REAL-TIME UNSUPERVISED INCREMENTAL SUPPORT VECTOR MACHINE FOR OIL AND GAS PIPELINE NDT SYSTEM

NIK AHMAD AKRAM BIN NIK ZULKEPELI

THESIS SUBMITTED TO THE UNIVERSITY OF NOTTINGHAM FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FEBRUARY 2016

ABSTRACT

Current Non Destructing Testing (NDT) techniques for oil and gas pipeline inspection are accurate and reliable but there are limited numbers of continuous monitoring technique available that can automatically make real-time decisions on the status of the pipeline. Furthermore, most of the NDT methods are deployed at predetermined interval which can last for several months. Sudden onsets of defects are undetected and lead to pipeline failure and unscheduled shutdown. A reliable inspection method is required whereby the pipelines are monitored continuously and are able to provide the operators sufficient time to plan and organize shutdowns. In order to implement this, a continuous monitoring technique is needed which can detect defects automatically with minimal human intervention.

Support Vector Machine (SVM) is a powerful machine learning technique for classification, however, the training phase requires batch data to find a model and this is not feasible for a continuous NDT system. This thesis proposes a novel method where the SVM training phase is able to find a model from the incremental dataset acquired from Long Range Ultrasonic Testing (LRUT) system. Results show that this method has comparable accuracy compared to the batch data method.

Traditionally, SVM training data is labeled by an expert, however in a continuous monitoring NDT, it is not practical to assign an expert to label the continuously acquired data. Therefore, a novel unsupervised training technique is proposed. The technique is able to cluster the acquired data into a few clusters

accurately. The performance of the proposed technique is compared to Self Organizing Map (SOM) method and shows better results.

This thesis also proposes a novel method to implement a Genetic Algorithm (GA) as the Quadratic Programming (QP) solver in the SVM efficiently. Conventional SVM implement Sequential Minimal Optimization (SMO) which requires that the data be sparse for optimal operation. The performance of the method is evaluated and shows comparable result to traditional methods.

As such, this thesis provides the framework to perform unsupervised continuous monitoring for oil and gas pipelines using LRUT in real time.

PUBLICATIONS

1. Akram, N. A., Isa, D., Rajkumar, R., & Lee, L. H. (2014). Active incremental Support Vector Machine for oil and gas pipeline defects prediction system using long range ultrasonic transducers. *Ultrasonics*, *54*(6), 1534-1544.

2. Akram, N. A., Shafiabady, N., & ISA, D. (2014). Genetic Algorithm as Quadratic Programming Solver for Support Vector Machine. *International Journal of Research in Computer Engineering & Electronics*, *3*(3).

3. Akram, N. A., Shafiabady, N., & Isa, D. (2013, December). Using Neural Networks as Pipeline Defect Classifiers. In *IT Convergence and Security (ICITCS), 2013 International Conference on* (pp. 1-4). IEEE.

4. Hassan, M., Isa, D., Rajkumar, R., Akram, N. A., & Arelhi, R. (2013). Reducing Support Vector Machine Classification Error by Implementing Kalman Filter. *International Journal of Intelligent Systems and Applications (IJISA)*, *5*(9), 10.

ACKNOWLEDGEMENTS

I would like to express the deepest appreciation to my supervisor, Professor Dino Isa for his supervision, guidance and never ending support during the course of the research. Without his supervision and constant help this research would not have been possible.

I would also like to thank Dr. Rajprasad K.Rajkumar for his technical insight and guidance throughout my studies.

I would also like to thank my family for their encouragement during the course of my studies. Special thanks to my wife Izzatul ‘Aliaa for her understanding, love and motivation. Her support and encouragement was in the end what made this thesis possible.

The equipment in this research is mainly funded by Ministry of Science, Technology and Innovation (MOSTI) through E-Science program.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AOPL	USA Association of Oil Pipe Lines
BPXA	BP Exploration Alaska
BWE	Back Wall Echo
CUI	Corrosion under Insulation
DAQ	Data Acquisition
FC	Full Circumferential
FFT	Fast Fourier Transform
GA	Genetic Algorithm
IDS	Intrusion Detection System
LRUT	Long Range Ultrasonic Testing
MOSTI	Ministry of Science, Technology and Innovation
NDT	Non Destructive Testing
NTSB	National Transportation Safety Board USA) (NTSB)
PCA	Principal Component Analysis
PHMSA	US Department of Transportation Pipeline and Hazardous Materials Safety Administration
QP	Quadratic Programming
RBF	Radial Basis Function
SCC	Stress Corrosion Cracking
SOM	Self Organizing Map
SV	Support Vector
SVM	Support Vector Machines
TCO	Columbian Gas Transmission

CHAPTER 1: INTRODUCTION

1.1. Introduction

Pipeline systems are a normally safe and an extremely effective method for mass-scale delivery of gas and liquid formed products [1][2]. According to the USA Association of Oil Pipe Lines (AOPL), system losses are around 1 gallon per million barrel-miles (One barrel-mile = one barrel transported a mile) where a standard barrel contains 42 gallons (159 liters)[3]. In other words, less than one teaspoon of oil is spilled per thousand barrel-miles. Pipeline systems are considered as the most cost-effective as compared to railway and road transportation in the long term [4].

Latest advancements in sensing technology have yielded a system that can **continuously** be used to monitor pipeline segments using Long Range Ultrasonic Testing (LRUT) [5]. It is specifically designed to detect corrosion under insulation (CUI) and has many advantages over other NDT techniques. Utilizing guided waves, this system is able to monitor pipelines over a distance of several hundred meters from a single location using a ring of ultrasonic transducers permanently fixed onto the pipe. Although this technology exists, however, there is currently no means of acquiring and transmitting the acquired data from remote locations **continuously** [6]. The reason is that a real time system would needs monitoring and classification software that can automatically make decisions on the condition of the pipe. This necessitates that the proposed system is able to work in real-time and can detect and make decision with no human intervention. Standard machine learning techniques such as conventional Support Vector Machines and Neural Networks are not suitable

because they require expert to assign labels to each training sample. In addition, the incremental data (one-at-a-time) acquired is not suitable with conventional Support Vector Machines and Neural Networks as they are designed to work on batch data from larger data files [7]–[9].

Figure 1: Experimental Rig

This project proposes a system that can detect and classify pipeline defects in real time. The system will be able to work unsupervised. However minimal human intervention will be required during deployment and start-up. This is because required information such as diameter and thickness of the pipe are unavailable and determined during the deployment stage. The state of the art technique in unsupervised learning is investigated as well as incremental learning techniques to develop software capable of performing real-time defect detection and classification with minimal human

intervention. The system is tested using acquired data from the experimental rig with simulated defects shown in Figure 1.

