

https://dspace.org/

# MACHINE CONDITION MONITORING USING NEURAL NETWORKS: FEATURE SELECTION USING GENETIC ALGORITHM

**Hippolyte, Djonon Tsague**

**2007-02-26T12:04:37Z**
http://hdl.handle.net/10539/2127

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

# MACHINE CONDITION MONITORING USING NEURAL NETWORKS: FEATURE SELECTION USING GENETIC ALGORITHM

**Hippolyte Djonon Tsague**

A dissertation submitted to the School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg, South Africa in fulfillment of the requirements of the degree of ***Master of Science in Engineering***

Johannesburg

April 2006

# DECLARATION

I, Hippolyte Djonon Tsague, declare that the work carried out in this dissertation is my own work, unless where reference and acknowledgement is made to other authors. It has not been submitted or being submitted for any other degree.

Signed Date

…….. ……

Hippolyte Djonon Tsague

# Executive Summary

Monitoring rotating machinery is often accomplished with the aid of vibration sensors. The vibration sensor signals contain a wealth of complex information that characterizes the dynamic behavior of the machinery. Transforming this information into useful knowledge about the health of the machine can be challenging due to the presence of extraneous noise sources and variations in the vibration signal itself. An approach for bearing fault diagnosis using neural networks, genetic algorithm, higher order statistics and frequency-domain bearing vibration analysis is developed to address this problem. Identification of the most useful features is done using genetic algorithm. This step is very important for an efficient classification as opposed to using all features which leads to very high computational cost and is consequently not desirable. The project also benchmarks these abilities against the best established conventional methods for motor condition monitoring. A case study is performed on a number of healthy and faulty roller bearing to illustrate the effectiveness of the proposed techniques.

# Acknowledgements

I would like to thank the following people who made this dissertation project a possibility by an extensive support:

Prof Tshilidzi Marwala, for his valuable support (educational, financial or social), insight and guidance throughout the development of this project. I also thank him for being such an inspiring supervisor.

My fellow PhD master and students Brain Leke, Gareth Setati, Jonathan Spiller and Michael Kola for their companionship and some insights and details into my project which I may have neglected if not of their intervention.

My family (Mathurine, Charles, Sophie, Sandrine, Nathalie, Freddy) for their moral support. To Christelle Lekefouet, for all the encouragement and understanding she gave me during this period. To Mr. Nya Georges, Mr. Fonkou Desire, I salute you.

I am appreciative of the opportunity awarded to me by my parents Tsague Paul and Tsague Martine, and would like to give them special thanks. They have continually given me more than I could ever have asked for and have nothing but more than inspired me to greater things. Thank you so much.

I am grateful to the ALMIGHTY for giving me the courage and strength, peace and joy in my life.

Johannesburg
South Africa
April 2006

H. Djonon Tsague

# Table of Contents

**DECLARATION** ..... **II**

**EXECUTIVE SUMMARY** ..... **III**

**ACKNOWLEDGEMENTS** ..... **IV**

**TABLE OF CONTENTS** ..... **V**

**LIST OF FIGURES** ..... **VII**

**LIST OF TABLES** ..... **VIII**

**CHAPTER 1: INTRODUCTION AND LITERATURE REVIEW** ..... **1**

**CHAPTER 2: MACHINE CONDITION MONITORING USING NEURAL NETWORKS: FEATURE SELECTION USING GENETIC ALGORITHM** ..... **5**

2.1 INTRODUCTION ..... 5
2.2 ROLLING ELEMENT BEARING ..... 7
2.3 FAST FOURIER TRANSFORM ..... 8
2.4 HIGHER ORDER STATISTICS ..... 9
*2.4.1 Moments and Cumulants* ..... *9*
*2.4.2 Gaussian Noise Elimination* ..... *9*
2.5 NEURAL NETWORKS ..... 10
2.6 GENETIC ALGORITHMS ..... 12
*2.6.1 Genetic Algorithms for Feature Selection* ..... *13*
2.7 CONFUSION MATRIX ..... 15
2.8 PROBLEM DESCRIPTION AND METHOD ..... 17
*2.8.1 Feature Extraction* ..... *17*
*2.8.2 Spectral Features* ..... *18*
*2.8.3 Statistical Features* ..... *19*
*2.8.4 Combination of Statistical and Spectral Features* ..... *20*
2.9 TRAINING AND SIMULATION ..... 21
*2.9.1 ANN without GA* ..... *21*
*2.9.2 ANN with GA* ..... *22*
2.10 RESULTS ..... 24
*2.10.1 Training Performance and Overfitting* ..... *24*
*2.10.2 Results: ANN without GA* ..... *26*
*2.10.3 Statistical Input* ..... *26*
*2.10.4 Spectral Input* ..... *28*
*2.10.5 Combination of Input* ..... *29*
2.11 GENETIC ALGORITHM WITH ANN AFTER 220 GENERATIONS ..... 30
2.12 ANALYSIS OF RESULTS ..... 32

2.13 CONCLUSION ..... 33
**REFERENCES ..... 34**
**APPENDIX A: TABLE OF RESULTS ..... 38**
A.1 STATISTICAL TEST SET RESULTS ..... 38
A.2 SPECTRAL TEST SET RESULTS ..... 42
A.3 COMBINATION OF INPUTS TEST SET RESULTS ..... 46

# List of Figures

Figure 1: Basic rolling element bearing geometry ..... 7
Figure 2: A simple ANN ..... 11
Figure 3: High level description of a GA algorithm ..... 13
Figure 4: Crossover: (a) parents. (b) Retain common parts. (c) Cross ..... 22
Figure 5: Subset mutation ..... 22
Figure 6: Flow diagram showing all the steps in the implementation ..... 24
Figure 7: Error on training and validation sets ..... 25
Figure 8: GA fitness function convergence curve ..... 31

# List of Tables

Table 1: Moments and cumulants properties for a ..... 10
Table 2: Confusion matrix for a two class classifier ..... 15
Table 3: Binary input encoding of the 10 type of faults ..... 18
Table 4: PDF of the vibration data ..... 20
Table 5: Confusion matrix entries on the statistical test set ..... 26
Table 6: System's performance on statistical test set ..... 27
Table 7: Confusion matrix entries on the spectral test set ..... 28
Table 8: System's performance on spectral test set ..... 28
Table 9: Confusion matrix entries on the combination of input test set ..... 29
Table 10: System's performance on spectral test set ..... 30
Table 11: Comparison between stand alone ANN and GA with ..... 32
Table 12: Statistical test set result using 3 neurons ..... 38
Table 13: Statistical test set result using 5 neurons ..... 39
Table 14: Statistical test set result using 7 neurons ..... 40
Table 15: Statistical test set result using 10 neurons ..... 41
Table 16: Spectral test set result using 3 neurons ..... 42
Table 17: Spectral test set result using 5 neurons ..... 43
Table 18: Spectral test set result using 7 neurons ..... 44
Table 19: Spectral test set result using 10 neurons ..... 45
Table 20: Combination of inputs test set result using 3 neurons ..... 46
Table 21: Combination of inputs test set result using 5 neurons ..... 47
Table 22: Combination of inputs test set result using 7 neurons ..... 48
Table 23: Combination of inputs test set result using 10 neurons ..... 49

# Chapter 1: Introduction and Literature Review

Condition monitoring of machinery has increased in importance as more engineering processes are automated and the manpower required to operate and supervise plants is reduced. The monitoring of the condition of machinery can significantly reduce the cost of maintenance. Firstly, it can allow an early detection of potential catastrophic fault, which could be extremely expensive to repair. Secondly, it allows the implementation of conditions based maintenance rather than periodic or failure based maintenance [1]. In these cases, significant savings can be made by delaying schedule maintenance until convenient or necessary.

Although there are numerous efficient methods for modeling of mechanical systems, they all suffer the disadvantage that they are only valid for a particular machine. Changes within the design or the operational mode of the machine normally require a manual adaptation. Using Neural Networks to model technical systems eliminates this major disadvantage. The basis for a successful model is an adequate knowledge base on which the network is "trained". Without prior knowledge of the machines systematic behavior or its history, training of a neural Network is not possible. Therefore, it is a pre-requisite that the knowledge base contains a complete behavior of the machine covering the respective operational modes whereby, not all rather the most important modes are required. Neural networks have a proven ability in the area of nonlinear pattern classification. After being trained, they contain expert knowledge and can correctly identify the different causes of bearing vibration. The capacity of artificial neural networks to mimic and automate human expertise is what makes them ideally suited for handling nonlinear systems. Neural networks are able to learn expert knowledge by being trained using a representative set of data [2]-[6]. At the beginning of a neural network’s training session, the neural network fault detector’s diagnosis of the motor’s condition will not be accurate. An error quantity is measured and used to adjust the neural network’s internal parameters in order to produce a more accurate output. This process is repeated until a suitable error is achieved. Once the network is sufficiently trained and the parameters have been saved, the neural network contains all the necessary knowledge to perform the fault detection.

One of the most important aspects of achieving good neural network performance has proven to be the proper selection of training features. The curse of dimensionality states that, as a rule of thumb, the required cardinality of the training set for accurate training increases exponentially with the input dimension [7]. Thus feature selection which is a process of identifying those features that contribute most to the discrimination ability of the neural network is required. Proposed methods for selecting an appropriate subset of features are numerous [8]-[11]. Methods based on generating a single solution, such as the popular forward step wise approach, can fail to select features which do poorly alone but offer valuable information together. Approaches that maintain a population of solutions, such as genetic algorithms (GA) are more likely to speedily perform efficient searches in high dimensional spaces, with strong interdependencies among the features. The emphasis in using the genetic algorithm for feature selection is to reduce the computational load on the training system while still allowing near optimal results to be found relatively quickly.

To obtain accurate measure of the condition of machinery, a wide range of approaches can be employed to select features indicative of condition. By comparing these features with features for known normal and probable fault conditions, the machine's condition can be estimated. The most common approach is that of analysis in the frequency domain by applying a Fast Fourier Transform (FFT) to the time domain history data. The idea is simply to measure the energy (mean square value) of the vibrations. As the machine condition deteriorates, this measure is expected to increase. The method is able to reveal the harmonics around the fundamental frequency of the machine and other predominant frequency component (such as the cage frequency) [12]. Frequency analysis is well established and may be used to detect, diagnose and discriminate a variety of induction motor faults such as broken rotor bars, cage faults, phase imbalance, inner and outer race faults. However, as common in the monitoring of any industrial machine, background noise in recorded data can make spectra difficult to interpret. In addition, the accuracy of a spectrum is limited due to energy leakage [12- 14].

