.18 BP - Evaluate results: the binary classification of the class two

3) Binary classification of the class tree

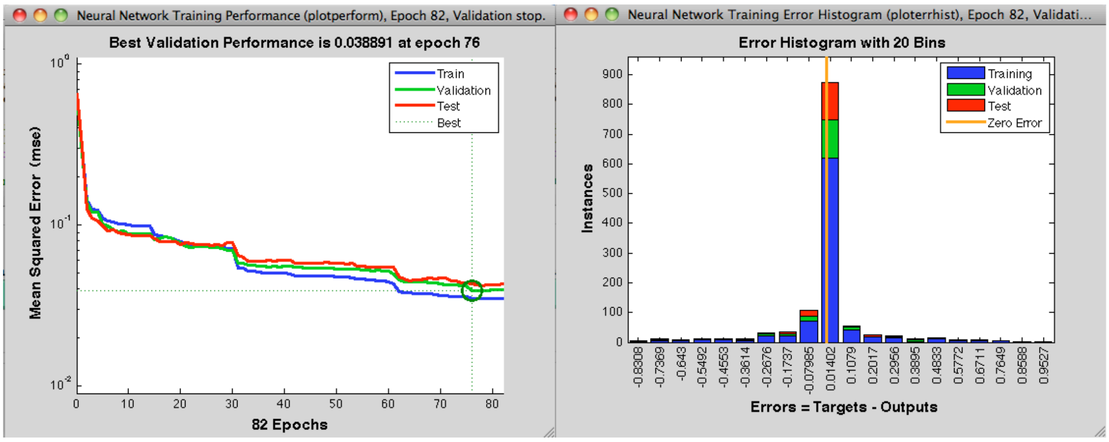


Figure 4.3 Training performance and error histogram: the binary classification of the class three

| | | Class 3 | Not Class 3 | Total |
|---|---|---|---|---|
| Count | Class 3 | 205 | 5 | 210 |
| | Not Class3 | 5 | 1045 | 1050 |
| % | Class 3 | **97.6%** | 2.4% | 100% |
| | Not Class 3 | 0.5% | **99.5%** | 100% |

Table 4.19 BP - Evaluate results: the binary classification of the class three

4) Binary classification of the class four

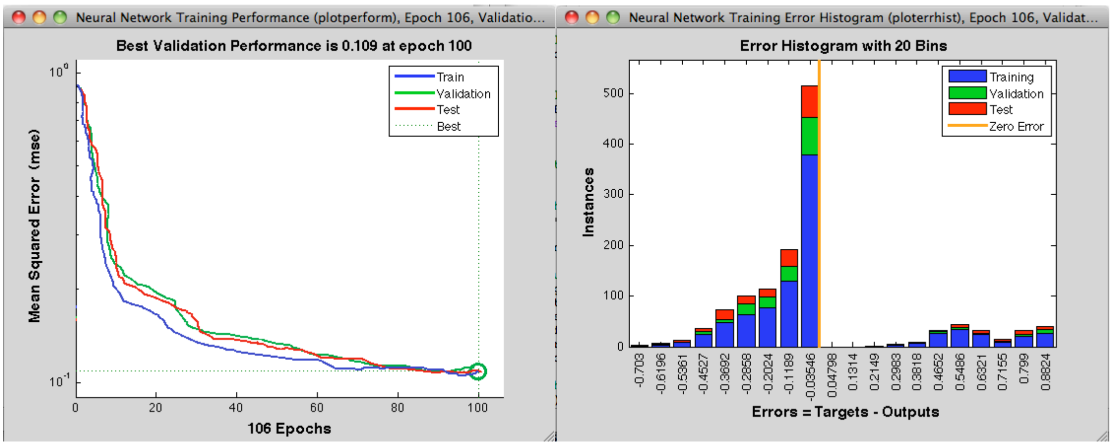


Figure 4.4 Training performance and error histogram: the binary classification of the class four

| | | Class 4 | Not Class 4 | Total |
|---|---|---|---|---|
| Count | Class 4 | 206 | 4 | 210 |
| | Not Class 4 | 41 | 1009 | 1050 |
| % | Class 4 | **98.1%** | 1.9% | 100% |
| | Not Class 4 | 4.0% | **96.0%** | 100% |

Table 4.20 BP - Evaluate results: the binary classification of the class four

5) Binary classification of the class five

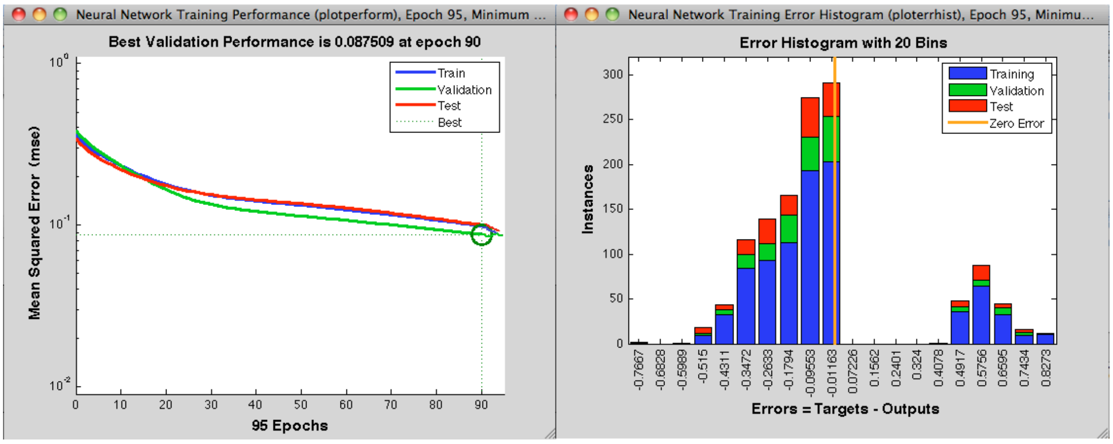


Figure 4.5 Training performance and error histogram: the binary classification of the class five

| | | Class 5 | Not Class 5 | Total |
|---|---|---|---|---|
| Count | Class 5 | 204 | 6 | 210 |
| | Not Class 5 | 39 | 1011 | 1050 |
| % | Class 5 | **97.1%** | 2.9% | 100% |
| | Not Class 5 | 3.7% | **96.3%** | 100% |

Table 4.21 BP - Evaluate results: the binary classification of the class five

6) Binary classification of the class six

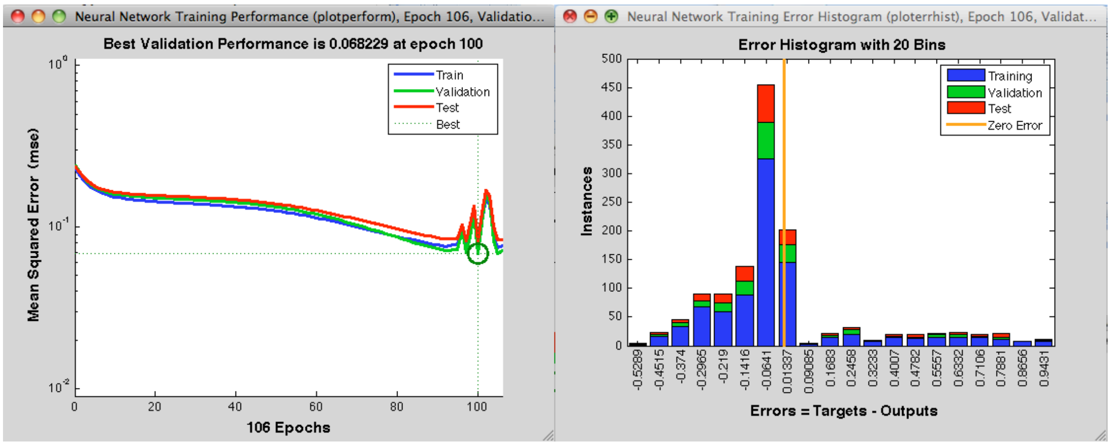


Figure 4.6 Training performance and error histogram: the binary classification of the class six

| | | Class6 | Not Class6 | Total |
|---|---|---|---|---|
| Count | Class6 | 170 | 40 | 210 |
| | Not Class6 | 6 | 1044 | 1050 |
| % | Class6 | **81.0%** | 19.0% | 100% |
| | Not Class6 | 0.6% | **99.4%** | 100% |