1.2. Project Scope

This project proposes a real-time unsupervised defect detection and classification in oil and gas pipelines. The designed experimental rig focuses on developing a system that can detect and classify defects level in a test pipe without needing function such as channel selection and multi wave mode. Thus, only defect depth is detected and classified. In the future, various defect types can be detected and classified.

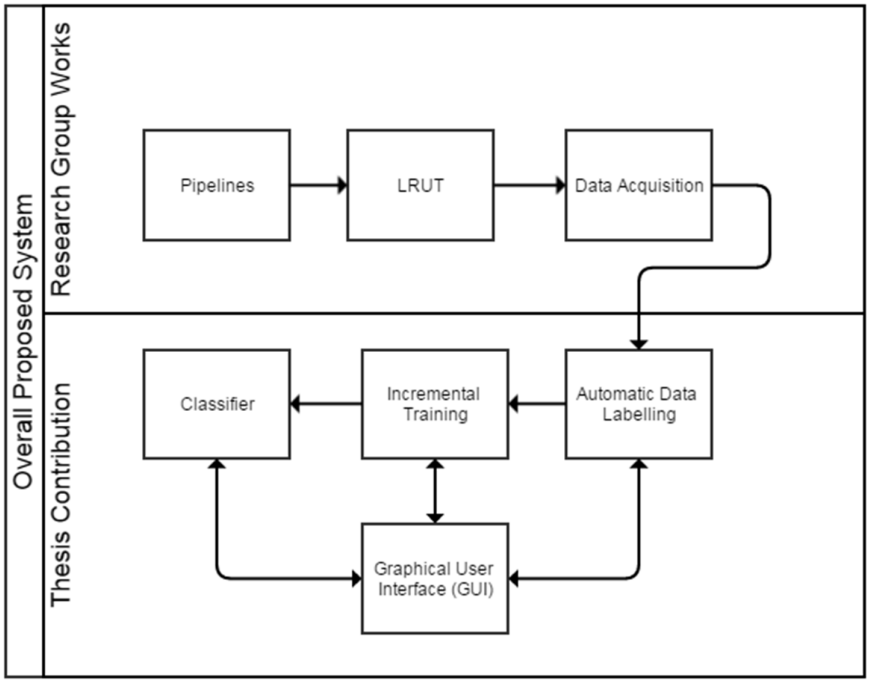


Figure 2: Overall proposed system and thesis contribution.

Figure 2 shows the thesis contribution on the project. The thesis contributes mostly on the algorithms and software part. The hardware parts are from the research group works Intelligent Systems Research Group (ISRG). The hardware parts are done by Dr. Rajprasad K.Rajkumar.

1.3. Research Issues

1. Conventional machine learning algorithms require human experts to interpret and assign labels to the training data. However this project deals with a real-time system that will collect huge amounts of data. Thus, it is not feasible to use humans as interpreters for this system in real-time. Also in this domain human experts are very rare. In short the proposed system must run with minimal or no human intervention.

2. Machine learning algorithms especially standard Support Vector Machines are in essence use batch learning, which make it unable to handle incremental data from an online system. However, this project deals with online systems which acquire data continuously. Therefore, an incremental learning technique is mandatory to the system

3. Training a support vector machine requires the solution of a very large quadratic programming (QP) problem. Numerical QP optimization technique such as active-set method is resource-consuming [10]. Thus, a new approach to solve the QP problems is essential for this project.

1.4. Aim and Objectives

1. To design and implement a machine learning technique on a NDT system that is able to detect and classify defects levels on experimental rig.
2. To develop a SVM incremental learning technique for defect detection in a continuous monitoring NDT system.
3. To develop an unsupervised learning method for automatic training data labeling.
4. To incorporate Genetic Algorithm (GA) in support vector machine for replacing the conventional Quadratic Programming solver in order to use less computational resources.
5. To test and justify functionality of the developed system using the experimental rig.

1.5. Deliverables

1. An incremental learning algorithm which can work in real-time and can process huge amount of data.
2. An unsupervised algorithm for assigning label to the training data with zero human intervention.

3. A novel technique to solve Quadratic Programming problems which can work in real-time and can process huge amount of data, over 100,000 data points per seconds with 16-bit resolution (200 kilobytes per seconds).

1.6. Organizations of Thesis

Chapter 2: Literature Review - This chapter reviews some of the related works in the area oil and gas pipeline NDT, support vector machine (SVM), genetic algorithm (GA), quadratic programming (QP) and unsupervised learning.

Chapter 3: Methodology - This chapter provides the approach takes in order to fulfill the deliverables of the projects. This includes the design and the setup for simulation rig for the oil and gas NDT system. It also details the proposed algorithms and techniques for incremental SVM, unsupervised learning and incorporating genetic algorithm in Support Vector Machine.

Chapter 4: Results and Discussion - This chapter discusses the results of the proposed algorithms on the acquired data obtained from the experimental rig. Detailed discussions on each set of results are also presented. The original idea of unsupervised and incremental learning machine learning algorithm are especially discussed and reviewed.

Chapter 5: Conclusion and future work - This chapter summarizes the major accomplishment and review the achieved objectives especially with regard to successfully detecting and classifying the defects. It also discusses the possibilities of improvements and future work for this project

CHAPTER 2: LITERATURE REVIEW

This chapter reviews some of the important and recent literature related to the project. An emphasis is placed on reviewing literature for pipeline corrosion, machine learning in NDT technology, SVM, genetic algorithm and unsupervised technique

2.1. Pipeline Corrosion

The main problem faced by the oil and gas industry, having been in operation for 185 years, is that aging pipeline systems are being corroded and are undergoing wall thinning which can eventually lead to pipeline failure [11]. The main causes of pipeline failures around the world are corrosion defects such as cracking, pitting and Stress Corrosion Cracking (SCC) [12][13].

Some of the causes of pipeline failures can be avoided; especially corrosion [14]. Figure 3 shows that corrosion causes 18.4% of total accidents. Currently, pipeline inspection is done at predetermined intervals using techniques such as pigging where operators must be physically present to perform measurements and make judgments on the integrity of the pipes. The condition of the pipe between these testing periods, which can last for several months, can go unmonitored [15]. As such there is a need for a continuous monitoring system.

The numbers of factors that cause failures in pipes are large and mostly unexpected. Thus the main cause for frequent pipe failures and leaks are the defects which occur suddenly.