Like many of the new techniques now finding application in machinery condition monitoring, Higher Order Statistics was originally confined to the realms of non-linear structural dynamics. It has of recent however found successful application to the identification of abnormal operation of

diesel engines and helicopter gearboxes [5, 7]. Higher Order Statistics provide convenient basis for comparison of data between different measurement instances and are sufficiently robust for on-line use. They are fast in computation compared with frequency or time-domain analysis. Furthermore, they give a more robust assessment than lower orders and can be used to calculate higher order spectra. This dissertation reports work which attempts to extend this capability to induction motors.

The aim of this project is therefore to examine the use of Genetic Algorithms to select the most significant input features from a large set of possible features in machine condition monitoring contexts. The results show the effectiveness of the selected features from the acquired raw and preprocessed signals in diagnosis of machine condition. This project consists of the following tasks:

- Using Fast Fourier transform and higher order signals techniques to preprocess data samples.
- Create an intelligent engine using computational intelligence methods. The aim of this engine will be to recognize faulty bearings and assess the fault severity from sensor data.
- Train the neural network using a back propagation algorithm.
- Implement a feature selection algorithm using genetic algorithms to minimize the number of selected features and to maximize the performance of the neural network.
- Retrain the neural network with the reduced set of features from genetic algorithm and compare the two approaches.
- Investigate the effect of increasing the number of hidden nodes in the performance of the computational intelligence engine.
- Evaluate the performance of the system using confusion matrices.

The output of the design is the estimate of fault type and its severity, quantified on a scale between 0-3. Where, 0 corresponds to the absence of the specific fault and 3 the presence of a severe machine bearing fault. This research should make contribution to many sectors of industry such as electricity supply companies, and the railroad industry due to their need of techniques that are capable of accurately recognizing the development of a fault condition within a machine

system component. Quality control of electric motors is an essential part of the manufacturing process as competition increases, the need for reliable and economical quality control becomes even more pressing. To this effect, this research project will contribute in the area of faults detection in the production line of electric motor.

# Chapter 2: MACHINE CONDITION MONITORING USING NEURAL NETWORKS: FEATURE SELECTION USING GENETIC ALGORITHM

Abstract: **Artificial neural network have been widely used for health diagnosis of rotating machinery using features extracted from vibration emission signals. One of the most important considerations in applying neural networks to condition monitoring of electrical machine is the proper selection of training features. Irrelevant or noisy features unnecessarily increase the complexity of the problem and can degrade modeling performance. A Genetic algorithm for feature selection is developed, based on the concept of dominance. The algorithm is used to effectively select a smaller subset of features that together form a genetically fit family for fault identification and classification tasks. The effectiveness of the method is demonstrated on recorded ball bearing test data.**

## 2.1 Introduction

With the increase in production capabilities of modern manufacturing systems, plants are expected to run continuously for extended hours. This has fueled the need for faster and more accurate methods of machine condition monitoring. Machine condition monitoring can significantly reduce the cost of maintenance. Firstly, it can allow early detection of potential catastrophic faults which could be very expensive to repair. Secondly, it allows the implementation of condition based maintenance rather than periodic or failure based maintenance. Thirdly, it is required to detect, identify and then classify different kind of faults that can occur within a machine system. Often, vibration data are collected using accelerometers placed at both the drive and fan end of the motor housing. Vibration data are then preprocessed to extract time invariant features. In many cases, the power spectrum changes when faults occur. This is especially likely in rotating machinery where the vibrations are dominated by harmonics of the rotating frequency [13]. Using the Fast Fourier Transform can provide a fast estimate of the power spectrum. This, however, produces a large number of features and in general, only a

subset of the frequency components generated is used for classification. By treating the vibration as a random variable, other Higher Order Statistical moments, cumulants and measures such as the crest factor can also be used as features. Neural Networks have shown great promise as a means to detect faults in rotating machinery. One of the most important aspects of achieving good neural network performance has proven to be the proper selection of training features. Feature selection is a challenging and crucial stage of many empirical modeling where the solution to the problem is neither trivial, nor unique. It is often noticed that relevant features are excluded from the subset of optimal features, as other features encode the same information. In addition, a feature that carries no independent information can become of critical importance when combined with other features. This fact is overlooked when feature selection is based on correlation tests or on information measures such as mutual information between the potential predictors and the output variable. Such interdependencies can become confounding factors for many feature selection techniques. Extraction techniques such as the Principle Component Algorithm (PCA) are based on the assumption that the greater the spread of the data on a particular axis, the greater their effect on the discrimination ability of the neural network [14]. This need not be true. Furthermore, methods based on generating a single solution such as the stepwise approach can fail to select features, which do poorly alone but offer valuable information together [14]. Approaches, that maintain populations of solutions, such as Genetic Algorithms (GA), are likely to speedily perform efficient searches in high dimensional spaces, with strong interdependencies among the features [15]. GA-based techniques have also been shown to be extremely successful in evolutionary design of various classifiers, such as those based on artificial neural network [16], partial least square [17] and principal components regression [18]. The work presented herein proposes the use of multi objective genetic algorithm for feature selection. In addition, applicability of such an approach in neural network modeling is shown. The key features of the developed methodology are its computational simplicity and its effectiveness on real world problems of considerable dimensionality. A confusion matrix is used to evaluate the performance of the classifier engine. The structure of the report is as follows. The next section introduces the rolling element bearing and its structure. Sections 2.3 and 2.4 discuss the use of Fast Fourier Transform and Higher Order Statistics as preprocessing tools. In section 2.5, a description of neural network model used is given. Section 2.6 introduces multi objective algorithm approach to feature selection. In section 2.7, a classifier evaluation technique known as

confusion matrix is presented. Sections 2.8 and 2.9 present the problem description, training and simulation respectively. Results without GA optimization are presented in section 2.10. In section 2.11 ANN coupled with GA optimization results are presented, followed by the overall analysis of the results in section 2.12 and a conclusion in 2.13.

## 2.2 Rolling Element Bearing

The basic purpose of a machine bearing is to provide a near frictionless environment and to support and guide a rotating shaft. Rolling element bearings, regardless of type (ball, cylindrical, spherical, tapered, or needle) consist of an inner and an outer race separated by the rolling elements, which are usually held in a cage as shown in Figure 1. Mechanical flaws may develop on any of these components. Using the basic geometry of a bearing, the fundamental frequencies generated by these flaws can be determined [19].

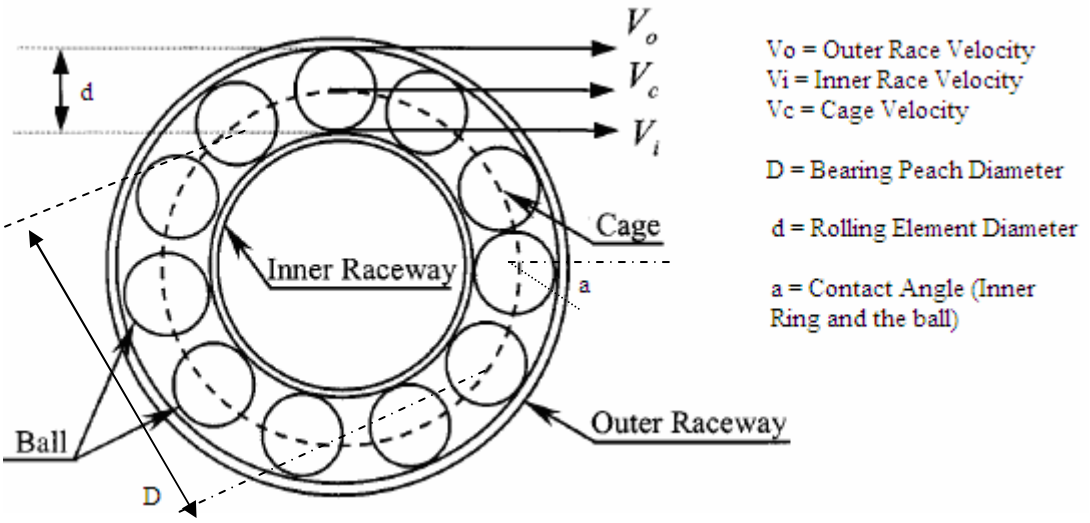



**Figure 1: Basic rolling element bearing geometry [20]**

All rolling element bearings have one thing in common: all parts must be in physical metal to metal contact at all times. Installation instructions specify the amount of bearing pre-load to maintain the component contact [19]. Two general bearing styles are utilized at this time: the journal bearing and the rolling element bearing. For lower horsepower and lighter loaded machines, the rolling element bearing is a popular choice [11, 12]. Some of the reasons why the rolling element bearings are used are: low starting friction, low operating friction, ability to

support loads at low (even zero) speed, lower sensitivity to lubrication and the ability to support both radial and axial loads in the same bearing [12]. By themselves, rolling element bearings have very little damping, so whenever a machine with rolling element bearings traverses a balance resonance, large vibration can result [11]. Also, compared to fluid film bearings which generally have a long life, rolling element bearings have a limited fatigue life due to the repeated stresses involved in their normal use. Rolling element bearings have some unique concerns not found in journal bearings. A rolling element bearing will always force a vibration node at its location. Because of the metal to metal contact, this bearing will provide very little vibration damping [19]. Although these bearings are a very precisely machined part they have a limited lifetime. Each component of the bearing will generate specific frequencies as defects initiate and become more prevalent [19].

## 2.3 Fast Fourier Transform

Since most bearing vibrations are periodical movements, it is easy to select vibration features from the frequency domain using the powerful and popular Fast Fourier Transform (FFT) technique. In many cases, the power spectrum changes when faults occur. This is especially likely in rotating machinery where the vibrations are dominated by harmonics of the rotating frequency [11]. The fast Fourier transform (FFT) is an efficient algorithm for computing the Discrete Fourier Transform (DFT) of a sequence; it is not a separate transform. It is particularly useful in areas such as signal and image processing, where its uses range from filtering, convolution, and frequency analysis to power spectrum estimation. Fourier analysis is extremely useful for data analysis, as it breaks down a signal into constituent sinusoids of different frequencies [11].

Matlab's FFT function is an effective tool for computing the discrete Fourier transform of a signal. The typical syntax for computing the FFT of a signal is FFT($x$, N) where $x$ is the signal to transform, and N is the number of points in the FFT. N must be at least as large as the number of samples in the signal [13]. For this research project, N was set to 64 i.e. a 64 point FFT of the raw data was carried and those values were used as input data for the given data sample.

## 2.4 Higher Order Statistics

Higher Order Statistics or Higher Order Spectra is a field of statistical signal processing which has become very popular in the last 15 years. It makes use of information extra to that usually used in traditional signal processing measures such as the power spectrum and autocorrelation function. This extra information can be used to get better estimates of parameters in noisy situations, or to shed light on nonlinearities in the signal's production mechanism.