Table 4.22 BP Evaluate results: the binary classification of the class six

#### 4.3.3.2 The classification of three classes

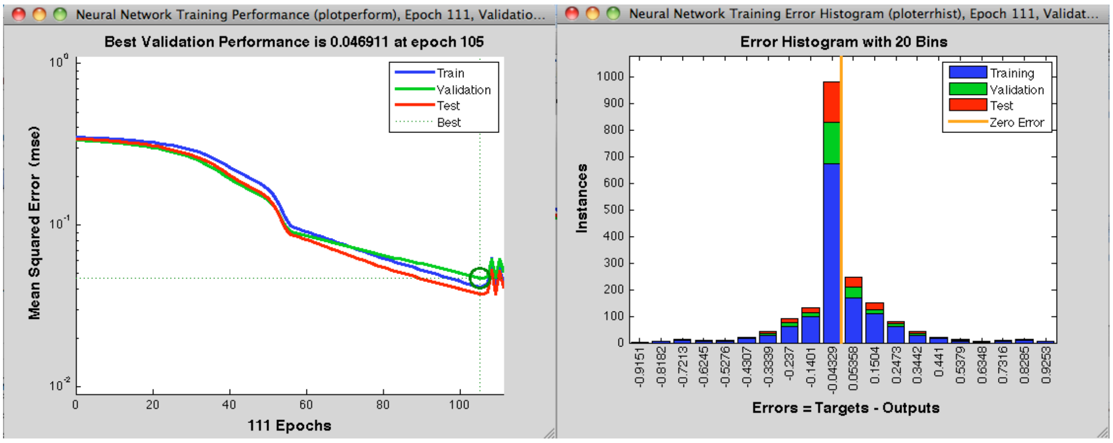


Figure 4.7 Training performance and error histogram: the classification of three classes

| | | Class1 | Class3 | Class6 | Total |
|---|---|---|---|---|---|
| Count | Class 1 | 205 | 5 | 0 | 210 |
| | Class 2 | 0 | 200 | 10 | 210 |
| | Class 6 | 2 | 16 | 192 | 210 |
| % | Class 1 | **97.6%** | 2.4% | 0.0% | 100% |
| | Class 2 | 0.0% | **95.2%** | 4.8% | 100% |
| | Class 3 | 1.0% | 7.6% | **91.4%** | 100% |

Table 4.23 BP - Evaluate results: the classification of three classes

### 4.3.3.3 The classification of four classes

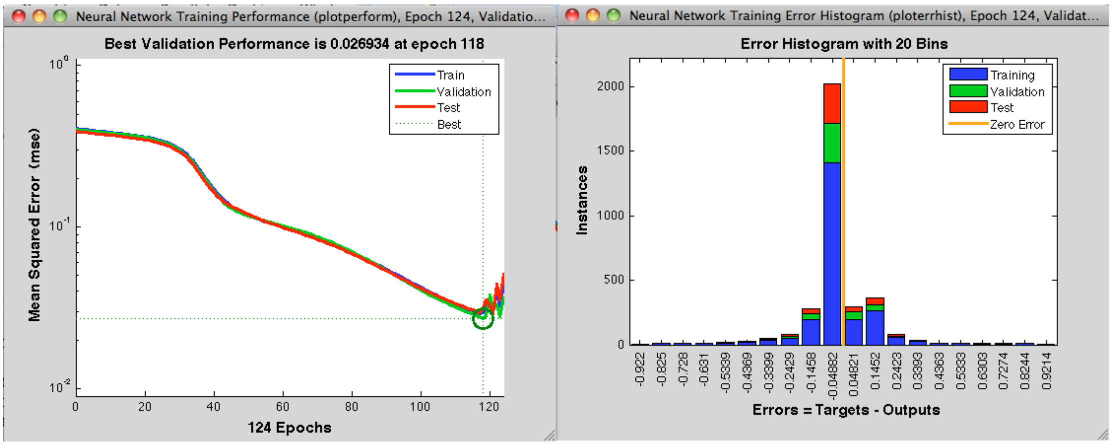


Figure 4.8 Training performance and error histogram: the classification of four classes

| | | Class 1 | Class 2 | Class 3 | Class 6 | Total |
|---|---|---|---|---|---|---|
| Count | Class 1 | 197 | 12 | 0 | 1 | 210 |
| | Class 2 | 0 | 207 | 0 | 3 | 210 |
| | Class 3 | 0 | 0 | 176 | 34 | 210 |
| | Class 6 | 0 | 0 | 1 | 209 | 210 |
| % | Class 1 | **93.8%** | 5.7% | 0.0% | 0.5% | 100% |
| | Class 2 | 0.0% | **98.6.%** | 0.0% | 1.4% | 100% |
| | Class 3 | 0.0% | 0.0% | **83.8%** | 16.2% | 100% |
| | Class 6 | 0.0% | 0.0% | 0.5% | **99.5.%** | 100% |

Table 4.24 BP - Evaluate results: the classification of four classes

#### 4.3.3.4 The classification of five classes

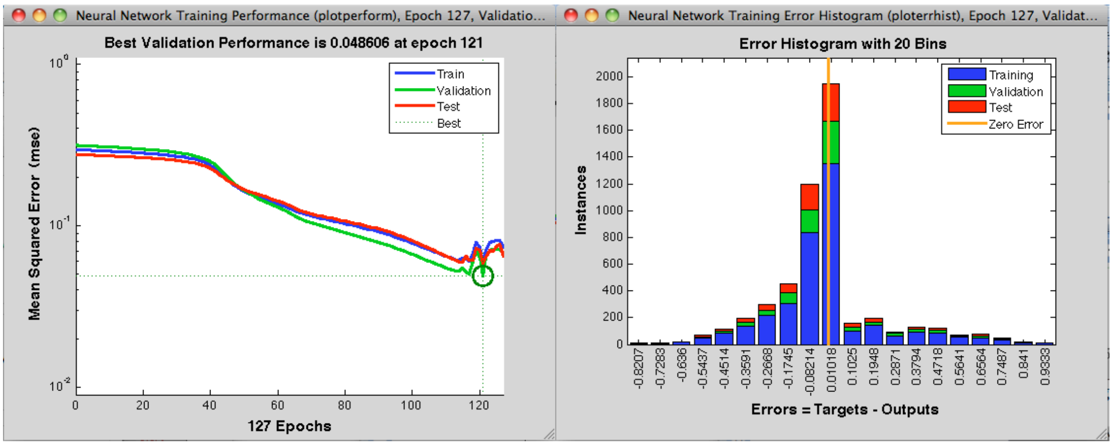


Figure 4.9 Training performance and error histogram: the classification of five classes

| Count | | Class 1 | Class 2 | Class 3 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|
| | Class 1 | 192 | 15 | 0 | 1 | 2 | 210 |
| | Class 2 | 0 | 207 | 0 | 3 | 0 | 210 |
| | Class 3 | 0 | 0 | 180 | 19 | 11 | 210 |
| | Class 5 | 0 | 3 | 14 | 179 | 14 | 210 |
| | Class 6 | 0 | 2 | 3 | 28 | 177 | 210 |
| % | Class 1 | **91.4%** | 7.1% | 0.0% | 0.5% | 1.0% | 100% |
| | Class 2 | 0.0% | **98.6%** | 0.0% | 1.4% | 0.0% | 100% |
| | Class 3 | 0.0% | 0.0% | **85.7%** | 9.0% | 5.3% | 100% |
| | Class 5 | 0.0% | 1.4% | 6.7% | **85.2%** | 6.7% | 100% |
| | Class 6 | 0.0% | 1.0% | 1.4% | 13.3% | **84.3%** | 100% |