Table 1 shows the reported cause of pipeline failure incidents.

Reported Cause of Incident	**Number**
Excavation Damage	1950
Corrosion	**1926**
Incorrect Operation	742
Material, Welding, Equipment Damage	2818
Natural Force Damage	714
Other Outside Force Damage	765
All Other Causes	1535
Totals	10450

Table 1: PHMSA All Reported Pipeline Incidents By Cause in USA 1993 – 2013.(Dec 6,2013)[16]

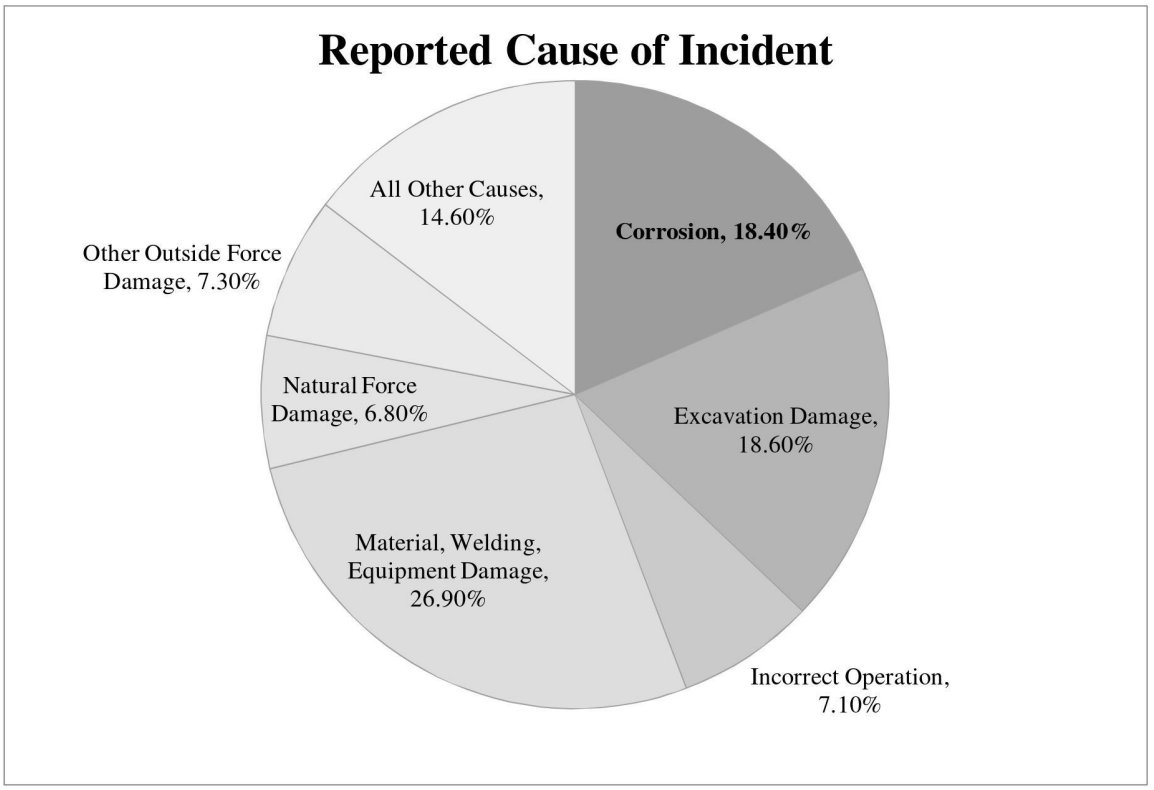


Figure 3: Causes of Reported Pipeline Incident by Percentage

In the year 2000, a study performed by CC Technologies (Ohio, USA) found that the costs of corrosion to the United State of America (USA) economy alone have been estimated to be between 70 billion USD and 126 billion USD annually. The estimated costs related to these defects for oil and gas industry are shown in Table 2. The study states that corrosion processes have great impact in the deterioration of industry utilities and it is one of the main risks for major accidents in oil-gas facilities [17].

Industry	Cost (billion USD)
Gas and Liquid Transmission Pipelines	7
Gas Distribution	5
Oil and Gas Exploration and Production	1.4
Petroleum Refining	3.7

Table 2: Corrosion cost related to industry in oil and gas sector 1999-2001. [17]

2.1.1. Reported Incidents Related to Corrosion in Oil and Gas Pipeline

There have been many reported pipeline accidents that resulted from corrosion related damage.

Figure 4: December 11, 2012. Natural gas pipeline explosion in Sissonville, West Virginia, USA.[18]

On December 11, 2012, a segment of natural gas pipeline operated by Columbian Gas Transmission (TCO) exploded at Archibald Hill. The accident caused 80 to 90 feet of shooting flames and a huge amount of debris. Figure 4 shows a picture of the incident. Furthermore, the explosion burned three homes and damaged a portion of nearby highway. The initial investigation into the incident revealed that the ruptured pipe was installed in 1967, and was heavily corroded[18][19].

Figure 5: August 19,2000. Natural gas pipeline explosion in Pecos River, New Mexico, USA .[20]

On August 19, 2000, a 30-inch-diameter natural gas transmission pipeline operated by El Paso Natural Gas Company ruptured adjacent to the Pecos River, New Mexico, USA. Figure 5 shows the location of the incident and the ruptured pipe. The released gas ignited and burned for 55 minutes. The incident killed 12 persons and destroyed 3 vehicles. Investigation by National Transportation Safety Board USA) (NTSB) found that the rupture was a result of the reduction in thickness of pipe due to

internal corrosion. The thin pipe wall was not able to withstand the pressure within the pipe. Property and other damages cost the operator an estimated 998,296 US dollar [20].

On March 2, 2006, BP Exploration Alaska (BPXA) underwent an orderly and phased shutdown of the Prudhoe Bay oil field following the discovery of unexpectedly severe corrosion and a spill from a Prudhoe Bay oil transit line. This incident reduced oil production of the United States by an estimated 400,000 barrels per day and in turn[21]. According to company officials, at least 73% of the pipeline needed replacement due to extensive corrosion[22][23]. In November 2007, BPXA pled guilty and was fine 20 million US Dollar and in July 2011, BPXA paid a 225 million US Dollar fine as a civil penalty. In addition, BPXA agreed to take measures to significantly improve inspection and maintenance of its pipeline infrastructure on the North Slope as shown in Figure 6 [24].