### 2.4.1 Moments and Cumulants

Moments are defined as the expectations of the correspondingly powered signal values. Cumulants are the Taylor expansion coefficients around zero of a function called the second characteristic function. Both moments and cumulants are single valued. A summary of the mathematical relationship between moments ($m_i$), cumulants ($c_i$) and the commonly used parameters of mean, variance, skewness, and kurtosis is provided below in Equations 1-5. Where the $k^{th}$ *moment* ($m_k$) of a random variable *x* is calculated as follow:

$k^{th}$ Order Moment, $m_k = \frac{1}{n}\sum (x - \bar{x})^k$ **(1)**

Where k = 1 … 4.

Cumulants are calculated using Equations 2 – 5 below.

1st Order Cumulant, $c_1 = m_1$ **(2)**

2nd Order Cumulant, $c_2 = m_2 - m_1^2$ **(3)**

3rd Order Cumulant, $c_3 = m_3 - 3m_2m_1 + 2m_1^3$ **(4)**

4th Order Cumulant, $c_4 = m_4 + 4m_3m_1 - 3m_2^2 + 12m_2m_1^2 - 6m_1^4$ **(5)**

### 2.4.2 Gaussian Noise Elimination

One of the benefits in the use of Higher Order Statistics (HOS) to maintenance engineers is based on the properties of a Gaussian signal or waveform. A Gaussian signal is completely

characterized by its mean and variance. It can be shown that all higher order cumulants of a zero-center Gaussian are identically zero. This property suggests that, measurement noise, which is often assumed to be Gaussian, disappears at third and higher cumulants order (as shown in Table 1) [5]. This is one of the key motivations behind the investigation into HOS for condition monitoring of electrical machines. It is worth mentioning at this stage that, higher-order cumulants are invariant to a shift of mean; hence, it is convenient to define them under the assumption of zero mean.

**Table 1: Moments and cumulants properties for a Gaussian noise [5]**

| Order | Moment Value | Cumulant Value |
|---|---|---|
| $1^{st}$ | 0 | 0 |
| $2^{nd}$ | $\sigma^2$ | $\sigma^2$ |
| $3^{rd}$ | 0 | 0 |
| $4^{th}$ | $3\sigma^2$ | 0 |

## 2.5 Neural Networks

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. It is a machine that is designed to model the way in which the brain performs a particular task. The network architecture or topology (including: number of nodes in hidden layers, network connections, initial weight assignments, activation functions) plays a very important role in the performance of the ANN, and usually depends on the problem at hand. Figure 2 shows a simple ANN and its constituents. In most cases, setting the correct topology is a heuristic model selection. Whereas the number of input and output layer nodes is generally suggested by the dimensions of the input and the output spaces, selecting the network complexity or regularization is yet again very important. Every ANN consists of at least one hidden layer in addition to the input and the output layers. The number of hidden units governs the expressive power of the Neural Network and thus the complexity of the decision boundary. For well-separated classes fewer units are required and

for highly interspersed data more units are required. The number of synaptic weights is based on the number of hidden units. This number represents the degrees of freedom of the network. Hence, we should have fewer weights than the number of training points. Network Learning pertains to training an untrained network. Input patterns are exposed to the network and the network output is compared to the target values to calculate the error, which is corrected in the next pass by adjusting the synaptic weights. Several training algorithms have been designed; the most commonly used being the Levenberg-Marquardt (LM) back-propagation algorithm [9]. In back-propagation, the output values are compared with the correct answer to compute the value of some predefined error function. The error is then fed back to the network. This process is repeated until the network converges to some state where the error of the calculation is reasonably small. This algorithm is used throughout this study. The stopping Criterion indicates when to stop the training process. It can be a predetermined limit of absolute error, Minimum Square Error (MSE) or just the maximum number of training epochs.

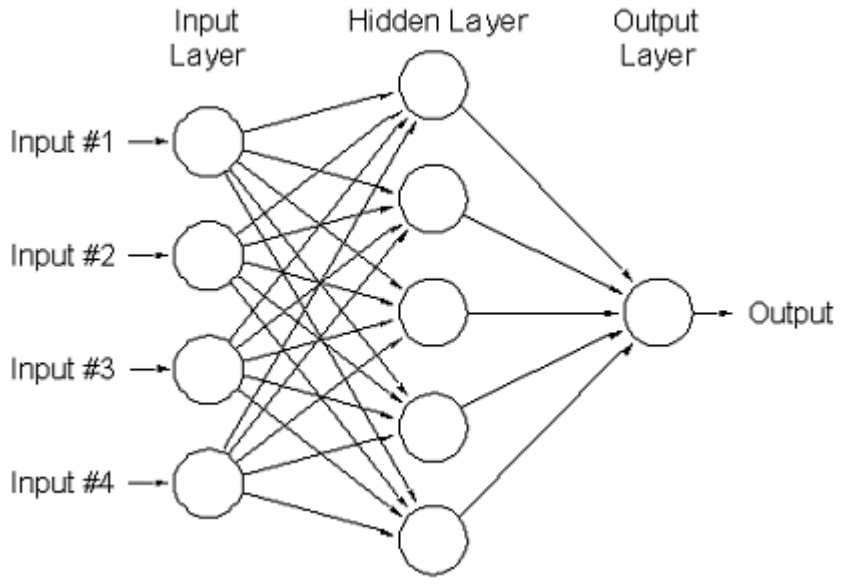



**Figure 2: A simple ANN [9]**

## 2.6 Genetic Algorithms

The GA is a stochastic global search method that mimics the metaphor of natural biological evolution [16]. GAs operate on a population of potential solutions applying the principle of survival of the fittest to produce (hopefully) better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. A Typical GA cycle and its high level description are shown in Figure 3. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation [16].

One may think that generating populations from only two parents may result in loosing the best chromosome from the last population. This is true, and so elitism is often used. This means, that at least one of a generation's best solution is copied without changes to a new population, so the best solution can survive to the succeeding generation [17].

1. **[Start]** Generate a random population of *n* chromosomes (suitable solutions for the problem)
2. **[Fitness]** Evaluate the fitness *f(x)* of each chromosome *x* in the population
3. **[New population]** Create a new population by repeating the following steps until the new population is complete
    1. **[Selection]** Select two parent chromosomes from a population according to their fitness (the better their fitness, the bigger their chance to be selected)
    2. **[Crossover]** With a crossover probability cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.
    3. **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).
    4. **[Accepting]** Place new offspring in the new population
4. **[Replace]** Use new generated population for a further run of the algorithm
5. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in the current population
6. **[Loop]** Go to step **2**

**Figure 3: High level description of a GA algorithm [15]**

### 2.6.1 Genetic Algorithms for Feature Selection

There has been widespread interest from the control community in applying Genetic Algorithm (GA) for feature selection in engineering. Feature selection is similar to an optimization problem of choosing an optimal combination of features from a large and possibly multimodal search space. Two major classes of optimization techniques have traditionally been used, namely:

calculus-based techniques, that use gradient-based search mechanisms, and enumerative techniques, such as Dynamic Programming (DP) [16]. DPs are useful when a global minimum may not be possible or very difficult to achieve. However, their computational complexities make them unsuitable for effective use in most practical cases as well as in problems where the dimension is too high for an exhaustive search. A technique called the backtrack search [17], also known as the branch-and-bound algorithm can be employed to ease the computational burden. The backtrack search guarantees the optimal solution if the criterion function satisfies the monocity condition. The monocity condition requires that the value of the criterion function be nondecreasing when additional features are added. GA developed in this project meets these requirements. The key concept here is dominance. A solution is dominant over another only if it has superior performance in all criteria. The concept of dominance can maintain population diversity to allow the algorithm to discover a range of features with different performance versus dimensionality trade-offs. In some situations, the dimension of the input vector is large, but the components of the vectors are highly correlated (redundant). It is useful in this situation to reduce the dimension of the input vectors. An effective procedure for performing this operation is Principal Component Analysis (PCA) [17]. This technique has three effects: it orthogonalizes the components of the input vectors (so that they are uncorrelated with each other); it orders the resulting orthogonal components (principal components) so that those with the largest variation come first; and it eliminates those components that contribute the least to the variation in the data set. It is the most widely used technique among similar techniques such as Multi Dimensional Scaling (MDS) and Singular Value Decomposition (SVD) [17]. PCA can be much less computationally expensive than a GA-based approach. However, all features need to be computed for PCA before a rotated feature space can be created for easier use. Therefore, using PCA still requires computation of all features demanding a large amount of data processing. GA facilitates a better scenario, in which although the computational cost will be very high during the offline training and feature selection phase, much less computing is required for online classification. Other methods for feature selection include Forward Sequential selection method. The forward sequential method starts by choosing the best individual features. Then the feature set is built from the ground up, by repeatedly adding the next feature that works best with the previously chosen features [21]. This algorithm requires a large number of evaluations and it chooses the candidates based on their respective individual performances. The major drawback

of this technique is that, in some cases, a feature that carries no independent information may be excluded from the subset of optimal features, but can become of critical importance when combined together with other features. Compared to traditional search and optimization procedures, mentioned earlier, the GA is robust, global and generally more straightforward to apply in situations where there is little or no a priori knowledge about the process to be controlled. As the GA does not require derivative information or a formal initial estimate of the solution region and because of the stochastic nature of the search mechanism, the GA is capable of searching the entire solution space with more likelihood of finding the global optimum. The GA approach for feature selection proposed in this wok addresses simultaneously but also independently, two optimization issues i.e. minimization of the number of selected features and maximization of the achieved performance.

## 2.7 Confusion Matrix

Confusion matrixes have long been used to evaluate classifier performance in many fields. A confusion matrix contains information about the actual and predicted classifications done by a classification system. It provides information on the tradeoff between the hit rate (true positives) and the false alarms (false positive) [31]. Table 2 below shows the confusion matrix for a two class classifier.

**Table 2: Confusion matrix for a two class classifier**

| | | **Predicted** | |
|---|---|---|---|
| | | Negative | Positive |
| Actual | Negative | **a** | **b** |
| | Positive | **c** | **d** |

In this project, the entries in the confusion matrix have the following meaning:

- **a** is the number of correct predictions that the bearing is healthy.
- **b** is the number of incorrect predictions that the bearing is faulty.
- **c** is the number of incorrect predictions that the bearing is healthy.
- **d** is the number of correct predictions that the bearing is faulty.

Several terms and abbreviations have been used by statisticians for a two class matrix:

- The accuracy (AC) is the proportion of the total number of predictions that were correct [31]. It is determined using Equation 6.

$$AC = \frac{a+d}{a+b+c+d} \quad \textbf{(6)}$$

- The sensitivity or True Positive (TP) fraction is the proportion of cases picked out by the classifier i.e. the proportion of positive cases that were correctly identified [31]. It is given by Equation 7.

$$TP = \frac{d}{c+d} \quad \textbf{(7)}$$

- The False Positive (FP) rate is the proportion of negative cases that were incorrectly classified, as calculated using Equation 8 [31].

$$FP = \frac{b}{a+b} \quad \textbf{(8)}$$

- The True Negative (TN) rate is defined as the proportion of negative cases that were classified correctly as given by Equation 9 [31].

$$TN = \frac{a}{a+b} \quad \textbf{(9)}$$

- The False Negative (FN) rate is the proportion of positive cases that were incorrectly classified as negative [31]. It is given by Equation 10.

$$FN = \frac{c}{c+d} \quad \textbf{(10)}$$

Provost *et.al.*[30] showed that for evaluation and comparison of different classifier, confusion matrix is better than classification accuracy. Confusion matrix was extensively used in this project to evaluate the performance of classifiers.