Table 4.25 BP - Evaluate results: the classification of five classes

### 4.3.3.5 The classification of six classes

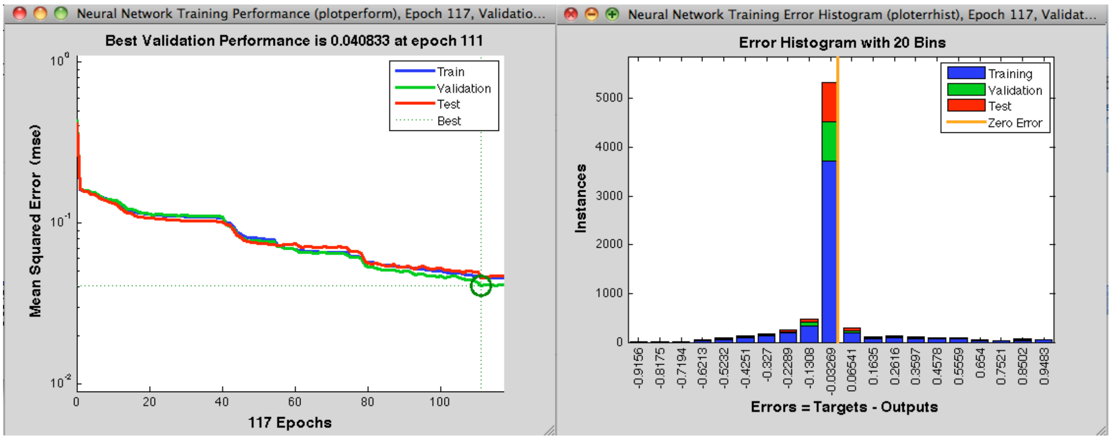


Figure 4.10 Training performance and error histogram: the classification of six classes

| Count | | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|---|
| | Class 1 | 196 | 10 | 0 | 0 | 4 | 0 | 210 |
| | Class 2 | 0 | 203 | 7 | 0 | 0 | 0 | 210 |
| | Class 3 | 0 | 4 | 178 | 8 | 20 | 0 | 210 |
| | Class 4 | 0 | 0 | 11 | 169 | 25 | 5 | 210 |
| | Class 5 | 0 | 0 | 6 | 2 | 168 | 34 | 210 |
| | Class 6 | 0 | 3 | 0 | 0 | 3 | 204 | 210 |
| % | Class 1 | **93.3%** | 4.8% | 0.0% | 0.0% | 1.9% | 0.0% | 100% |
| | Class 2 | 0.0% | **96.7%** | 3.3% | 0.0% | 0.0% | 0.0% | 100% |
| | Class 3 | 0.0% | 1.9% | **84.8%** | 3.8% | 9.5% | 0.0% | 100% |
| | Class 4 | 0.0% | 0.0% | 5.2% | **80.5%** | 11.9% | 2.4% | 100% |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Class 5 | 0.0% | 0.0% | 2.9% | 1.0% | **80.0%** | 16.1% | 100% |
| | Class 6 | 0.0% | 1.4% | 0.0% | 0.0% | 1.4% | **97.2%** | 100% |

Table 4.26 BP - Evaluate results: the classification of six classes

#### 4.3.3.6 Conclusions

After the training the network (see section 3.5.2 and 3.5.3), the test set has been examined to test the network performance. The percentages of correctly classified samples are listed. The results show the gradient descent with momentum and adaptive learning rate back-propagation has the ability to classify up to six rust classes with very high accuracy. The results have been obtained by training the network with different initial conditions to avoid trapping in a local minimum. The cross-validated has also been applied during the training progress.

Back-propagation with standard gradient descent algorithm had also been tested as a reference. The network fails to classify no more than three classes. The heuristic modifications really help the performance of the back-propagation network.

## 4.4 The Results of Statistical Pattern Recognition Methods

### 4.4.1 Discriminant Analysis

#### 4.4.1.1 Binary classification

1) Binary classification of the class one

| | | Class 1 | Not Class 1 | Total |
|---|---|---|---|---|
| Count | Class 1 | 420 | 0 | 420 |
| | Not Class 1 | 0 | 2100 | 2100 |
| % | Class 1 | **100%** | 0.0% | 100% |
| | Not Class 1 | 0.0% | **100.%** | 100% |

Table 4.27 Discriminant analysis cross-validated result: Binary classification of the class one

2) Binary classification of the class two

| | | Class 2 | Not Class 2 | Total |
|---|---|---|---|---|
| Count | Class 2 | 420 | 0 | 420 |
| | Not Class 2 | 0 | 2100 | 2100 |
| % | Class 2 | **100%** | 0.0% | 100% |
| | Not Class 2 | 0.0% | **100%** | 100% |

Table 4.28 Discriminant analysis cross-validated result: Binary classification of the

class two

3) Binary classification of the class three

| | | Class 3 | Not Class 3 | Total |
|---|---|---|---|---|
| Count | Class 3 | 404 | 16 | 420 |
| | Not Class3 | 116 | 1984 | 2100 |
| % | Class 3 | **96.2%** | 3.8% | 100% |
| | Not Class 3 | 5.5% | **94.5%** | 100% |

Table 4.29 Discriminant analysis cross-validated result: Binary classification of the class three

4) Binary classification of the class four

| | | Class 4 | Not Class 4 | Total |
|---|---|---|---|---|
| Count | Class 4 | 386 | 34 | 420 |
| | Not Class 4 | 162 | 1938 | 2100 |
| % | Class 4 | **91.9%** | 8.1% | 100% |
| | Not Class 4 | 7.7% | **92.3%** | 100% |

Table 4.30 Discriminant analysis cross-validated result: Binary classification of the class four

5) Binary classification of the class five

| | | Class 5 | Not Class 5 | Total |
|---|---|---|---|---|
| Count | Class 5 | 380 | 40 | 420 |
| | Not Class 5 | 178 | 1922 | 2100 |
| % | Class 5 | **90.5%** | 9.5% | 100% |
| | Not Class 5 | 8.5% | **91.5%** | 100% |

Table 4.31 Discriminant analysis cross-validated result: Binary classification of the class five

6) Binary classification of the class six

| | | Class6 | Not Class6 | Total |
|---|---|---|---|---|
| Count | Class6 | 372 | 48 | 420 |
| | Not Class6 | 254 | 1846 | 2100 |
| % | Class6 | **88.6%** | 11.4% | 100% |
| | Not Class6 | 12.1% | **87.9%** | 100% |

Table 4.32 Discriminant analysis cross-validated result: Binary classification of the class six

### 4.4.1.2 Classification of three classes

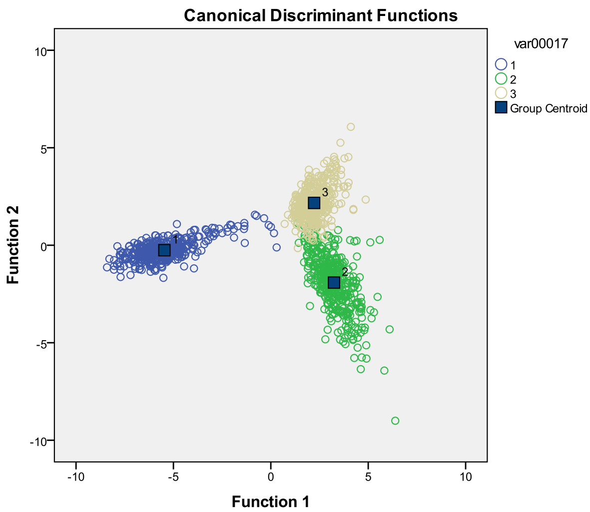


Figure 4.11 Canonical discriminate functions of three classes

| | | Class1 | Class3 | Class6 | Total |
|---|---|---|---|---|---|
| Count | Class 1 | 409 | 0 | 11 | 420 |
| | Class 2 | 0 | 390 | 30 | 420 |
| | Class 6 | 0 | 0 | 420 | 210 |
| % | Class 1 | **97.4%** | 0.0% | 2.6% | 100% |
| | Class 2 | 0.0% | **92.9%** | 7.1% | 100% |
| | Class 3 | 0.0% | 0.0% | **100.0%** | 100% |

Table 4.33 Discriminant analysis cross-validated result: Classification of three classes