Figure 6: A map of northern Alaska; the dotted line shows the southern boundary of the North Slope. [25]

In Malaysia, on 10th June 2014, Sabah-Sarawak interstate pipeline operated by Petronas exploded. The incident happened at around 2am near Bukit Beriwan in the district of Lawas, about 135 km from the Sabah Oil & Gas Terminal in Kimanis, Sabah. The pipeline is meant to channel raw gas from Kimanis in Sabah to Bintulu in Sarawak via a 600km pipeline [26], [27].

Figure 7: Exposed pipeline that exploded near Bukir Beriwan. [26]

In conclusion, it is crucial to implement a suitable pipeline Non-Destructive Testing (NDT) system in real time with continuous monitoring systems because it is able to provide operator the information on the pipeline integrity and then prevent catastrophic pipeline failures.

2.2. Ultrasonic Guided Wave in NDT

The integrity of pipelines used in the oil and gas industry need to be monitored on a regular basis to avoid unplanned interruption and accidents that may cost money to the operator. In the worst case, the accidents may cause fatality to nearby population and damage to the environment[28]–[30]. The process to monitor pipeline is challenging in offshore applications where most of the pipes are submerged in the

sea. NDT system implementing ultrasonic sensors are excellent for monitoring pipelines integrity as it provides accurate information on pipe wall condition and does not affect the process flow[15][31][32].

Traditionally, pipeline operators implement point-by-point NDT techniques, for example the ultrasonic thickness gauging method. The method may take a long time to execute and sense especially for inspection of pipes circumferentially for several hundred meters is needed as shown in Figure 8 [33], [34].

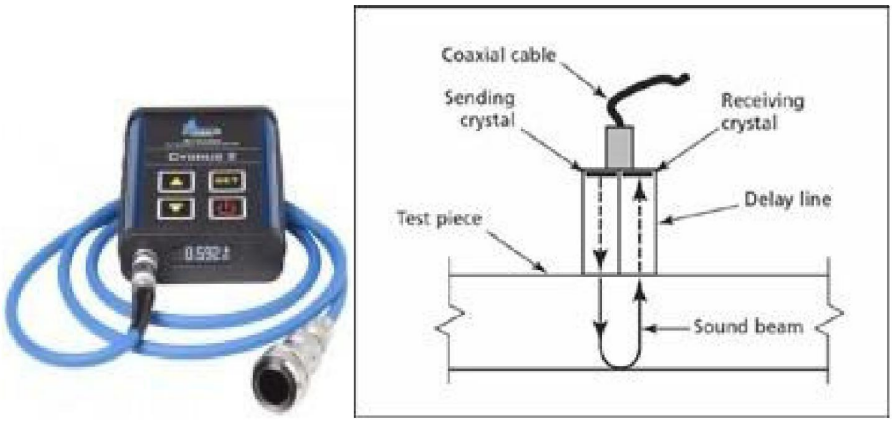


Figure 8: Ultrasonic thickness gauging technique. [35]

An initial screening method that is fast and able to provide sufficient information for pattern recognition to determine the location of the corrosion may save a lot of time. Once, the method successfully locates the corrosion, the point-by-point procedure is carried out to determine the extent of the corrosion.

Ultrasonic guided wave techniques offer fast screening using a method commonly known as Long Range Ultrasonic Testing (LRUT)[36][37].

Figure 9: Commercial LRUT probe. [38]

Originally, the method is developed for applications for insulated pipes where the removal of pipe insulation is difficult [39][40]. The ultrasonic signals are transmitted and received by the same transducers using pulse-echo transmission procedure. The signals travel along the length of the pipe. Some of the signals are reflected at the defects[6][41]. Illustration in Figure 10 shows the comparison between conventional ultrasonic and guided wave testing technique.

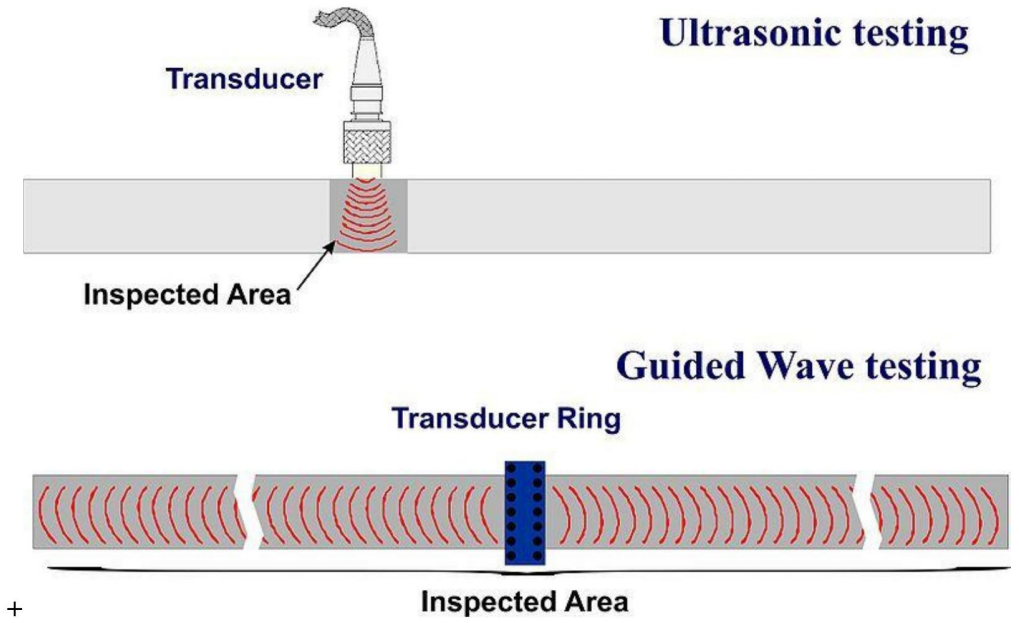


Figure 10: Comparison between conventional ultrasonic and guided wave technique. [42]

2.3. Long Range Ultrasonic Testing (LRUT)

LRUT is able to screen pipeline for corrosion under insulation. Screening means that the method is able to locate the position of corrosion for less or 100% of the pipeline length economically[43][44].

LRUT implements guided wave and offer 100% coverage of the pipe wall. Therefore, pipes can be examined from one location for lengths ranging from several meters up to several 100 meters [45][46].In LRUT systems, multiple ultrasonic transducer elements are fixed around a special collar that can be placed circumferentially on a pipe as shown in Figure 11. These transducers will transmit and

acquire the return waves using the same device and transceivers (transmitter-receiver) and. The transducers are specifically designed to transmit ultrasonic signal that travel along the length of the pipe rather than perpendicularly below. The signals that are reflected by the defects are a function of depth and circumferential extent metal loss.