## 2.8 Problem Description and Method

A simpler problem of a roller bearing health monitoring has been used to illustrate the effectiveness of Genetic Algorithms in feature selection for fault classification using ANNs. Several bearings with different outer race, inner race and ball defects were used in the test setup. The experiment described below was conducted at the **Case Western Reserve University Bearing Data Center** [26]. Motor bearings were seeded with faults using electro-discharge machining (EDM). EDM is a process that uses a fine, accurately controlled electrical spark to erode metal. The cutting electrode is made of carbon or a wire feed system [32]. Faults ranging from 0.007 inches in diameter to 0.021 inches in diameter (on increment of 0.007 inches) were introduced separately at the inner raceway, rolling element (i.e. the ball) and outer raceway. Faulty bearings were reinstalled into the test motor and vibration data was recorded for motor load of 2 horsepower. Two sensors (accelerometers) attached to the housing with magnetic bases were employed to get the signal from all faulty bearings plus one bearing without any defect. Accelerometers were placed at the 12 o'clock position at both the fan and the drive end of the motor housing. A total of 10 samples (9 faults with varying groove size to simulate growing crack and 1 without any defect) of ball bearing, inner race and outer race vibration data at different severity levels were collected using a 16 channel DAT recorder. Digital data was collected at 48000 samples per second at 50 kHz to yield $2^{18}$ samples points per experiments depending on how long the experiment was [22].

### 2.8.1 Feature Extraction

Of the 10 type of faults obtained earlier, 9 faults are categorized into three risks categories low (1), medium (2), high (3) and the healthy bearing formed category 4. Data per experiment ( $2^{18}$ samples points) was divided into 128 segments of 2048 samples each. The 128 segments were then divided into three independent subsets for training, validation and testing respectively (as described in section 2.9.1). Up to 124 features (as described in the next sections) were calculated for each segment from all experiments. In constructing the neural network model, a binary representation was used to code faults type and severity levels in a 5 word binary format

as shown in Table 3. This binary encoding scheme was used because it allows for faster training [22].

**Table 3: Binary input encoding of the 10 type of faults**

| Fault Description | Binary Code |
|---|---|
| Normal bearing (No defect) | 00000 |
| Inner Race Fault (low) | 10001 |
| Inner Race Fault(medium) | 10010 |
| Inner Race Fault (high) | 10011 |
| Outer Race Fault(low) | 11001 |
| Outer Race Fault (medium) | 11010 |
| Outer Race Fault (High) | 11011 |
| Rolling Element (low) | 11101 |
| Rolling Element (medium) | 11110 |
| Rolling Element (high) | 11111 |

Because the output of the neural network had to be one of the values shown in Table 3 depending on the type of fault and severity level under consideration, any other outcome not shown in the table was taken as an incorrect classification. Using the measured signal, three dataset were prepared: the spectral feature set, the statistical feature set and the combination of both the spectral and the statistical feature sets.

## 2.8.2 Spectral Features

A spectrally based feature set was created. For each of the two channels sampled, a 64 point FFT was carried out. Based on visual inspection, it was observed that the frequency component were very small in magnitude (very close to 0) beyond the first 32 values. Consequently, only 32 values per channel were retained, i.e. a total of 64 values from the combined two channels. These were then stored as a column vector of 64 values used as input features for the given sample.

### 2.8.3 Statistical Features

The statistical features dataset contained cumulants of up to order 4. An approach suggested in [27] was adapted to create a variety of features in a systematic manner. As stated in section 2.7 above, two accelerometers were used to measure the individual vibration components in two directions *x* and *y*. In order to compute the combined effect, another signal *z* was generated using values from the two accelerometers as shown in Equation 11.

$$z = \sqrt{(x^2 + y^2)} \tag{11}$$

Cumulants, as described in Equations 1-5 were computed for all *x, y* and *z* and a 60 element matrix feature vector was formed for each data sample as described by Equation 12. The 60 elements contained in Equation 12 were used as statistical input feature to the neural network.

$$\begin{aligned}
k = [&C_{x_1} C_{x_2} C_{x_3} C_{x_4} C_{y_1} C_{y_2} C_{y_3} C_{y_4} C_{z_1} C_{z_2} C_{z_3} C_{z_4} \\
&C_{x_1} * C_{y_1} C_{x_1} * C_{y_2} C_{x_1} * C_{y_3} C_{x_1} * C_{y_4} \\
&C_{x_1} * C_{z_1} C_{x_1} * C_{z_2} C_{x_1} * C_{z_3} C_{x_1} * C_{z_4} \\
&C_{x_2} * C_{y_1} C_{x_2} * C_{y_2} C_{x_2} * C_{y_3} C_{x_2} * C_{y_4} \\
&C_{x_2} * C_{z_1} C_{x_2} * C_{z_2} C_{x_2} * C_{z_3} C_{x_2} * C_{z_4} \\
&C_{x_3} * C_{y_1} C_{x_3} * C_{y_2} C_{x_3} * C_{y_3} C_{x_3} * C_{y_4} \\
&C_{x_3} * C_{z_1} C_{x_3} * C_{z_2} C_{x_3} * C_{z_3} C_{x_3} * C_{z_4} \\
&C_{x_4} * C_{y_1} C_{x_4} * C_{y_1} C_{x_4} * C_{y_3} C_{x_4} * C_{y_4} \\
&C_{x_4} * C_{z_1} C_{x_4} * C_{z_2} C_{x_4} * C_{z_3} C_{x_4} * C_{z_4} \\
&C_{y_1} * C_{z_1} C_{y_1} * C_{z_2} C_{y_1} * C_{z_3} C_{y_1} * C_{z_4} \\
&C_{y_2} * C_{z_1} C_{y_2} * C_{z_2} C_{y_2} * C_{z_3} C_{y_2} * C_{z_4} \\
&C_{y_3} * C_{z_1} C_{y_3} * C_{z_2} C_{y_3} * C_{z_3} C_{y_3} * C_{z_4} \\
&C_{y_4} * C_{z_1} C_{y_4} * C_{z_2} C_{y_4} * C_{z_3} C_{y_4} * C_{z_4} ]
\end{aligned} \tag{12}$$

## 2.8.4 Combination of Statistical and Spectral Features

As a further preprocessing on raw data, combining statistical and spectral dataset produced a larger test set of 124 features (64 spectral features and 60 statistical features). The raw data was normalized using Equation 13 prior to the feature calculation. It has been shown that the normalized data performs much better in terms of training time and training success [9]**.** This helps in fast and uniform learning of all categories and results in small training errors [25].

$$x_i = \frac{(x_i - \mu_x)}{\sigma_x} \quad \textbf{(13)}$$

Where, $\sigma_x$ is the variance and, $\mu_x$ is the mean of the sequence *x*.

Of course, Equation 13 is applicable under the assumption that, the vibration data has a Gaussian distribution. A Gaussian signal is completely characterized by its mean and standard deviation. The mean of a distribution is, in common parlance the average value. The standard deviation is a measure of the spread of the probability distribution around the mean. A small standard deviation means the distribution are close to the mean. A large value indicates a wide range of possible outcomes [23]. An estimate of the PDF (Probability Density Function) was carried out and the results are shown in Table 4.

**Table 4: PDF of the vibration data**

| Signal | Mean | Std Deviation | RMS difference |
|---|---|---|---|
| Probability Density | 0.23600 | 0.32130 | 0.00375 |

In order to compare the measured vibration signal with the theoretical Gaussian curve (in this case a zero mean and unity standard deviation curve), the RMS (Root Mean Square) value of the difference curve was calculated using the PROB module in DATS for Windows [24]. This is a single value which quantifies the difference. In this case, the difference is very small which suggests that the vibration data had a Gaussian distribution.

## 2.9 Training and Simulation

### 2.9.1 ANN without GA

For the training of this neural network, the Batch Training Algorithm was used. To start the primary training, the first set of input and target vectors are presented to the network [29]. During the training process the weightings within the network are arranged to minimize the error between the real output and the calculated value, thereby obtaining a better simulated output. After a training period that provides satisfying results with regard to a specified number of training epochs, the training is stopped. Through this procedure, trained Neural Network is therefore able to simulate (predict) the vibration overall level of the bearing of interest by providing various process parameters as input vectors.

Training was carried out using three data sets. One feature set comprised all the statistically based features (60 features). The second feature set comprised all the spectral features (64 features). These two dataset were combined to form a larger input feature set of 124 inputs.

To cater for "catastrophic forgetting" for which Neural Networks are prone (i.e. training the network to do one task, then train it to do the second and then find out it has forgotten how to do the first task), all the 10 type of input (as shown in Table 3) were used to train the network at the same time.

The neural network was trained using each feature set separately. A fully connected two layered MLP architecture was used in the experiment. Each neuron in one layer is directly connected to the neurons of the subsequent layer. A NETLAB [29] toolbox that runs in MATLAB was used to implement the MLP neural network. A two-layered MLP architecture was used because it resulted in better results and due to the universal approximation theorem, which states that a two layered architecture is adequate for MLP [29]. A linear activation function was selected. The optimization technique used for training this architecture was the Scaled Conjugate Gradient (SCG) method. The method is a batch-mode optimization routine with the advantage that, there is no need to specify the learning rate; the algorithm uses line searches instead [30]. Furthermore, SCG method was found to solve optimization problems encountered when training an MLP network more efficiently than the gradient descent and conjugate gradient methods [30]. In order

to describe the accuracy of the model and to test the ability of the network to generalize with unknown data, the network is forced to simulate the vibration by presenting process values that have not been used before during the training.

### 2.9.2 ANN with GA

For this problem, a binary chromosomal representation was used. The design problem is to search for optimal subset of inputs. Among the potential predictor variables, there is considerable information redundancy and strong interdependencies [18]. The genetic operators used should guarantee that only valid models are produced, preferably without redundancy. Redundancy will exist when there are several different representations of the same model structure. A variation on the recombination operator proposed by Lucasius and Kateman [18] is used here. Offspring subsets are first produced from parents, preserving their intersection. The remaining elements are then shuffled and exchanged to produce two offspring. In this way, the order in which terms appear in parents is ignored and redundancy is eliminated as shown in Figure 4.