### 4.4.1.3 Classification of four classes

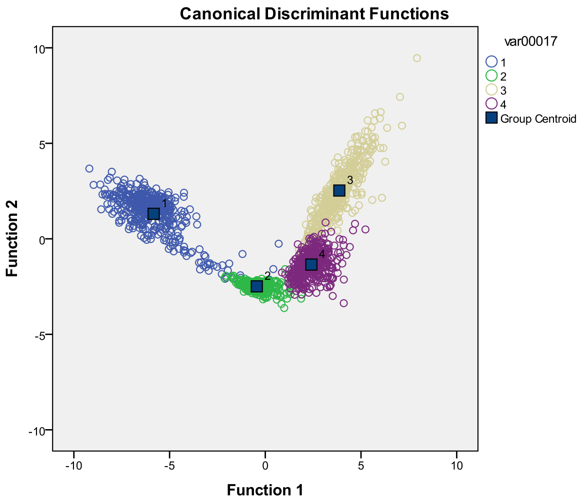


Figure 4.12 Canonical discriminate functions of four classes

| | | Class 1 | Class 2 | Class 3 | Class 6 | Total |
|---|---|---|---|---|---|---|
| Count | Class 1 | 389 | 31 | 0 | 0 | 420 |
| | Class 2 | 0 | 411 | 0 | 9 | 420 |
| | Class 3 | 0 | 1 | 389 | 30 | 420 |
| | Class 6 | 0 | 8 | 0 | 412 | 420 |
| % | Class 1 | **92.6%** | 7.4% | 0.0% | 1.4% | 100% |
| | Class 2 | 8.1% | **97.9%** | 0.0% | 2.1% | 100% |
| | Class 3 | 0.0% | 0.2% | **92.6%** | 7.1% | 100% |
| | Class 6 | 0.0% | 1.9% | 0.0% | **98.1%** | 100% |

Table 4.34 Discriminant analysis cross-validated result: Classification of four classes

### 4.4.1.4 Classification of five classes

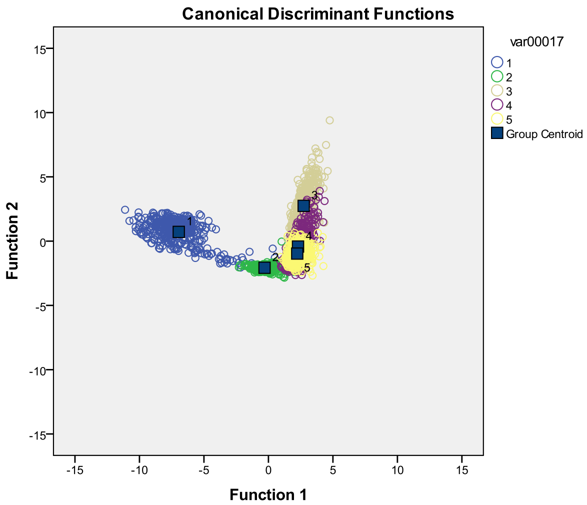


Figure 4.13 Canonical discriminate functions of five classes

| Count | | Class 1 | Class 2 | Class 3 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|
| | Class 1 | 389 | 31 | 0 | 0 | 0 | 420 |
| | Class 2 | 0 | 411 | 0 | 1 | 8 | 420 |
| | Class 3 | 0 | 1 | 376 | 28 | 15 | 420 |
| | Class 5 | 0 | 10 | 16 | 352 | 42 | 420 |
| | Class 6 | 0 | 3 | 3 | 78 | 336 | 420 |
| % | Class 1 | **92.6%** | 7.4% | 0.0% | 0.0% | 0.0% | 100% |
| | Class 2 | 0.0% | **97.9%** | 0.0% | 0.2% | 1.9% | 100% |
| | Class 3 | 0.0% | 0.2% | **89.5%** | 6.7% | 3.6% | 100% |
| | Class 5 | 0.0% | 2.4% | 3.8% | **83.8%** | 10.0% | 100% |
| | Class 6 | 0.0% | 0.7% | 0.7% | 18.6% | **80.0%** | 100% |

Table 4.35 Discriminant analysis cross-validated result: Classification of five classes

#### 4.4.1.5 Classification of six classes

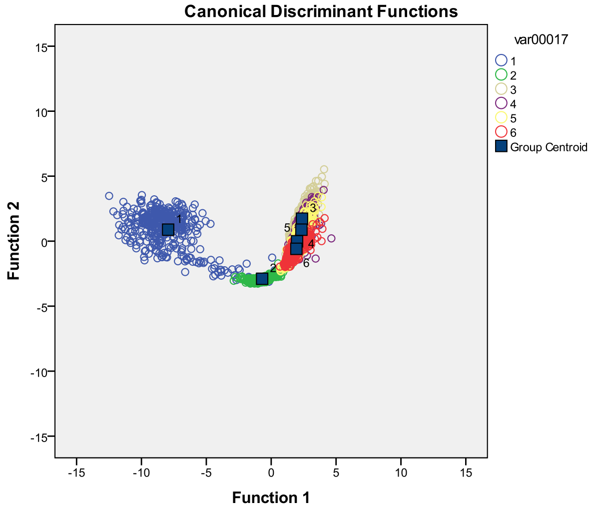


Figure 4.14 Canonical discriminate functions of six classes

| Count | | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|---|
| | Class 1 | 388 | 32 | 0 | 0 | 0 | 0 | 420 |
| | Class 2 | 0 | 412 | 0 | 0 | 1 | 7 | 420 |
| | Class 3 | 0 | 0 | 336 | 63 | 11 | 10 | 420 |
| | Class 4 | 0 | 1 | 86 | 189 | 98 | 46 | 420 |
| | Class 5 | 0 | 7 | 5 | 66 | 309 | 33 | 420 |
| | Class 6 | 0 | 2 | 1 | 15 | 80 | 322 | 420 |
| % | Class 1 | **92.4%** | 7.6% | 0.0% | 0.0% | 0.0% | 0.0% | 100% |
| | Class 2 | 0.0% | **98.1%** | 0.0% | 0.0% | 0.2% | 1.7% | 100% |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Class 3 | 0.0% | 0.0% | **80.0%** | 15.0% | 2.6% | 2.4% | 100% |
| | Class 4 | 0.0% | 0.2% | 20.5% | **45.0%** | 23.3% | 11.0% | 100% |
| | Class 5 | 0.0% | 1.7% | 1.2% | 15.7% | **73.6%** | 7.9% | 100% |
| | Class 6 | 0.0% | 0.5% | 0.2% | 3.6% | 19.0% | **76.7%** | 100% |

Table 4.36 Discriminant analysis cross-validated result: Classification of six classes

#### 4.4.1.6 Conclusions

The summary of number and percent of subjects classified correctly and incorrectly is the classification result. The 'leave-one-out classification' is a cross-validation method of which the results are also presented.

The graphs of canonical discriminate function have also been provided, which can be used to indicate the relationships between classes in 2D view.

The discriminant analysis is able to classify up to five different classes with acceptable accuracy. Although 77.6% cases can be correctly classified when it has been applied in the six classes case, but it is difficult for the discriminant analysis to distinguish the class four from other classes. It will only be able to generate the discriminant functions when groups are not overlapped. It is not the case when feature space contains six classes. The Figure 4.14 has also proved this.