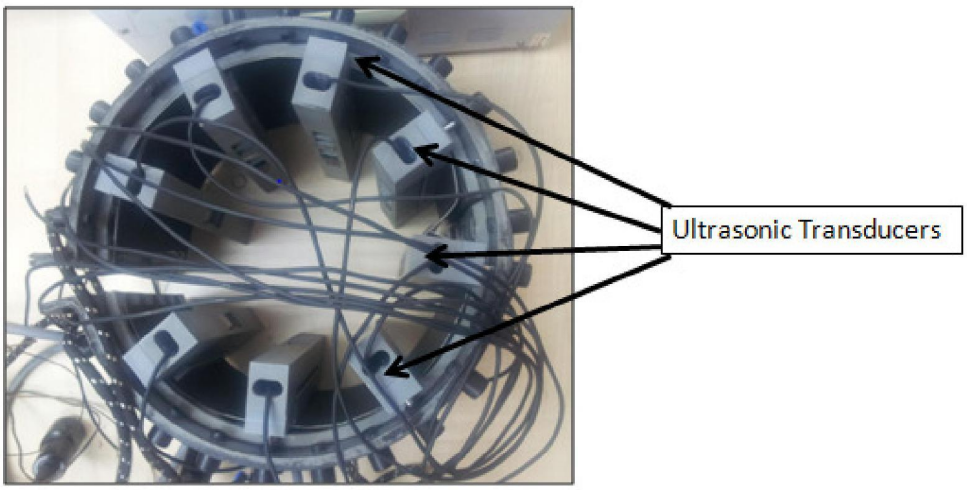


Figure 11: Array of transducer elements fitted around a collar.

The LRUT method is normally used as a screening tool for more detailed inspection. The method is capable in detecting corrosion within several 100 meters from the transducers in both directions. In addition, this method has excellent performance in detecting circumferential damage and wall loss.[47].

One of the related works that implement ultrasonic guided wave inspection is by Zhang, L et al[48]. The objective of that work was to inspect a long pipe using a single LRUT transducer and classify multiple types of defect. The author proposed a phased array focusing method. The proposed method improves the performance of

LRUT by focusing the energy onto a defect. By concentrating the energy, the proposed method is able to reduce false alarm, and able to locate defects more accurately. The results of the experiment showed the proposed method improved the ability of LRUT to detect and locate multiple types of defects. Figure 12 shows the arrangement of the experiment.

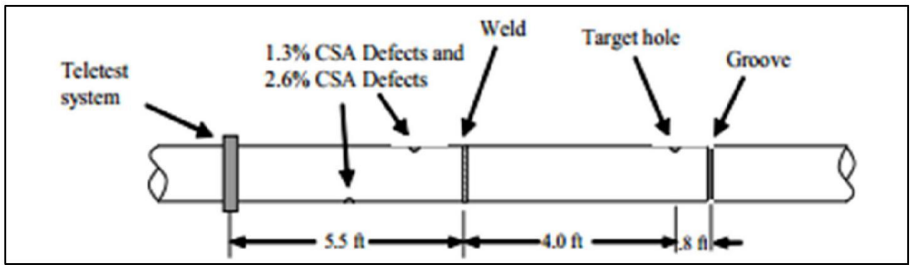


Figure 12: Zhang. L et al Experiment Arrangement. [48]

In conclusion, the latest advancements in sensing technology have yielded a system that is able to continuously monitor pipeline segments using LRUT [40]. This system is able to monitor pipelines over a distance of several hundreds of meters from a single location using a ring of ultrasonic transducers permanently fixed onto the pipe. LRUT was specifically designed for inspection of Corrosion under Insulation (CUI) and has many advantages over other NDT techniques and have seen its widespread use in many other applications [43][49][50]. It is also able to detect both internal and external corrosion which makes it a more efficient and cost-saving alternative. With the recent developments in permanent mounting system using a special compound, the ability to perform a continuous monitoring system has now

become a reality[40][50]. Although this technology exists, however, currently there is no continuous monitoring and classification software available that can automatically make real-time decisions on the status of the pipeline [51].

2.4. Artificial Intelligence used in the NDT domain

Piervincenzo Rizzo et al proposed an NDT system which implements the LRUT method and uses Neural Network as the classifier for acquired data [52]. In the work, the pipe is inspected by ultrasonic guided wave technique. The defect type is a small notch cut in a steel pipe at depths ranging from 1% to 17% of the pipe cross-sectional area. In the first stage, a semi-analytical finite element method was implemented to model wave propagation in the pipe. Then, reflections of the waves were measured. Six features were extracted from the discrete wavelet decomposition of the raw signals and from the Hilbert and Fourier transforms of the reconstructed signals. A six-dimensional damage index was then constructed. Next, a neural network was used to classify the size and the location of the notch. Overall, the proposed method demonstrated good classification performance and robustness against noise. Figure 13 shows the setup used in the experiment. The method proposed by Piervincenzo Rizzo et al is not feasible for this project as it require a lot of signal processing which is not practical for real-time application.

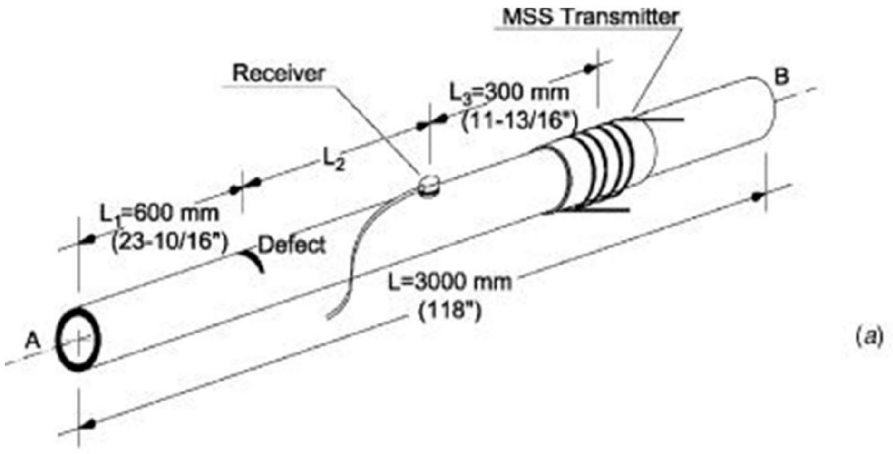


Figure 13: Piervincenzo Rizzo et al Experiment Arrangement.[52]

Recently the Support Vector Machine (SVM) classification algorithm has been used to classify NDT data [53][54][55]. One such work is from C.Wan and A,Mita, the research proposed a system for monitoring the health of underground pipelines [56]. The research focused on acoustically recognizing road cutters as these are a prelude to most construction activities in modern cities. For efficient pipeline monitoring, the proposed system implemented Principal Component Analysis (PCA) to reduce data dimension. Eigenvalues were used as the feature vector for sound recognition. The denoising ability of PCA improved the proposed method in term of robustness to noise interference. One class SVM was used to classify the acquired data. The results of experiment showed that the proposed PCA and SVM based acoustic recognition system were very effective in classifying the acquired data with a low tendency for raising false alarms. Figure 14 shows the flowchart of the proposed method. Work by R. Rajkumar shows that using PCA on LRUT data acquired from

pipeline defect reduce the SVM accuracy thus it is not a good approach for the proposed NDT system [57].