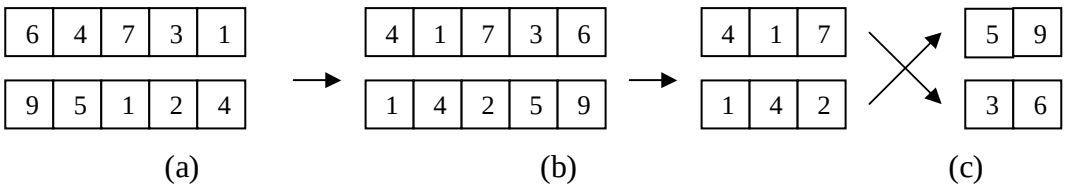



**Figure 4: Crossover: (a) parents. (b) Retain common parts. (c) Cross**

The mutation operator simply replaces terms in individuals with another term from the complimentary subset as shown in Figure 5. Thus, any mutated individual can be guaranteed to be valid.

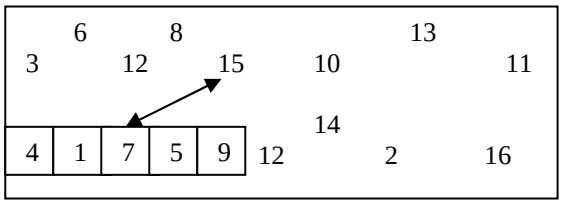



**Figure 5: Subset mutation**

The following GA settings were applied:

*Population size:* Several experiments were carried out in order to find an appropriate population size using trial and error. It was found that, a population of size 20 was appropriate for convergence in reasonable time and was kept fixed throughout the experiments.

*Mutation rate*: As with all current machine learning problems it is worth tuning the parameters such as mutation probability to find reasonable settings for the problem class you are working on. A very small mutation rate may lead to premature convergence of the genetic algorithm in a local optimum. A mutation rate that is too high may lead to loss of good solutions. There are theoretical but not yet practical upper and lower bounds for these parameters that can help guide selection. To this effect, a multi point mutation based on a specific probability was implemented and a mutation rate of 0.03 was used in all experiments.

*Crossover:* A two-point crossover was implemented and the crossover rate was set to 0.5. All members of a population were made to crossover after they were paired based on their fitness. The chromosomes were then sorted in order of decreasing fitness and pairs were formed with immediate neighbors; i.e. the best chromosomes were paired with the next best and the third best with the next one on the list and so on.

*Elitism* was implemented to retain the best chromosomes. This means, that the generation's best solution was copied without changes to a new population, so the best solution can survive to the succeeding generation.

*Fitness function:* The number of correct classification by the ANN (using the combination of statistical and spectral dataset) was used as the fitness function to capture the system's performance.

*Number of generation*: To limit the computation time to reasonable extents, the number of generation was set to 220. Figure 6 shows the flow diagram and a summary of the steps followed during the implementation of this stage. The Genetic Algorithm toolbox used is the open source GAOT [19].

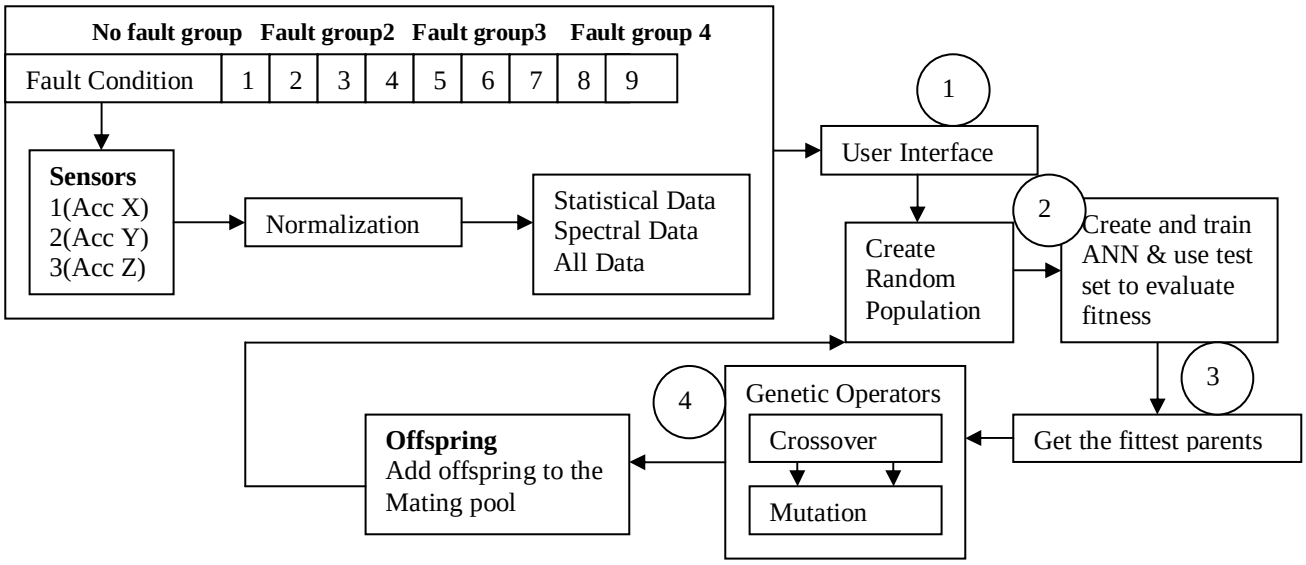



**Figure 6: Flow diagram showing all the steps in the implementation**

# 2.10 Results

## 2.10.1 Training Performance and Overfitting

The critical issue in developing this neural network was generalization i.e. how well will the network make predictions for cases that are not in the training set? ANNs, like other flexible nonlinear estimation methods such as kernel regression, can suffer from underfitting or overfitting. Overfitting is especially dangerous because it can easily lead to predictions that are far beyond the range of the training data with many of the common types of ANNs [3]. Overfitting can also produce wild predictions in multilayer perceptrons even with noise-free data. This is because of the usually large number of synaptic weight in these networks. This means that the back propagation algorithm might train the network "too perfectly", so that the weights are even tuned to fit idiosyncrasies or noise of the training pattern that are not representative for their respective classes [4]. This results in decreased generalization of the network. In order to avoid overfitting, the network should not be trained for too many epochs as overfitting usually occurs during later iterations. To cater for overfitting, a technique known as **early stopping** was implemented and is explained below.

Dataset were divided up into training, validation and test subsets. Half of the data was reserved for training and the other half for validation and testing. As an initial guess, seven neurons were used in the hidden layer. The network was trained using only the training data. Every now and then, however, training was stopped, and the network performance tested on the independent validation subset. It is important to note that no weight updates were made during this test. Learning curves showing error on the training and validation sets are shown in Figure 7.

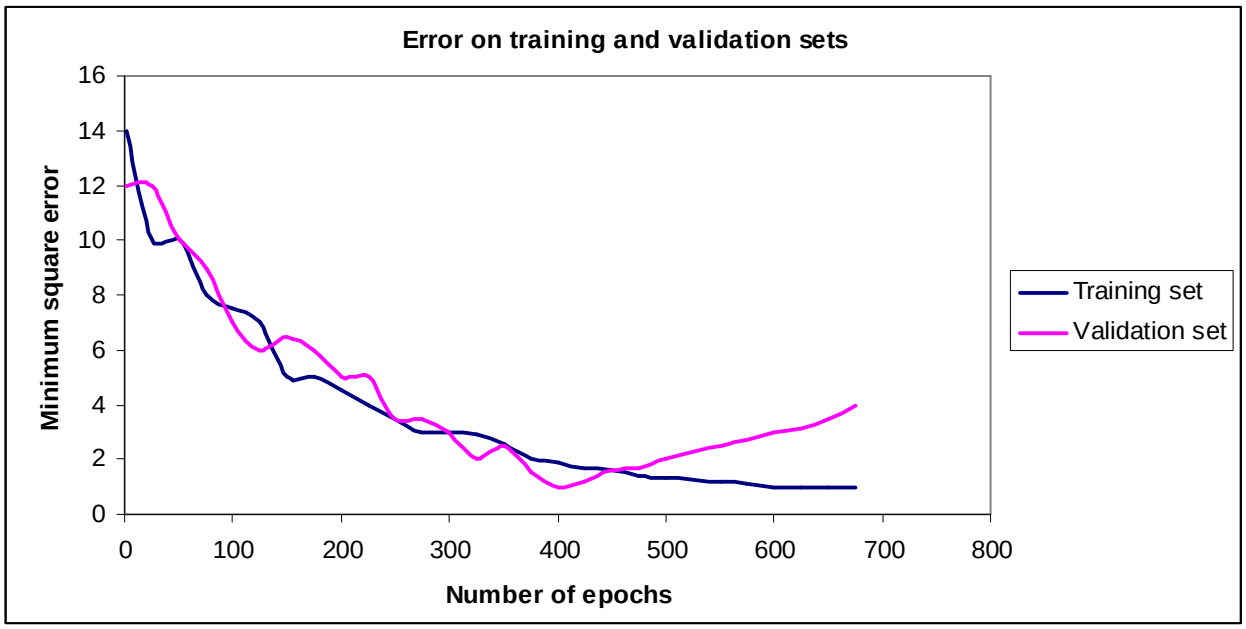



**Figure 7: Error on training and validation sets**

It is interesting to note, from Figure 7, that the training and validation sets have similar characteristics. However, performance on the validation set will stop improving, and will typically get worse after 400 epochs of training. At that stage, the network stops learning things which are expected to be true of any data sample. To avoid overfitting, training was simply stopped after 400 epochs, where performance on the validation set is optimal.

With the maximal number of iterations at hand, the next step was to test the trained network on a set of independent data to measure its ability to generalize; this was done using the third set (test set) and the results are presented in the following sections.

## 2.10.2 Results: ANN without GA

After being trained using the training set (as explained in section 2.9.1), the classifier was tested on test sets. Ten types of faults were investigated. For each type, 330 data were available for testing. The test sets were presented to the network and the number of correct and incorrect classification recorded. The output of the neural network was expected to be one of the faults type in binary format shown in Table 3. Tables 5, 6, 7, 8, 9 and 10 show the results obtained with a simple neural networks trained with the statistical training set (i), the spectral training set (ii) and the combination of statistical and spectral training sets (iii), respectively. These are average performances as each experiment has been performed at least 30 times in order to achieve statistically reliable results. The results shown in these tables are a summary of all 10 types of faults combined i.e. there are 3300 cases of which 330 are negative cases (healthy bearing). For a detailed representation of individual cases please, refer to appendix A. In the tables to follow, Neg. means negative while Pos. means positive.

## 2.10.3 Statistical Input

Table 5 shows the confusion matrix entries on the statistical input set.