### 4.4.2 K-Nearest Neighbour Methods (KNN)

#### 4.4.2.1 Binary classification

1) Binary classification of the class one

| | | Class 1 | Not Class 1 | Total |
|---|---|---|---|---|
| Count | Class 1 | 402 | 18 | 420 |
| | Not Class 1 | 0 | 2100 | 2100 |
| % | Class 1 | **95.7%** | 4.3% | 100% |
| | Not Class 1 | 0.0% | **100.0%** | 100% |

Table 4.37 KNN result: Binary classification of the class one

2) Binary classification of the class two

| | | Class 2 | Not Class 2 | Total |
|---|---|---|---|---|
| Count | Class 2 | 397 | 23 | 420 |
| | Not Class 2 | 87 | 2013 | 2100 |
| % | Class 2 | **94.5%** | 5.6% | 100% |
| | Not Class 2 | 4.1% | **95.9%** | 100% |

Table 4.38 KNN result: Binary classification of the class two

3) Binary classification of the class three

| | | Class 3 | Not Class 3 | Total |
|---|---|---|---|---|
| Count | Class 3 | 401 | 19 | 420 |
| | Not Class3 | 121 | 1979 | 2100 |
| % | Class 3 | **95.5%** | 4.5% | 100% |
| | Not Class 3 | 5.8% | **94.2%** | 100% |

Table 4.39 KNN result: Binary classification of the class three

4) Binary classification of the class four

| | | Class 4 | Not Class 4 | Total |
|---|---|---|---|---|
| Count | Class 4 | 379 | 41 | 420 |
| | Not Class 4 | 169 | 1931 | 2100 |
| % | Class 4 | **90.2%** | 9.8% | 100% |
| | Not Class 4 | 8.0% | **92.0%** | 100% |

Table 4.40 KNN result: Binary classification of the class four

5) Binary classification of the class five

| | | Class 5 | Not Class 5 | Total |
|---|---|---|---|---|
| Count | Class 5 | 368 | 52 | 420 |

| | | | | |
|---|---|---|---|---|
| | Not Class 5 | 197 | 1903 | 2100 |
| % | Class 5 | **87.6%** | 12.4% | 100% |
| | Not Class 5 | 9.4% | **90.6%** | 100% |

Table 4.41 KNN result: Binary classification of the class five

6) Binary classification of the class six

| | | Class6 | Not Class6 | Total |
|---|---|---|---|---|
| Count | Class6 | 379 | 41 | 420 |
| | Not Class6 | 239 | 1861 | 2100 |
| % | Class6 | **90.2%** | 9.8% | 100% |
| | Not Class6 | 11.4% | **88.6%** | 100% |

Table 4.42 KNN result: Binary classification of the class six

#### 4.4.2.2 Classification of three classes

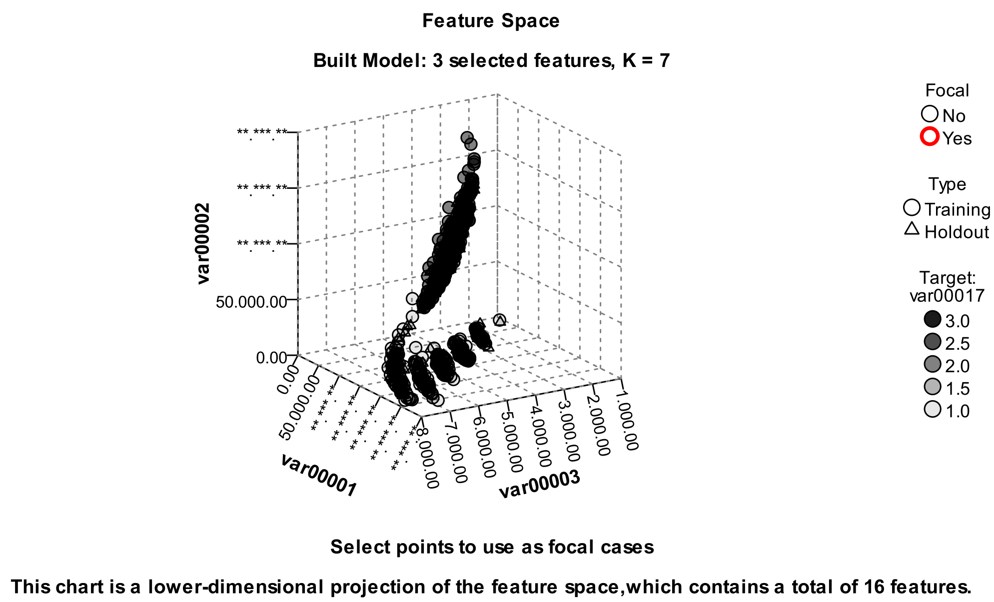


Figure 4.15 KNN feature space – classification of three classes

| | | Class1 | Class3 | Class6 | Total |
|---|---|---|---|---|---|
| Count | Class 1 | 415 | 4 | 1 | 420 |
| | Class 2 | 0 | 393 | 27 | 420 |
| | Class 6 | 0 | 17 | 403 | 420 |
| % | Class 1 | **98.8%** | 1.0% | 0.2% | 100% |
| | Class 2 | 0.0% | **93.6%** | 6.4% | 100% |
| | Class 3 | 0.0% | 4.0% | **96.0%** | 100% |

Table 4.43 KNN result: Classification of three classes

### 4.4.2.3 Classification of four classes

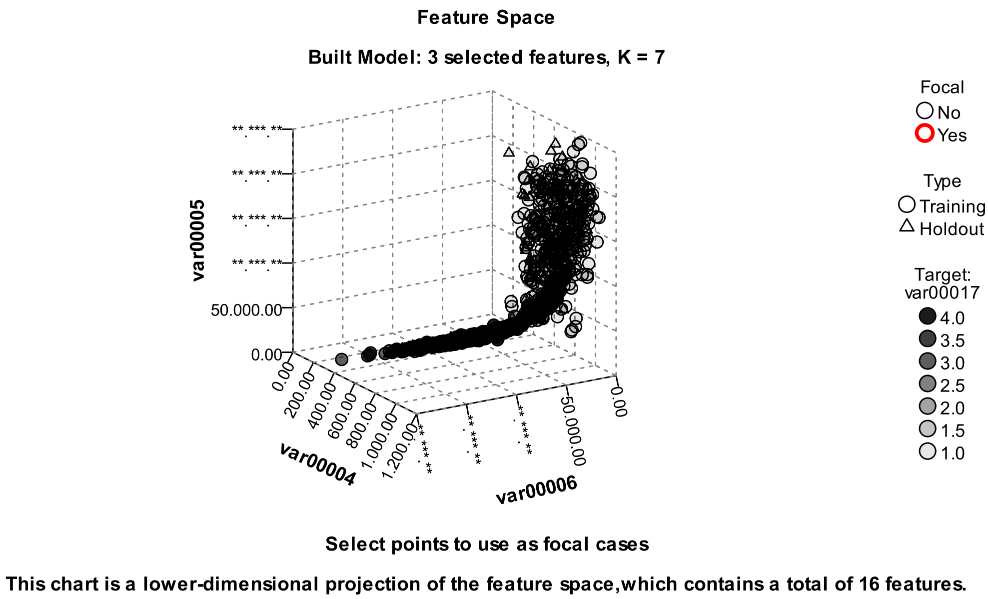


Figure 4.16 KNN feature space – classification of four classes

| | | Class 1 | Class 2 | Class 3 | Class 6 | Total |
|---|---|---|---|---|---|---|
| Count | Class 1 | 407 | 13 | 0 | 0 | 420 |
| | Class 2 | 1 | 417 | 2 | 0 | 420 |
| | Class 3 | 0 | 0 | 398 | 22 | 420 |
| | Class 6 | 0 | 1 | 11 | 408 | 420 |
| % | Class 1 | **96.9%** | 3.1% | 0.0% | 0.0% | 100% |
| | Class 2 | 0.2% | **99.3%** | 0.5% | 0.0% | 100% |
| | Class 3 | 0.0% | 0.0% | **94.8%** | 5.2% | 100% |
| | Class 6 | 0.0% | 0.2% | 2.6% | **97.1%** | 100% |