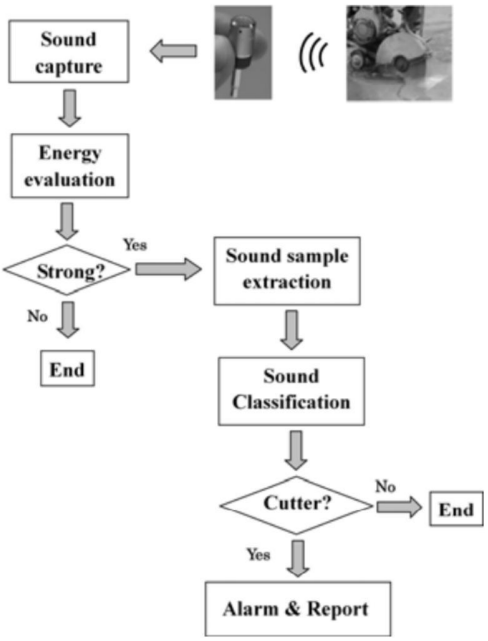


Figure 14: Proposed PCA and SVM based acoustic recognition by C.WAn and A,Mita. [56]

Bernieri et al proposed a methodology to estimate crack shape and dimensions in conductive materials using an integrated Eddy Current and Support Vector Machine [58]. The defects were characterized by integrating suitable measurement hardware with a powerful elaboration procedure using a Support Vector Machine based technique. This proposed system was able to determine defect location and characterize the defect in terms of crack length, height and depth in the specimen with noticeable accuracy. The research also shows the choice of adopting a simulated environment to perform the SVM setup proved to be a powerful solution in order to

investigate all the fault situations in an easier way without losing tin accuracy and robustness. Figure 15 shows the proposed system architecture. However, the objective of this project requires the proposed NDT system to be able to monitor the pipeline continuously, thus it is not feasible to implement Eddy Current as it require a large amount of sensor for a length of pipe.

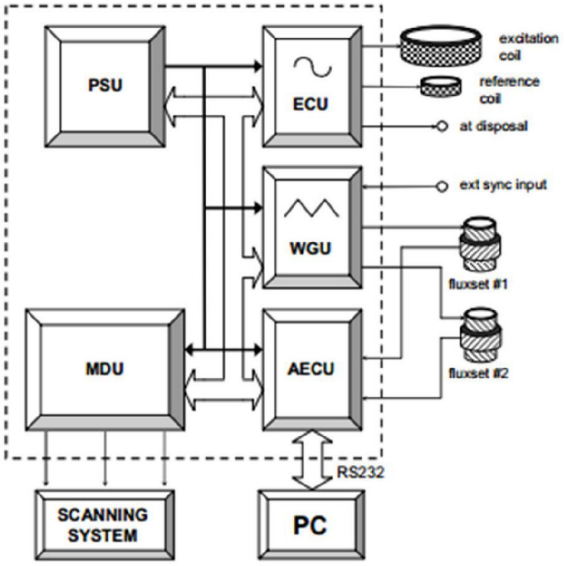


Figure 15: Bernieri et al Proposed System Architecture. [58]

2.5. Unsupervised Learning Algorithms

The use of supervised learning algorithm in NDT has been well-known in recent years. However, the use of unsupervised learning algorithm in this domain is limited. For unsupervised setting, the signals acquired are not labeled to any class member by an expert. Therefore, the acquired signal must be group with similar signals using clustering technique such as Self Organizing Map (SOM) and this is challenging to execute accurately [59][60][61].

2.5.1. Self Organizing Map (SOM)

One of the more influential techniques in NDT uses the self-organizing map to cluster ultrasonic time-of-flight data [62]. SOM is one of the established unsupervised learning algorithm where a lot of research work was done throughout the last 30 years [63]. This particular research uses ultrasonic data from A-scan (Amplitude Modulation Scan) and categorized the signal using SOM. Any new data point is evaluated to the map using cross-correlation to assign the class for the new data.

SOM is an algorithm for mapping from one (typically high-dimensional) space to another (typically low-dimensional) space producing a discretized representation of the input space composed of training samples, called the SOM map. The SOM learns the correct mapping independent of any expert supervision; therefore SOM is an unsupervised learning technique. Like other learning techniques, SOM works in two phase, training and mapping. In the training phase, the SOM generates a map using

input examples while the mapping phase automatically classifies a new input vector [64].

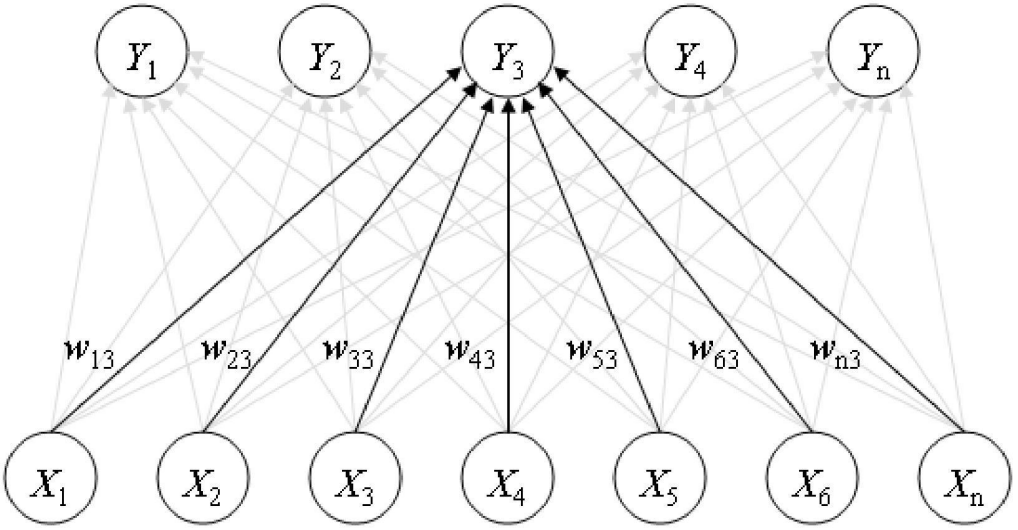


Figure 16: Example self-organizing network with five cluster units, Yi, and seven input units, Xi. The five cluster units are arranged in a linear array [65].