**Table 5: Confusion matrix entries on the statistical test set**

| | | **3 Neurons** | | **5 Neurons** | | **7 Neurons** | | **10 Neurons** | |
|---|---|---|---|---|---|---|---|---|---|
| | | Predicted | | Predicted | | Predicted | | Predicted | |
| | | Neg. | Pos. | Neg. | Pos. | Neg. | Pos. | Neg. | Pos. |
| Actual | Neg. | 215 | 115 | 285 | 45 | 301 | 29 | 307 | 23 |
| | Pos. | 2573 | 397 | 2195 | 775 | 1547 | 1423 | 1042 | 1928 |

From Table 5, it is clear that the performance of the single classifier with 3 neurons in the hidden layer produces poor results; as out of the 330 healthy bearing only 215 cases are correctly identified. Likewise, out of 2970 (the product of 330 by 9) faulty cases, only 397 are correctly identified by the system. This poor performance results in a high number of misclassified faults

and false alarms. With 5 neurons in the hidden layer, there is a slight improvement but the outcome remains poor. With 7 and 10 neurons in the hidden layer, the outcome is much better as out of 330 healthy cases, 301 and 307 respectively are correctly identified. Similarly, out of 2970 faulty bearings, 1423 and 1928 respectively are correctly identified. The numbers of faults incorrectly classified however still remain high. From the confusion matrix shown in Table 5, the overall performance of the system was computed using Equations 5 – 10. The results are shown in Table 6.

**Table 6: System's performance on statistical test set**

| Number of neurons | Accuracy (%) | True Positive (TP) | False Positive (FP) | True Negative (TN) | False Negative (FN) |
|---|---|---|---|---|---|
| 3 Neurons | 18.55 | 0.134 | 0.350 | 0.651 | 0.867 |
| 5 Neurons | 32.12 | 0.261 | 0.136 | 0.863 | 0.739 |
| 7 Neurons | 52.24 | 0.479 | 0.089 | 0.912 | 0.516 |
| 10 Neurons | 67.73 | 0.863 | 0.069 | 0.930 | 0.350 |

Recall that the accuracy of the system was defined as the number of predictions that were correct. The accuracy in this case varies between 18.55% and 67.73% with 3 and 10 neurons, respectively, in the hidden layer. TP rate represents the proportion of faulty bearings correctly identified by the system. The closer this value is to 1 the better the performance of the classifier. In this case, the value varies between 0.134 when 3 neurons are used in the hidden layer to 0.863 with 10 neurons in the hidden layer. The FP rate was defined as the proportion of healthy bearing incorrectly classified it can be seen that the value move from 0.350 with 3 neurons in the hidden layer to about 0.069 with 10 neurons in the hidden layer. Let’s note that the closer this value is to 0, the better the performance of the classifier at identifying healthy bearings. The ability of the system at detecting healthy bearing is further confirmed by the TN rate which is the proportion of healthy bearing correctly identified. From Table 5, the TN rate moves from 0.651 with 3 neurons in the hidden layer to 0.912 with 7 neurons in the hidden layer and finally to 0.930 when 10 neurons are used in the hidden layer. The FN rate, which is the proportion of faulty bearings misclassified, is very high. This is the main area of concern in this case.

### 2.10.4 Spectral Input

The single classifiers that used the spectral input vectors generally performed better as shown in Table 7.

**Table 7: Confusion matrix entries on the spectral test set**

| | | 3 Neurons | | 5 Neurons | | 7 Neurons | | 10 Neurons | |
|---|---|---|---|---|---|---|---|---|---|
| | | Predicted | | Predicted | | Predicted | | Predicted | |
| | | Neg. | Pos. | Neg. | Pos. | Neg. | Pos. | Neg. | Pos. |
| Actual | Neg. | 280 | 50 | 293 | 37 | 317 | 13 | 324 | 6 |
| | Pos. | 1850 | 1120 | 1162 | 1808 | 487 | 2483 | 187 | 2783 |

Out of 330 healthy bearings, up to 280 are correctly classified with only 3 neurons in the hidden layer. With 5 neurons, this number goes up to 293. With 7 and 10 neurons in the hidden layer, 317 and 324 healthy bearings respectively are correctly identified. Similarly, out of 2970 faulty bearings, 1120 are correctly identified with 3 neurons in the hidden layer; with 5 neurons in the hidden layer, 1808 faulty bearings are correctly identified. This performance gets better when 7 and 10 neurons are used as 2483 and 2783 faulty bearings, respectively, are correctly identified. The general performance of the system is shown in Table 8.

**Table 8: System's performance on spectral test set**

| Number of neurons | Accuracy (%) | True Positive (TP) | False Positive (FP) | True Negative (TN) | False Negative (FN) |
|---|---|---|---|---|---|
| 3 Neurons | 42.42 | 0.378 | 0.151 | 0.849 | 0.622 |
| 5 Neurons | 63.64 | 0.608 | 0.112 | 0.888 | 0.391 |
| 7 Neurons | 84.84 | 0.837 | 0.039 | 0.960 | 0.164 |
| 10 Neurons | 94.09 | 0.937 | 0.018 | 0.981 | 0.063 |

In this case, the accuracy of the system is good; moving from 42.42 % with 3 neurons in the hidden layer to about 94.09 % with 10 neurons. The TP rate as well is very good getting closer to

1 as the number of neurons in the hidden node is increased to 10. Because of the system's ability to accurately identify both faulty and healthy bearings, the false alarm rate has gone down considerably as shown by both FP and FN. Recall that the closer these two values to 0, the better the performance of the classifier.

### 2.10.5 Combination of Input

In this case, statistical and spectral features are put together and the behavior of the system at recognizing faults is investigated. The results are shown in Tables 9 and 10.

**Table 9: Confusion matrix entries on the combination of input test set**

| | | **3 Neurons** | | **5 Neurons** | | **7 Neurons** | | **10 Neurons** | |
|---|---|---|---|---|---|---|---|---|---|
| | | Predicted | | Predicted | | Predicted | | Predicted | |
| | | Neg. | Pos. | Neg. | Pos. | Neg. | Pos. | Neg. | Pos. |
| Actual | Neg. | 288 | 42 | 312 | 18 | 321 | 9 | 325 | 5 |
| | Pos. | 1558 | 1412 | 682 | 2288 | 276 | 2694 | 85 | 2885 |

Out of 330 healthy bearings, up to 288 are correctly classified with only 3 neurons in the hidden layer. With 5 neurons in the hidden layer, 312 healthy bearings are correctly identified. With 7 and 10 neurons in the hidden layer, 321 and 325 healthy bearings, respectively, are correctly identified. Similarly, out of 2970 faulty bearings, 1412 (almost half of the total number of faulty bearings) are correctly identified with only 3 neurons in the hidden layer; with 5 neurons in the hidden layer, 2288 faulty bearings are correctly identified. This performance gets better when 7 and 10 neurons are used as 2694 and 2885 faulty bearings, respectively, are correctly identified. The general performance of the system is shown in Table 10.

**Table 10: System's performance on spectral test set**

| Number of neurons | Accuracy (%) | True Positive (TP) | False Positive (FP) | True Negative (TN) | False Negative (FN) |
|---|---|---|---|---|---|
| 3 Neurons | 51.51 | 0.475 | 0.127 | 0.872 | 0.525 |
| 5 Neurons | 78.79 | 0.770 | 0.054 | 0.945 | 0.229 |
| 7 Neurons | 91.36 | 0.907 | 0.027 | 0.972 | 0.092 |
| 10 Neurons | 97.27 | 0.971 | 0.015 | 0.985 | 0.028 |

Clearly, this bigger input vector size performed better than both the statistical and the spectral input datasets. Here, the accuracy of the system moves rapidly from 51.51% to 97.27% when 3 and 10 neurons are used in the hidden layer, respectively. Similarly, the TP and the FN rates which determine the ability of the system at classifying faulty and healthy bearings respectively are very good. Consequently, the false alarm rates given by both FP and FN are low.

## 2.11 Genetic Algorithm with ANN after 220 Generations

Although the combination of input dataset produces the best results so far, the major drawback with these bigger networks is that they need about 70% to 80% more of the time the statistical or spectral networks needed to be trained. To cater for this weakness, a GA based algorithms was developed to select the best inputs to the neural network. The goal of this exercise was to determine whether optimizing neural network architecture using genetic programming as a machine learning strategy would improve the ability of neural networks to model, detect and accurately classify bearing faults in a reasonable time span.

A simple GA was used with a population of 20 individuals, starting with randomly generated genomes and run for 220 generations. The number of features required in the solution set for input to the neural network was varied between 1 and 30. This number was finally set to 10 as it produced the best results in a reasonable time span. However, a GA-based search combined with additional constraints and cost functions for fitness evaluation could have been used to dynamically find an optimal value for the number of features to use rather than pre-specifying it

explicitly. To evaluate the performance in terms of training efficiency, the error rate criterion was used. The error rate represents the number of correctly classified faults to the total number of faults. The number of hidden nodes was kept fixed at 10 nodes; the reason being that it was shown that in most cases, the best performance without GA optimization was achieved with 10 hidden nodes. The convergence of the optimization is shown in Figure 8. These results are obtained from 20 independent runs and show the GA performance at each generation. From Figure 8, it can be seen that the GA consistently converges towards a minimum.

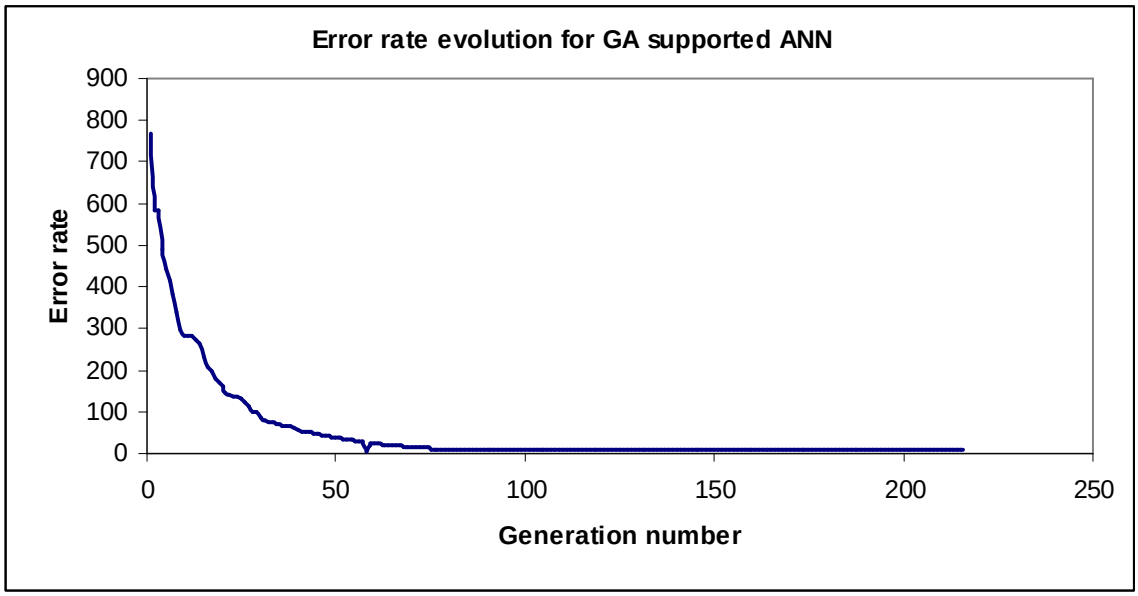



**Figure 8: GA fitness function convergence curve**

Table 11 compares performance of the system with and without GA optimization.