Table 4.44 KNN result: Classification of four classes

#### 4.4.2.4 Classification of five classes

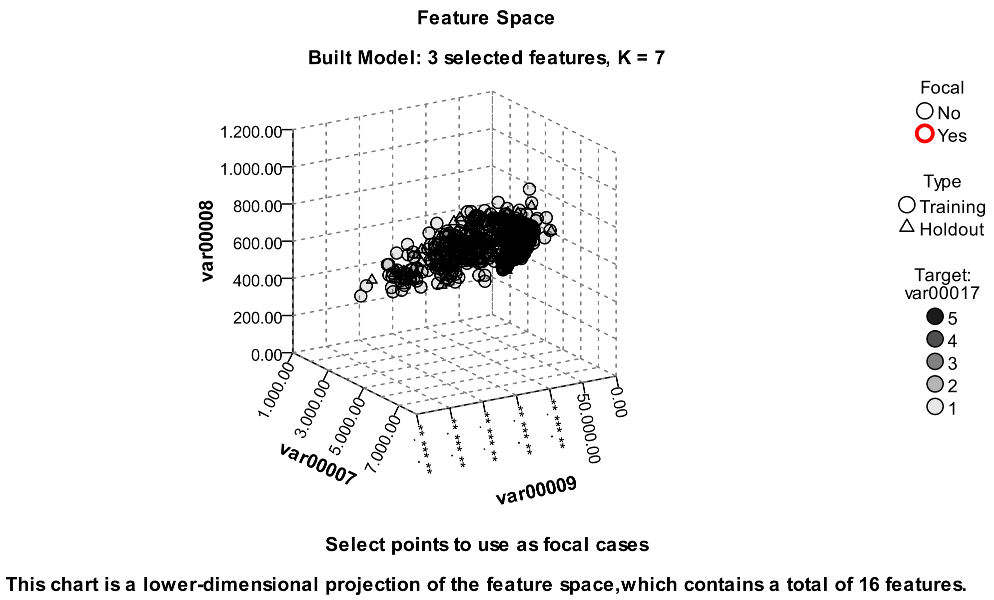


Figure 4.17 KNN feature space – classification of five classes

| Count | | Class 1 | Class 2 | Class 3 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|
| | Class 1 | 407 | 13 | 0 | 0 | 0 | 420 |
| | Class 2 | 0 | 414 | 6 | 0 | 0 | 420 |
| | Class 3 | 0 | 0 | 361 | 52 | 7 | 420 |
| | Class 5 | 0 | 0 | 22 | 337 | 61 | 420 |
| | Class 6 | 0 | 0 | 2 | 67 | 351 | 420 |
| % | Class 1 | **96.9%** | 3.1% | 0.0% | 0.0% | 0.0% | 100% |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | Class 2 | 0.0% | **98.6%** | 1.4% | 0.0% | 0.0% | 100% |
| | Class 3 | 0.0% | 0.0% | **86.0%** | 12.4% | 1.7% | 100% |
| | Class 5 | 0.0% | 0.0% | 5.2% | **80.2%** | 14.5% | 100% |
| | Class 6 | 0.0% | 0.0% | 0.5% | 16.0% | **83.6%** | 100% |

Table 4.45 KNN result: Classification of five classes

#### 4.4.2.5 Classification of six classes

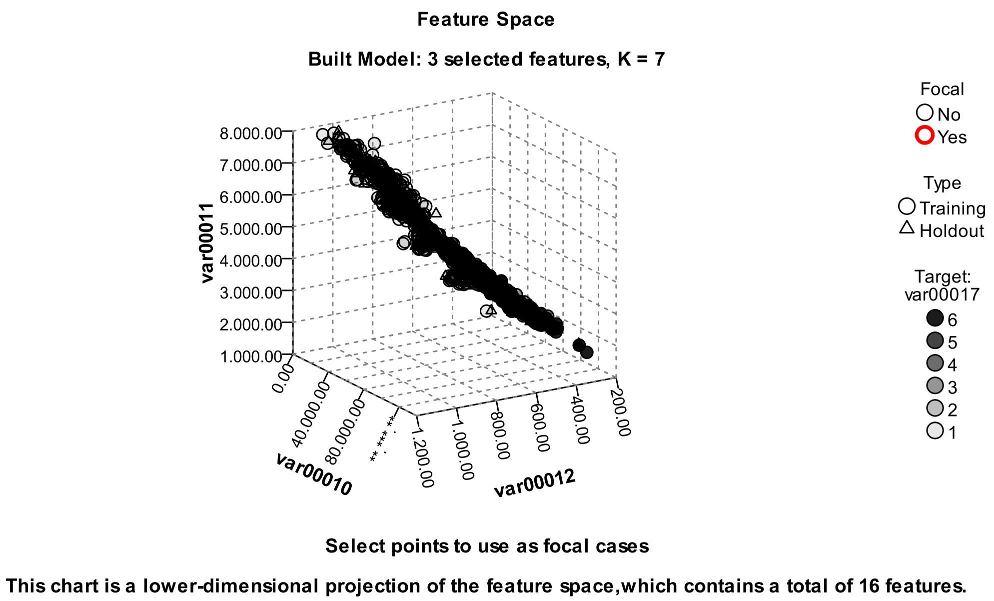


Figure 4.18 KNN feature space – classification of six classes

| Count | | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Class 6 | Total |
|---|---|---|---|---|---|---|---|---|
| | Class 1 | 407 | 13 | 0 | 0 | 0 | 0 | 420 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Class 2 | 0 | 413 | 4 | 3 | 0 | 0 | 420 |
| | Class 3 | 0 | 0 | 251 | 148 | 19 | 2 | 420 |
| | Class 4 | 0 | 0 | 33 | 243 | 133 | 11 | 420 |
| | Class 5 | 0 | 0 | 3 | 96 | 284 | 37 | 420 |
| | Class 6 | 0 | 0 | 1 | 20 | 163 | 236 | 420 |
| % | Class 1 | **96.9%** | 3.1% | 0.0% | 0.0% | 0.0% | 0.0% | 100% |
| | Class 2 | 0.0% | **98.3%** | 1.0% | 0.7% | 0.0% | 0.0% | 100% |
| | Class 3 | 0.0% | 0.0% | **59.8%** | 35.2% | 4.5% | 0.5% | 100% |
| | Class 4 | 0.0% | 0.0% | 7.9% | **57.9%** | 31.7% | 2.6% | 100% |
| | Class 5 | 0.0% | 0.0% | 0.7% | 22.9% | **67.6%** | 8.8% | 100% |
| | Class 6 | 0.0% | 0.0% | 0.2% | 4.8% | 38.8% | **56.2%** | 100% |

Table 4.46 KNN result: Classification of six classes

#### 4.4.2.6 Conclusions

The predicated values which represent average weighted by inverse distance are used as results to examine the performance of KNN method. There are two steps to calculating the predicated value from a set of KNN training date. The first is identifying n, or the number of nearest neighbours to use for this calculation. The second is to weight the contribution to the predicted value.

As a non-parametric discrimination technique, KNN has very similar results compared with discriminant analysis. It has done a brilliant job to classify up to five classes but not performed well in the case to classify six classes.

### 4.5 Conclusions

In this chapter, the feature sets obtained from the rust image are examined by five different classification methods

The five classification methods fall into two categories - artificial neural network method and statistic pattern recognition methods.

The classes need to be well represented in the feature space in order to have good classification results. The previous researches (see section 1.3) have proved the features extracted by co-occurrence matrix method can well represent the different classes.

For each surface type (rust class), the results of every classification methods are given in the tables as well as some graphs to provide some extra information.

The results of the experiments clearly show that the ADALINE method is the weakest classification method. The feature space of rust images is too complicated to be linearly separated and the ADALINE only has the ability to solve the linearly separated problem.

Two statistic pattern recognition methods performed well to classify up to five classes but found difficulty to handle the six classes' classification. It is caused by the some groups are overlapped in the feature space.