During mapping, there will only be one winning neuron where the weights are closest to the input vector. This is determined by calculating the Euclidean distance between input vector and weight vector. The winning node is selected using Equation (1).

$$c(t) = arg\min_i(\left|\left|x(t) - w_i(t)\right|\right|_2) \qquad \text{- Equation (1)}$$

Where $w_i(t)$ is the weight vector of node i at time t. $||.||_2$ denotes the L^2-norm or n-dimensional Euclidean Distance. The weights of all nodes are then updates using Equation (2) – Equation (4).

$$w_i(t+1) = w_i(t) + \Delta w_i(t) \quad \text{- Equation (2)}$$

$$\Delta w_i(t) = \alpha(t) h_{c,i}(t)[x(t) - w_i(t)] \quad \text{- Equation (3)}$$

$$h_{c,i}(t) = e^{\frac{-d(i,c)^2}{\beta(t)^2}} \quad \text{- Equation (4)}$$

Where,

$h_{c,I}(t)$ is referred to as the neighbourhood function,

$d(i,c)$ is the Euclidean distance from node I to the winning node c in the node grid,

$\alpha(t)$is the learning rate at time t,

and $\beta(t)$ is the neighbourhood size at the time t.

Finally, the learning rate, α and neighbourhood size, β are decreased in accordance with the annealing scheme. The annealing scheme relies on the time step number t and not the actual fitness of the network. One of the possible annealing schemes is given by Equation (5) and Equation (6) [66][67].

$$\alpha(t+1) = \alpha(t)\delta_\alpha, 0 < \delta_\alpha < 1 \quad \text{–Equation (5)}$$

$$\beta(t+1) = \beta(t)\delta_\beta, 0 < \delta_\beta < 1 \quad \text{–Equation (6)}$$

Where, δ_α and δ_β are scaling constants determined beforehand.

These steps are repeated until predetermined condition is met, usually when some measurement of error reaches a certain level.

In this project, SOM method is investigated. The result shows the SOM is not able to cluster the training data accurately. Thus a novel approach is proposed and discussed in Chapter 3.

2.6. Statistical Learning Theory

One of the significant contributions of statistical learning theory is that the VC-confidence which is able to capture the generalization error of a model. The key insight is that the sum of the empirical risk and the VC-confidence denotes an upper bound on the expected risk of a model [68].

In order to understand VC-confidence, one must know the definition of VC-dimension and the relation with VC-confidence. VC-dimension is a measure of the complexity of a classifier [69]. It measures how well a binary classifier can model the boundary between the two classes. More complex classifiers have larger VC-dimension [68].

Vapnik proposed an idea to measure the generalization error of a model based on its VC-dimension, Equation (7). It is called the VC-confidence [70].

$$\text{VC} - \text{confidence},\ \nu(l,h,\eta) = \sqrt{\frac{h\left(\log\left(\frac{2l}{h}\right)+1\right)-\log\left(\frac{\eta}{4}\right)}{l}} \qquad \text{- Equation (7)}$$

Where,

l is the size of training data,

h is the VC-dimension of the model class, and,

$\eta$is some small number such that 0 <η< 1.

Notice that VC-confidence,ν is directly proportional to VC-dimension, h. This suggests that a large VC-dimension implies a complex model and also large generalization error. Also take note as the VC-confidence is increased, the size of dataset is decreased. In other words, a bigger the training data used to model a classifier creates a smaller generalization error.

Therefore, VC-confidence can be considered the upper bound on the expected risk of a model.

Given the empirical risk, R_{emp}, the upper bound on the expected loss of the model over the entire underlying data universe can be estimated using Equation (8) [68].

$$R\left[\hat{f}\right] \leq R_{emp}\left[\hat{f}\right] + \nu(l, h_{\hat{f}}, \eta) \qquad \text{- Equation (8)}$$

Where,

$R_{emp}$is the empirical risk,

$h_{\hat{f}}$is the VC-dimension of $\hat{f}$, and,

$l$is the size of training data.

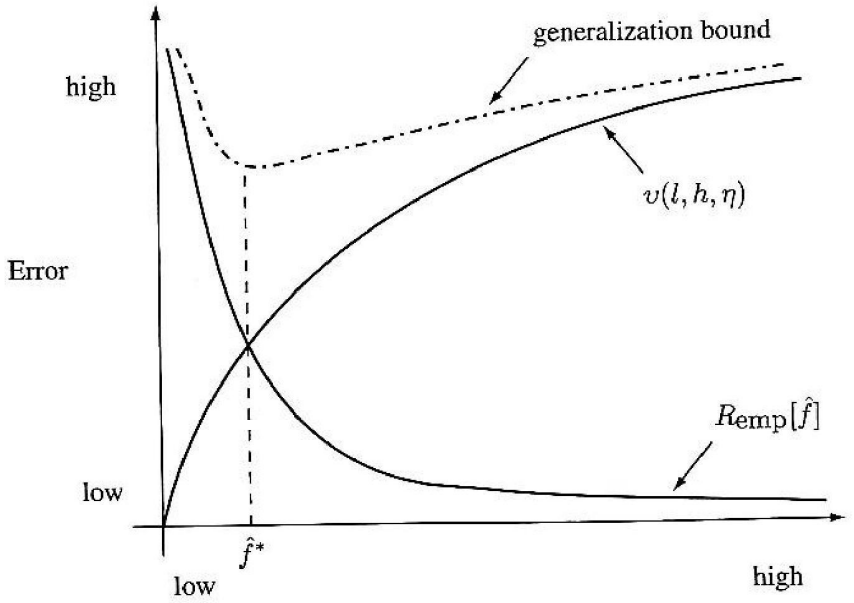


Figure 17: Model Complexity, Relationship between the Empirical Risk and the VC-confidence.