From the Table, it is clear that all GA supported ANN have performance in excess of 96 % with the combination of feature set producing the best performance at 100%; indicating accurate classification. This performance is achieved using only 10 inputs out of a possible 124. One point worth mentioning in this particular problem is that, of the 10 features selected by the GA, around 6 were spectral features and the remaining 4 features were either entirely selected from the statistical set or from both sets. The GA algorithm was run multiple times and it was found that, in about 80% of cases, selected features were the same for that particular dataset. In all

cases, GA supported ANNs clearly outperformed the stand alone ANNs. The time taken to train GA supported ANN was reduced by a factor of 10.

Using 10 neurons in the hidden layer, a relatively small network has been created that fulfils the criteria set earlier on. A network of this size would be ideal for a real time implementation on a microcontroller.

**Table 11: Comparison between stand alone ANN and GA with ANN after 220 generation for all data sets**

| | **Best ANN without GA** | | | **GA with best ANN** | | |
|---|---|---|---|---|---|---|
| Data Set | No. of Inputs | No. of Hidden | Perf. % | No. of Inputs | No. of Hidden | Perf. % |
| Statistical Features | 60 | 10 | 67.73 | 10 | 10 | 96.4 |
| Spectral Features | 64 | 10 | 94.09 | 10 | 10 | 99.5 |
| All Features | 124 | 10 | 97.27 | 10 | 10 | 100 |

## 2.12 Analysis of Results

A direct comparison of the statistical methods and spectral analysis is difficult because the two approaches are very different. However, as can be seen from Tables 5 and 7, the spectral content of the data is ideally suited to recognizing several of the periodic type of faults that are generated by the different conditions. This feature set clearly outperformed the dataset made of statistical features. The low performance observed on the statistical methods may be explained by the presence of too many irrelevant features in the dataset that not only degrade the modeling performance of the neural network, but also unnecessarily increase the complexity of the problem. It was observed during preprocessing that this dataset contained too many null entries due to very small or very low-frequency content change in the signal and as a result, the statistical signal did not show any significant differences with time.

The feature set containing all the data had the highest training time compared with any of the other sets. However its classification performance is the best. This is a direct result of the fact that some features from the statistical features set also work very well; in conjunction with good spectral features, they gave the best performance.

The excellent performance of GA supported ANN is due to the fact that the GA only extracts the significant (non zero) features to train the ANN and thereby results in significantly better performance and faster training time.

The hidden layers are responsible for the properties of the artificial neural network. Especially the ability to generalize depends on the number of neurons in the hidden layer. It was shown that if this number is too small, the network is not able to learn. If it is too large, the ability to generalize is lost. For this project, the number of neurons for good performance was found to be around 10 in all cases

## 2.13 Conclusion

This paper presented an approach for motor rolling bearing fault diagnosis using neural networks, genetic algorithms, higher order statistics and frequency-domain bearing vibration analysis. The capability of spectral and statistical measures to detect induction motor faults, to distinguish between them and assess their severity was presented. It was shown that Neural Networks together with Fast Fourier Transforms and statistical analysis are suited for condition monitoring of electrical machinery. One major drawback of the use of single valued statistical and spectral measures is that they produce a large number of features and in general only a subset of those components generated are used for classification. A genetic algorithm approach based on the concept of dominance was developed for optimal feature selection. It is shown that GA is well suited to feature selection as it can produce a diverse set of solutions with differing performance versus complexity trade-off characteristics in a single population. The methodology was tested with a large number of inputs and very good results were achieved. It was demonstrated that genetic algorithm is capable of selecting a subset of 10 inputs from a set of 124 features that allow the artificial neural network to perform with 100% accuracy.

# References

[1] Peter J. Tavern and James Penman, "*Condition Monitoring of Electrical Machines*" John Wiley and Sons, Letchworth: Research Studies Press, 1987

[2] P. L. Timar, A. Fazekas, J. Kiss, A. Miklos and S. J. Yang, "*Noise and Vibration of Electrical Machines*" Elsevier Science Publishers 1989

[3] A. C. Mc Cormick and A. K. Nandy, "*Classification of Rotating Machine Condition Using Artificial Neural Networks*", Proc. Of the Institute of Mechanical Engineers, Part C, vol.11, no. 6, pp 439-450, 2003.

[4] M. Y. Chow, R. N. Sharpe and J. C. Hung, "*On the Application and Design Consideration of Artificial Neural Network Fault Detectors*" IEEE Trans. Ind. Electron, vol. 40, pp. 181–198, April 1993

[5] A. D. Ball, W. Li, F. Gu and Z. Shi, "*The Analysis of Engine Acoustic Signal Using Higher Order Statistics*" Maintenance and Reliability Conference, USA, vol. 2, pp. 33.01-33.09, May 2000.

[6] B. Li, G. Goddu and M. Y. Chow, "*Detection of Common Motor Bearing Faults using Frequency Domain Vibration Signals and a Neural Network Based Approach*" in *Proc.* American Control Conf. pp. 2032–2036, 1998.

[7] B. Randall, "*Detection and diagnosis of incipient bearing failure in helicopter gearboxes*", Engineering Failure Analysis, vol.11, pp. 177–190, 2004.

[8] D. J. Sobajic and Y. H. Pao, "*Artificial Neural-net Based Dynamic Security Assessment for Electric Power Systems*" IEEE Trans. Power Systems. vol. 4, no. 1 pp. 220-228, February 1989.

[9] M. A. El-Sharkawi, R. J. Marks II, M. E. Aggoune, D. C. Park, M. J. Damborg and L. E. Atlas, "*Dynamic Security assessment of Systems using Back Error Propagation Artificial Neural Networks*" In Second Symposium on Expert Systems Applications to Power Systems, Seattle USA, vol. 1, pp. 620-627, April 2004.

[10] J. Kittler, A. Etemadi and N. Chaokjarenwanti, "*Feature Selection and Extraction in Pattern Recognition and Image Processing*" R. A Vaughan, Ed., Proceedings of the 37th Scottish University Summer School in Physics, 1990.

[11] S. Lu, "*A Transfer Matrix Method for Nonlinear Vibration Analysis of Rotor-Bearing Systems*" presented at the 1991 ASME Design Tech. Conf.—13th Biennial Conf. Mechanical Vibration and Noise, Miami, FL, pp. 75–83, September 22–25, 1991

[12] Lou, X., K. A. Loparo, F. M. Discenzo, J. Yoo and A. Twarowski, "*A model-based technique for rolling element bearing fault detection*" Mechanical Systems and Signal Processing, vol. 18, pp. 1077–1095, 2004.

[13] Chu, E. and George, A. "*Inside the FFT Black Box: Serial and Parallel Fast Fourier Transform Algorithms"* Boca Raton, FL: CRC Press, 2000.

[14] H. Ocak, and K. A. Loparo, "*Estimation of the running speed and bearing defect frequencies of an induction motor from vibration data*", Mechanical Systems and Signal Processing, vol. 18, pp. 515–533, 2004.

[15] W. siedlecki and J. Sklansky "*A Note on Genetic Algorithms for Large Scale*" Pattern Recognition Letter. pp. 335-347, 1989

[16] Tang, K. S. Man, K. F. Kwong and Q. He, "*Genetic Algorithms and Their Applications*" IEEE Signal Processing Magazine, vol. 3, pp. 21-37, November, 1991.

[17] M. Vlachos, M. Domeniconi, C. Gunopulos and G. Kollios, "*Non-Linear Dimensionality Reduction Techniques for Classification and Visualization*" Proceedings of 8th SIGKDD Edmonton, Canada, 2002.

[18] Lucasius and G. Kateman, "*Towards Solving Subset Selection Problems with the Aid of the Genetic Algorithm*" PPSN 2, pp. 239-247, 1992.

[19] M. Y. Chow, A. V. Chew, and S. O. Yee, "*Performance of a Fault Detector Artificial Neural Network using Different Paradigms*" in Proc. SPIE Applications of Artificial Neural Networks III Conf., 1992.

[20] Bo Li, Mo-Yueng Chow, Yodyim Tipsuwn and C. Hung, "*Neural Network Based Motor Rolling Bearing Fault Diagnosis*" IEEE Transactions on Industrial Electronics, vol. 47, no. 5, pp. 637-646, October, 2000.

[21] X. Lou and K. A. Loparo, "*Bearing fault diagnosis based on wavelet transform and fuzzy inference*", Mechanical Systems and Signal Processing, vol. 18, pp. 1077–1095, 2004.

[22] S. Ericsson, N. Grip, E. Johansson, L. E. Persson, R. Sjöberg and J. O. Strömberg, "*Towards automatic detection of local bearing defects in rotating machines*" Mechanical Systems and Signal Processing, vol. 19, pp. 509-535, 2004.

[23] S. Prabhakar, A. R. Mohanty and A. S Sekhar, "*Application of discrete wavelet transform for detection of ball bearing race faults*", Tribology International, vol. 35, pp. 793-800, 2002.

[24] C. Mercer, "*Does the Signal have a Gaussian Probability Density*", Institute of Sound and Vibration Research, Southampon University, UK., 2001.

[25] R. O. Duda, P. E. Hart and D. G. Stork "*Pattern Classification"* Second Edition, Wiley-Interscience Publications, 2001.

[26] Kenneth A. Loparo, "*Seeded Fault Test Data*", Bearing Data Center, Case Western Reserve University, Cleveland, Ohio, 2004 last accessed: August, 2005, <http://www.eecs.case.edu/laboratory/bearing/welcome_overview.htm>

[27] L. B. Jack and A. K. Nandy "*Genetic Algorithms for Feature Selection in Machine condition Monitoring with Vibration Signals*" IEEE Processing, Image Signal Processing, vol. 147, no. 3, pp. 205-212, June 2000.

[28] Hansen, J. Steven and G. Roger "*A New Method for Rolling Element Bearing in Petrochemical Industry*" Presented at the Vibration Institute Seminar, New Orleans, Luoisiana, June 2004, last accessed: October, 2005, <http://www.bently.com/articles/apnotes/apnotes.htm>

[29] Ian T. Nabney, "*Netlab: Algorithms for Pattern Recognition"* Springer-Verlag, United Kingdom, 2001.

[30] F. Provost, T. Fawcett, and R. Kohavi "The *case against Accuracy Estimation for Comparing Induction Algorithms*" Proc. 15th International Conf. on Machine Learning, Morgan Kaufman, San Francisco, CA, pp. 445-453, 1998.