The co-occurrence matrix method describes the spatial relation between tonal pixels. The performance of the co-occurrence matrix is affected by three parameters, which are the size of the co-occurrence matrix, and size of the image and the distance between two neighbouring pixels. Twelve different data sets are obtained from the combinations of those three parameter, and classification result made by back-propagation (with momentum and adaptive learning rate) are list in Table 4.47, Where: M – size of the co-occurrence matrix, I – size of the sub image and D – the distance between two neighbouring pixels.

| Feature Set ( M/I/D) | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Class 6 |
|---|---|---|---|---|---|---|
| 32/32/1 | 41.3% | 49.4% | 42.3% | 46.1% | 41.2% | 51.2% |
| 32/32/6 | 82.8% | 89.4% | 79.6% | 76.3% | 73.3% | 89.4% |
| 32/32/12 | 89.1% | 90.1% | 81.5% | 80.2% | 79.5% | 94.5% |
| 32/64/1 | 42.5% | 52.2% | 40.5% | 43.5% | 41.3% | 50.2% |
| 32/64/6 | 83.1% | 89.7% | 79.2% | 77.9% | 79.3% | 85.7% |
| 32/64/12 | 92.3.9% | 94.5% | 83.5% | 80.3% | 81.2% | 96.7% |

| | | | | | | |
|---|---|---|---|---|---|---|
| 64/32/1 | 43.5% | 51.2% | 45.1% | 45.1% | 43.5% | 51.6% |
| 64/32/6 | 84.2% | 87.3% | 80.2% | 80.2% | 74.1% | 87.2% |
| 64/32/12 | 92.1% | 95.1% | 82.4% | 81.7% | 79.9% | 95.5% |
| 64/64/1 | 44.2% | 51.3% | 44.2% | 45.7% | 44.2% | 50.9% |
| 64/64/6 | 84.5% | 88.2% | 80.3% | 79.9% | 74.3% | 86.9% |
| 64/64/12 | 93.3% | 96.7% | 84.8% | 80.5% | 80.0% | 97.2% |

Table 4.47 The performances of the BP trained by different feature sets

The results show that the distance of co-occurrence matrix method is the most important feature to solve the rust classification problem. The feature represents rust grade can be obtained from studying the spatial relationship in a large area.

The results show that although class three, class four and class five can be classified, but as the results of vision elevation, every single pattern recognition methods have its worst performance when classifying those three classes.

Table 4.48 lists the overall performances of all classification methods (the percentage of cases correctly classified against the number of all samples).

| | ADALINE | LVQ | BP | DA | KNN |
|---|---|---|---|---|---|
| Binary classification (class one) | 85.9% | 98.0% | **99.7%** | 100.0% | 99.3% |

| | | | | | |
|---|---|---|---|---|---|
| Binary classification (class two) | 88.2% | 98.0% | **98.9%** | 100% | 95.6% |
| Binary classification (class three) | 59.8% | 96.5% | **99.2%** | 94.8% | 94.4% |
| Binary classification (class four) | 52.7% | 95.3% | **96.4%** | 92.2% | 91.7% |
| Binary classification (class five) | 51.6% | 95.3% | **96.4%** | 91.3% | 90.1% |
| Binary classification (class six) | 76.0% | 96.5% | **96.3%** | 88.0% | 88.9% |
| Classification of three classes | N/A | 91.1% | **94.8%** | 96.7% | 96.1% |
| Classification of four classes | N/A | 89.2% | **93.9%** | 95.3% | 97.0% |
| Classification of five classes | N/A | 89.0% | **89.0%** | 88.8% | 89.0% |
| Classification of six classes | N/A | 80.7% | **88.7%** | 77.6% | 72.8% |

Table 4.48 The overall performances of all classification methods

The overall performances are proved that the back-propagation trained by the adaptive learning rate and momentum training is the most powerful classification method in this research. This method has the consistent performances of all cases.

Steel surface of the six class are taken from a visual library based the continuous time scale (1368 hours different). The result indicates that, classification of continuous and less subjective grades of the rust can be implemented in real time applications.

## Chapter 5 - Conclusions

In this thesis, the problems which steel surface preparation industry are currently facing (see section 1.1) are stated. This thesis propose that the development of surface preparation technologies make it essential to develop the non-destructive testing techniques which allows patch restore of corrode steel structure in practice.

Various pattern recognition methods and texture analysis methods (see Chapter 2) have been reviewed in order to find a right method to solve the problem of steel surface classification.

The Chapter 3 states the main steps in practising the pattern recognition methods. Two most popular pattern recognition approaches are applied and compared.

The primary contribution is that the pattern recognition algorithms can achieve the non-destructive testing of the continuous and less subjective rust grades of steel surfaces.

Some conclusions can be made based on research evidence:

- The co-occurrence matrices texture analysis methods can be used to extract features for classification of steel surface.

- A proper constructed and well-trained artificial neural network can be applied to create a reliable rust condition and classification system.
- The artificial neural network pattern recognition system outperformed than the statistic pattern recognition system for this particular problem. It due to the complexity of the feature space which are clearly not normally distributed when more classes are involved.

Future studies for this research can focus on the following areas:

a) For industrial realization, future study can be focused on the implementation the real industrial applications. This research provides a simulation system of rust classification. However, in the real industrial application, the images will be captured with a CCD camera that is assembled into a robot in a real environment. In this case, not all conditions (or restriction of the image library can be met (such as distance between the samples and the CCD camera, the light condition etc.). More researches need to be done to improve the robust ability of the system.

b) More feature extraction methods and pattern recognition methods can be studied to generate the better results. For example increate the accuracy of the classification, or classify more classes.

c) Combining multiple classifiers to achieve better results.

a. Combination by voting principle; a simple method to combine results provided by different classifiers to interpret each classification results as a vote. Consequently, the data class that receives a number of votes higher than a prefixed threshold is taken as the 'final' classification. Typically, the threshold is half the number of the considered classifier (majority rule).

b. Combination by belief functions; it is well known that some classification algorithms can provide an estimate of the posterior probability. For example, estimates of the post-probabilities that are provided by multilayer perceptrons, can be computed in a straightforward manner of the k-NN classifier.

# REFERENCES


1. [Adamosn 98], Performance of coating systems using numerical life prediction. J Protect, Coat & Linings, Vol. 15, No. 7, 50-62, Adamson, N, 1998.

2. [Bässman 95], *Ad Oculos(Multimedia)-Digital Image Processing: Student Version 2.0*, London: International Thomson Publishing, Hemming Bässman and Philipp W. Besslich, 1995.

3. [Bruno 99], *Image Analysis for Ornamental Stone Standard's Characterisation*, International Symposium on Imaging Applications in Geology, pp.29-32, May 6-7, 1999, R. Bruno, S. Persi paoli, P. Laurenge, M. Coluccino, F. Muge, V. Ramos, P. Pina, M. Mengucci, M. Cica Olmo, E. Serrano Olmedo, 1999.

4. [Castleman 95], *Digital Image Processing*, Prentice Hall, Kenneth R. Castleman, 1995.

5. [Chamberlain 01], *Task 2: Non-destructive testing strategies for classification of existing coated and prepared steel surfaces, Image processing rust recognition system*, City University, London, D. A. Chamberlain , V. Hlebarov and M. Trujillo, 2001.

6. [Cunningham 07], k-Nearest Neighbour Classifier, Technical Report UCD-CSI-2007-4, Padraig Cunningham and Sarah Jane Delany, 2007.

7. [Don 84], *Metal surface inspection using image processing techniques*, IEEE Trans On. SMC, vol smc-14 no1 pp 139-146, H Don, K. Fu, C.R. Liu, W.C. Lin, 1984.

8. [Dubes 92], *Performance evaluation for four class of texture features*, Pattern Recognition, vol. 25, no. 8, pp. 819-833, P.P. Ohanian and R.C. Dubes, 1992.

9. [Ďurčeková 09], *Comparison of Ten Biochemical Laboratory Tests Before And After Treatment By Statins*, Nova Biotechnologica 9-2, Tatiana Ďurčeková, Ján Mocák, Ján Balla, Gabriela Gromanová and Katarína Boronová, 2009.

10. [Ercil 94], *Automated Visual Inspection of Metallic Surfaces*, Proceedings of the Third International Conference on Automation, Robotics and Computer Vision, pp 1950-1954, Singapore, A. Ercil, 1994.

11. [Fahlman 89], *Fast learning variations on back-propagation: An empirical study*, In Proceedings of the 1988 Connectionist Models Summer School, D. Touretzky, G. Hinton and T. sejnowski, eds., San Mateo, CA:Morgan Kaufmann, pp. 38-51, S. E. Fahlman, 1989.

12. [Field 00], *Discovering Statistics Using SPSS for Windows*", SAGE Publications, Andy Field, 2000.

13. [Freeman 92], *Neural Networks – Algorithms, Applications, and Programming Techniques*, Addison-Wesley Publishing Company, James A Freeman and David M Skapurs, 1992.

14. [Fukunaga 90], *Introduction to Statistical Pattern Recognition - Second Edition*, Morgan Kaufmann Academic Press, Keinosuke fukunaga, 1990.