[31] H. Hamilton, E. Gurak, L. Findlater "*Confusion Matrix*" University of Regina, Regina, February 2002, last accessed January, 2006. <http://www2.cs.uregina.ca/~hamilton/courses/831/notes/confusion_matrix>

[32] G. Nikolas "*Rolling Element Fault Diagnostics*" August 2005, last accessed January, 2006, <www.itiball.com/Glossary.html>

# Appendix A: Table of Results

As defined earlier, a **confusion matrix** is a visualization tool typically used in supervised learning. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. One benefit of a confusion matrix is that it is easy to see if the system is confusing two classes (i.e. commonly mislabeling one as an other). Tables below present confusion matrixes for the 10 different types of faults with statistical test set, spectral test set and the combination of both.

## A.1 Statistical Test Set Results

**Table 12: Statistical test set result using 3 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 215 | | | | 57 | | | 58 | | |
| IR Low | 300 | 30 | | | | | | | | |
| IR Medium | 263 | | 67 | | | | | | | |
| IR High | 289 | | | 41 | | | | | | |
| OR Low | 291 | | | | 39 | | | | | |
| OR Medium | 281 | | | | | 49 | | | | |
| OR High | 290 | | | | | | 40 | | | |
| BO Low | 310 | | | | | | | 20 | | |
| BO Medium | 269 | | | | | | | | 61 | |
| BO high | 280 | | | | | | | | | 50 |

From Table 12 above, of the 330 healthy bearings, the system predicted that 57 were low level Outer Race faults, 58 were low level Ball faults while 215 where actually correctly classified. Likewise of the 330 low level Inner Race (IR low) faults, 300 were misclassified as healthy bearings while only 30 were actually correctly classified. Also of the 330 medium level Inner Race faults (IR medium), 263 were misclassified as healthy bearings while only 67 were properly identified. The same logic can be utilized in the remainder of the confusion matrix tables. Clearly the current system has trouble distinguishing between faulty and healthy bearings. The number of neurons in the system was increased to 5 and the results are shown in Table 13 below.

**Table 13: Statistical test set result using 5 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 285 | | | | 45 | | | | | |
| IR Low | 255 | 75 | | | | | | | | |
| IR Medium | 235 | | 95 | | | | | | | |
| IR High | 242 | | | 88 | | | | | | |
| OR Low | 244 | | | | 86 | | | | | |
| OR Medium | 234 | | | | | 96 | | | | |
| OR High | 242 | | | | | | 88 | | | |
| BO Low | 285 | | | | | | | 45 | | |
| BO Medium | 225 | | | | | | | | 105 | |
| BO high | 233 | | | | | | | | | 97 |

Although there is a slight increase in the number of faulty bearings correctly classified as shown in Table 13, the general performance of the system remains poor. In all cases, less than the third of the total number of faulty bearings are recognized by the system.

**Table 14: Statistical test set result using 7 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 301 | | 19 | | | 10 | | | | |
| IR Low | 179 | 151 | | | | | | | | |
| IR Medium | 163 | | 167 | | | | | | | |
| IR High | 174 | | | 156 | | | | | | |
| OR Low | 172 | | | | 158 | | | | | |
| OR Medium | 162 | | | | | 168 | | | | |
| OR High | 170 | | | | | | 160 | | | |
| BO Low | 206 | | | | | | | 124 | | |
| BO Medium | 153 | | | | | | | | 177 | |
| BO high | 168 | | | | | | | | | 162 |

Table 14 shows the statistical test set performance of the system when 7 neurons are used in the hidden layer. It can be see from the Table that the system in question still has trouble classifying faulty bearings, but seems to be able to correctly classify healthy bearings. Clearly, of the 330 healthy bearings, the system did a good job of correctly classifying 301 and wrongly predicted that 19 of them were medium Inner Race faults and 10 were predicted as medium Outer Race faults. 151 low IR faults were correctly classified and 179

misclassified as normal bearings. Out of 330 high level Ball faults (BO high) 162 were correctly classified while 168 were misclassified. Again there is a slight increased in the performance of the system but the number of misclassified bearings is still high.

**Table 15: Statistical test set result using 10 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 307 | | | | | 23 | | | | |
| IR Low | 121 | 209 | | | | | | | | |
| IR Medium | 107 | | 223 | | | | | | | |
| IR High | 118 | | | 212 | | | | | | |
| OR Low | 116 | | | | 214 | | | | | |
| OR Medium | 106 | | | | | 224 | | | | |
| OR High | 113 | | | | | | 217 | | | |
| BO Low | 148 | | | | | | | 182 | | |
| BO Medium | 97 | | | | | | | | 233 | |
| BO high | 112 | | | | | | | | | 218 |

Table 15 shows the statistical test result using 10 neurons in the hidden layer. Here 307 healthy bearings are correctly identified by the system compare to 301 in the previous case. The total number of faulty bearings properly identified by the system as well has increased. In spite of these improvements, it is clear that the general performance of the system is not satisfactory as the number of false alarms remains high.

## A.2 Spectral Test Set Results

The classifier that used the spectral input vector generally performed better. With only 3 neurons in the hidden layer as shown in Table 16, more than a third of almost all faulty bearings are correctly identified. Of the 330 healthy bearings, 280 are correctly identified while 50 are predicted as low level IR faults. When the number of neurons is increased to 5 this performance is further increased as shown in Table 17. Of the 330 healthy bearings, 293 are correctly identified by the system and 37 are predicted as low level IR faults. In almost all faulty bearings cases, more than 200 of them are properly identified except for the IR low and the BO low cases. This performance gets even better when the number of neurons is increased to 7 and 10 as shown in Table 18 and 19. With 10 neurons more than 300 bearings are correctly identified with a case where all bearings are correctly identified by the system

**Table 16: Spectral test set result using 3 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 280 | 50 | | | | | | | | |
| IR Low | 217 | 113 | | | | | | | | |
| IR Medium | 197 | | 133 | | | | | | | |
| IR High | 204 | | | 126 | | | | | | |
| OR Low | 206 | | | | 124 | | | | | |
| OR Medium | 196 | | | | | 134 | | | | |
| OR High | 204 | | | | | | 126 | | | |
| BO Low | 244 | | | | | | | 86 | | |
| BO Medium | 187 | | | | | | | | 143 | |
| BO high | 195 | | | | | | | | | 135 |

**Table 17: Spectral test set result using 5 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 293 | 37 | | | | | | | | |
| IR Low | 141 | 189 | | | | | | | | |
| IR Medium | 121 | | 209 | | | | | | | |
| IR High | 128 | | | 202 | | | | | | |
| OR Low | 130 | | | | 200 | | | | | |
| OR Medium | 120 | | | | | 210 | | | | |
| OR High | 128 | | | | | | 202 | | | |
| BO Low | 168 | | | | | | | 162 | | |
| BO Medium | 111 | | | | | | | | 219 | |
| BO high | 119 | | | | | | | | | 211 |

**Table 18: Spectral test set result using 7 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 317 | 13 | | | | | | | | |
| IR Low | 66 | 264 | | | | | | | | |
| IR Medium | 53 | | 277 | | | | | | | |
| IR High | 53 | | | 277 | | | | | | |
| OR Low | 55 | | | | 275 | | | | | |
| OR Medium | 45 | | | | | 285 | | | | |
| OR High | 53 | | | | | | 277 | | | |
| BO Low | 82 | | | | | | | 248 | | |
| BO Medium | 36 | | | | | | | | 294 | |
| BO high | 44 | | | | | | | | | 286 |

**Table 19: Spectral test set result using 10 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 324 | 6 | | | | | | | | |
| IR Low | 33 | 297 | | | | | | | | |
| IR Medium | 20 | | 310 | | | | | | | |
| IR High | 27 | | | 307 | | | | | | |
| OR Low | 22 | | | | 308 | | | | | |
| OR Medium | 12 | | | | | 318 | | | | |
| OR High | 20 | | | | | | 310 | | | |
| BO Low | 49 | | | | | | | 281 | | |
| BO Medium | 0 | | | | | | | | 330 | |
| BO high | 11 | | | | | | | | | 319 |

## A.3 Combination of Inputs Test Set Results

**Table 20: Combination of inputs test set result using 3 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 288 | | 42 | | | | | | | |
| IR Low | 185 | 145 | | | | | | | | |
| IR Medium | 172 | | 158 | | | | | | | |
| IR High | 163 | | | 167 | | | | | | |
| OR Low | 174 | | | | 156 | | | | | |
| OR Medium | 164 | | | | | 166 | | | | |
| OR High | 172 | | | | | | 158 | | | |
| BO Low | 201 | | | | | | | 129 | | |
| BO Medium | 155 | | | | | | | | 175 | |
| BO high | 172 | | | | | | | | | 158 |

**Table 21: Combination of inputs test set result using 5 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 312 | | 18 | | | | | | | |
| IR Low | 88 | 242 | | | | | | | | |
| IR Medium | 83 | | 247 | | | | | | | |
| IR High | 66 | | | 264 | | | | | | |
| OR Low | 77 | | | | 253 | | | | | |
| OR Medium | 67 | | | | | 263 | | | | |
| OR High | 75 | | | | | | 255 | | | |
| BO Low | 101 | | | | | | | 229 | | |
| BO Medium | 58 | | | | | | | | 272 | |
| BO high | 67 | | | | | | | | | 263 |

**Table 22: Combination of inputs test set result using 7 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 321 | | 9 | | | | | | | |
| IR Low | 88 | 287 | | | | | | | | |
| IR Medium | 83 | | 292 | | | | | | | |
| IR High | 66 | | | 309 | | | | | | |
| OR Low | 77 | | | | 298 | | | | | |
| OR Medium | 67 | | | | | 308 | | | | |
| OR High | 75 | | | | | | 300 | | | |
| BO Low | 101 | | | | | | | 274 | | |
| BO Medium | 58 | | | | | | | | 317 | |
| BO high | 67 | | | | | | | | | 308 |

**Table 23: Combination of inputs test set result using 10 neurons**

| | Normal | IR Low | IR Medium | IR High | OR Low | OR Medium | OR High | BO Low | BO Medium | BO high |
|---|---|---|---|---|---|---|---|---|---|---|
| Normal | 325 | | 5 | | | | | | | |
| IR Low | 22 | 308 | | | | | | | | |
| IR Medium | 17 | | 313 | | | | | | | |
| IR High | 0 | | | 330 | | | | | | |
| OR Low | 11 | | | | 319 | | | | | |
| OR Medium | 0 | | | | | 330 | | | | |
| OR High | 10 | | | | | | 320 | | | |
| BO Low | 25 | | | | | | | 305 | | |
| BO Medium | 0 | | | | | | | | 330 | |
| BO high | 0 | | | | | | | | | 330 |