15. [Ghosh 91], *Nerual Networks for Texture Image Processing*, J. Ghosh, A.C. Bovik, 1991.

16. [Hagan 95], *Neural Network Design*, Brooks Cole, Martin T. Hagan, Howard B. Demuth, Mark H. Beale, Beale Hagan Demuth, 1995.

17. [Hall-Beyer 00], *GLCM Texture: A Tutorial*, Department of Geography, University of Calgary. Mryka Hall-Beyer, 2000.

18. [Hawkins 69], *Textural Properties for Pattern Recognition, In Picture Processing and Psychopictorics*, Academic Press, New York, J. K. Hawkins, B. Lipkin and A. Rosenfeld (editors), 1969.

19. [Haykin 94], *Neural Networks – A Comprehensive Foundation*, Macmillan Publishing Company, Simon Haykin, 1994.

20. [Hecht-Nielsen 90], *Neurocomputing*, Addison-Wesley, R. Hecht-Nielsen, 1990.

21. [Huberty 06], Applied MANOVA and Discriminant Analysis 2nd Edition, A John Wiley & Sons, Inc. Publication, Carl J. Huberty, 2006.

22. [Jain 90], *Texture Analysis of Automotive Finishes*, Proceeding of Vision 90, pp 8.1-8.16, Anil K. Jain, F. Farrokhnia, D.H. Alman, 1990.

23. [Jiao 93], *The Calculation of Neural Networks*, Publishing house of Xi'an University of Electronic Science and Technology, China, Jiao Licheng 1993.

24. [Kung 93], *Digital Neural Networks*, A Simon & Schuster Company, S. Y. Kung 1993.

25. [Laws 79], *Texture energy measures*, In DARPA Image Understanding Workshop, Los Angeles, CA, pages 47-51, DARPA, Los Altos, CA, K I Laws, 1979.

26. [Lepistö 03], *Rock image classification using non-homogenous textures and spectral imaging*, the 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2003, Leena Lepistö, Iivari Kunttu, Jorma Autio and Ari Visa, 2003.

27. [Li 10], Texture image retrieval using compact texton co-occurrence matrix descriptor, MIR '10 Proceedings of the international conference on Multimedia information retrieval, Qingyong Li, Zhiping Shi, 2010.

28. [Linggard 92], *Neural Networks for Vision, Speech and Natural Language*, Chapman & Hall, R Linggard., D. J. Myers, and G.Nightingale, 1992.

29. [Lu 00], *SPSS for Windows, Statistic Analysis*, Publishing House of Electronics Industry, China. Wendai Lu, 2000.

30. [Marzban 97], *Neural networks vs gussian discriminant analysis*, AI Application, Vol. 10, No.1. 49-58, C.H. Marzban, 1997.

31. [Masters 93], *Practical Neural Networks Recipes in C++*, Acdemic Press Inc. Timothy Masters, 1993.

32. [MiPa 69], *Perceptrons*, Cambridge, MA:MIT Press, M. Minsky, S. Papert, 1969.

33. [Momber 03], *Hydroblasting and Coating of Steel Structures*, Elsevier Advanced Technology, Andreas W. Momber, 2003.

34. [Mridula 11], *Utilization of grey level co-occurrence matrix and Markov random field model for segmentation of colour textured images*, ICCCS '11 Proceedings of the 2011 International Conference on Communication, Computing & Security, Mridula J, Dipti Patra, 2011.

35. [Nixon 02], *Feature Extraction and Image Processing*, Newnes. Mark S. Nixon, Alberto S. Aguado, 2002

36. [Pandya 96], *Pattern Recognition with Neural Networks in C++*, Boca Raton, Fla, CRC Press. Abhijit S Pandya and Robert B Macy, 1996.

37. [Paquis 00], *Road surface classification by thresholding using morphological pyramid*, Pattern Recognition, 2000. Proceedings15th International Conference, S. Paquis, V. Legeay and H. Konik, 2000.

38. [Reniers 08], Robust Classification and Analysis of Anatomic Surfaces Using 3D Skeletons, Eurographics Workshop on Visual Computing for Biomedicine. D. Reniers, A. Jalba and A. Telea, 2008.

39. [Pietsch, Kaiser 02], *Atmosphäre. Wasser Erdboden – Prinzipen der Korrosionsschutzauswahl*, Der Maler und Lackiermeister, Nr, 7, 18-21. S. Pietsch, W-D. Kaiser, 2002.

40. [Ripley 94], *Flexible non-linear approaches to classification*, From statistics to Neural networks. Theory and Pattern Recognition Application eds V. Cherkassky, J.H. Friedman and H. Wechsler, Springer, pp. 105-126, B.D Ripley, 1994.

41. [Ripley 96], *Pattern recognition via neural networks*, Oxford Graduate Lectures on Neural Networks. Oxford University Press, B.D. Ripley, 1996.

42. [Ripley 96(2)], *Pattern recognition and neural networks*, Cambridge University Press, B.D. Ripley, 1996.

43. [Ripley 97], *Can statistical theory help us use neural networks better*, In proceedings of "Interface 97. 29th Symposium on the Interface: Computing Science and Statistics", B.D. Ripley, 1997.

44. [Roberge 06], *Corrosion Basics: An Introduction 2nd Edition,* NACE Press Book, Pierre R. Roberge, 2006

45. [Sharma 00], *Evaluation of texture methods for image analysis*, Pattern Recognition Letters, M. Sharma, M. Markou, S. Singh, 2000.

46. [Sonka 98], *Image Processing, Analysis, and Machine Vision, Second Edition*, Brooks/Cole, Milan Sonka, Vaclav Hlavac and Roger Boyle, 1998.

47. [Stork 00], *Pattern Classification (2nd Edition)*, Wiley-Interscience, David G. Stork, Elad Yom-Tov, 2000.

48. [Theodoridis 06], *Pattern Recognition – third Edition*, Elsevier Academic Press, Sergios Theodoridis, Konstantinos Koutroumbas, 2006.

49. [Trujillo 03], *Texture analysis techniques applied to rust classification in steel surface*, Proceedings of the 5th Lasted international conference Signal and Image processing August 13-15 2003 Honolulu Hawaii USA, M Trujillo, V. Hlebarov, M. Sadki and R. Paul, 2003

50. [Tuceryan 98], *Texture Analysis*, The Handbook of Pattern Recognition and Computer Vision (2nd Edition), by C. H. Chen, L. F. Pau, P. S. P. Wang (eds.), pp. 207-248, World Scientific Publishing Co., Mihran Tuceryan, Anil K. Jain, 1998.

51. [Ünsalan 95], *Pattern Recognition Methods for Texture Analysis, Case study: Steel surface classification*, Master Thesis, Bogazici University, Cem Ünsalan, 1995.

52. [Ünsalan 98], *Classification of rust grades on steel surface*, Dept. of Electrical & Electronics Engineering, Bo azici University, Turkey, Cem Ünsalan and Aytül Ercil, 1998.

53. [Ünsalan 99], *Automated Inspection of Steel Structures*, Recent Advances in Mechatronics, Springer Verlag, pp. 468-480. Cem Ünsalan and Aytül Ercil, 1999.

54. [VanGool 85], *Texture Analysis Anno 1983*, Computer Vision Graphics and Image Processing, vol.29, pp. 336-357, L. Van Gool, P. Dewaele and A. Oosterlinck, 1985.

55. [Vemuri 90], *Artificial Neural Networks: Theoretical Concepts*, IEEE Computer Society Press. V.Vemuri, 1990.

56. [Weszka 76], *A comparative study of texture measures for terrainclassification, IEEE Transactions on Systems, man and Cybernetics*, vol. 6, pp. 269-285, J.S. Weszka, C. R. Dyer and A. Rosenfeld, 1976.

57. [Yuan 99], *Artificial Neural Networks and Applications*, Publishing House of Tsinghua University, China. Zhengren Yuan, 1999.

58. [Zacharias 10], Combining Singular Point and Co-occurrence Matrix for Fingerprint Classifcation, Compute'10 Bangalore, Karnataka, India, Geevar C Zacharisa, P Sojan Lal, 2010.