Figure 17 shows the relation between VC-confidence and empirical risk. Notice that model complexity is proportional to VC-confidence. On the other hand complexity of the models is inversely proportional to empirical risk. This indicates complex models will produce more errors on data not contained in the training samples compared to less complex models.

The sum of the VC-confidence and empirical risk represents an envelope of these two curves. This envelope is called the generalization bound. The minimum of the generalization bound is formulated to find optimal model, $\hat{f}^*$, Equation (9).

$$\hat{f}^* = arg\ \min_{\hat{f}\in\hat{F}}\left(R_{emp}[\hat{f}] + \nu(l, h_{\hat{f}}, \eta)\right) \qquad \text{- Equation (9)}$$

Where,

$\hat{F}$is the superclass of all model classes

Equation (9) is used to find a model $\hat{f}^*$ that has the optimal trade-off between generalization error and complexity of a model. However, it is not feasible to compute all model $\hat{F}$as it is infinite. Vapnik suggested that instead computing all $\hat{F}$, the VC-dimension of the individual model subclasses of $\hat{F}$ can act as guide to findthe optimal model, $\hat{f}^*$. Vapnik called it structural risk minimization [68][69].

In order to illustrate the concept of structural risk minimization, an example is shown below [68]. Suppose $\hat{F}$ are linear models with

$$\hat{F}[\gamma_1], \dots, \hat{F}[\gamma_k] \subset \hat{F} \qquad \text{- Equation (10)}$$

and,

$$\hat{F}[\gamma_1] \subset \hat{F}[\gamma_2] \subset \cdots \subset, \hat{F}[\gamma_k]\ if\ h_1 < h_2 < \cdots < h_k \qquad \text{- Equation (11)}$$

Where,

$h_i$is the VC-dimension of the model class $\hat{F}[\gamma_1]$.

Equation (10.14) implies that the margins of the various model classes are also partially ordered, Equation (12).

$$\gamma_k < \cdots < \gamma_2 < \gamma_1 \qquad \text{- Equation (12)}$$

Initially, empirical risk, R_{emp} and VC confidence for model$\hat{F}[\gamma_1]$ are computed using Equation (9). Then, empirical risk and VC confidence for model class $\hat{F}[\gamma_2]$ are computed. The search is terminated as the generaliztion bound of some model $\hat{F}[\gamma_{i+1}]$ is larger than generalization bound of model, $\hat{F}[\gamma_i]$. The process is call structural risk minimization because it uses the structure of the model classes to guide the search.

In short, the theory of structural risk minimization provides two essential concepts for SVM. First, structural risk minimization justify that there is a trade-off between generalization error and complexity of a classifier. Second, structural risk minimization formalizes SVM maximum-margin classifiers. The idea of maximum-margin is to model a classifier with the largest margin and generalizes well.

In this project, structural risk minimization is important because it shows the SVM has a better generalization capability. The generalization capability is important to the proposed system as the defect characteristics are varied.

2.7. Support Vector Machine

Support vector machine (SVM) has been a popular choice for pattern recognition and regression applications [71][72]. It is known to have excellent generalization advantage and good computational efficiency. SVM are based on Vladimir Vapnik's idea of the structural risk minimization detailed in previous section [70]. The SVM finds the model of recognition for of two-class problems (binary classification) by forming a hyperplane that separates the two classes finding the maximum distance to the class points as shown in Figure 18 . These points are called support vector. In many cases, data are not linearly separable in the input space, thus nonlinear transformation is applied [73].

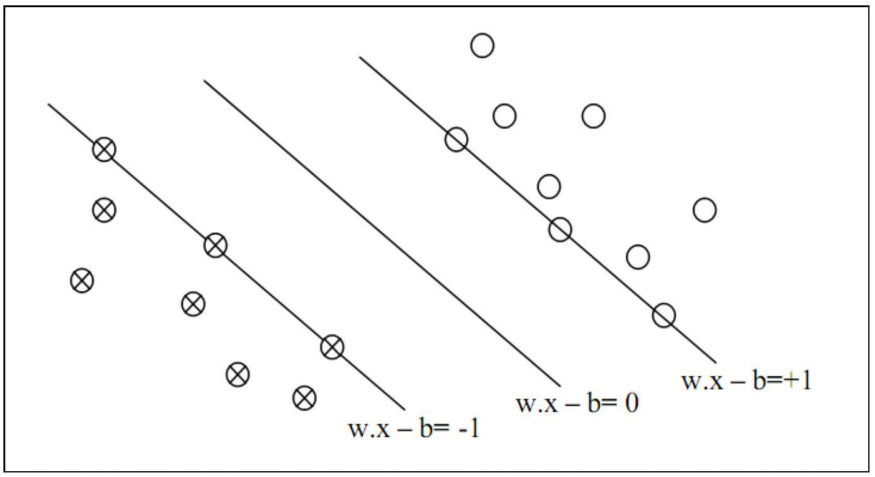


Figure 18: The hyperplane $w.x - b = 0$. [74]

As binary classifiers go, SVMs have been used in a variety of problems such as abnormal human-activity detection [75], image classification [76], and text identification [77], amongst others. As regressors, they have been used in wireless communications [78] and non-linear control systems [79], to name a few.

Mathematical Explanation [68][73][80]:

The optimal hyperplane is defined by Equation (13),

$$\boldsymbol{w} \cdot \boldsymbol{x} + b = 0 \qquad \text{- Equation (13)}$$

Therefore, the maximal margin is define Equation (14),

$$\frac{1}{2} ||\boldsymbol{w}||^2 \qquad \text{- Equation (14)}$$

Within constraint, Equation (15),

$$y_i + (\boldsymbol{x}_i \cdot \boldsymbol{w} + b) \geq 1 \,, \forall \text{ i} \qquad \text{- Equation (15)}$$

By minimizing Equation (14) within constraint Equation (15), the optimal separating hyperplane can be found. After that, a positive slack variable, ξ_i is introduced in the constraint, Equation (15).

Introducing a positive slack variable, ξ_i in Equation (16),

$$y_i + (\boldsymbol{x}_i \cdot \boldsymbol{w} + b) \geq 1 - \xi_i \,, \forall \text{ i} \qquad \text{- Equation (16)}$$

If error happens, the corresponding $\xi_i$exceeds unity, therefore, $[\sum_i \xi_i]$ become an upper bound for the number of classification errors. Therefore, a logical way to

assign an extra cost for errors is to change the objective function Equation (14) to be minimized into:

$$min\{\frac{1}{2}\ ||\boldsymbol{w}||^2 + \ C\ \cdot (\ \sum_i \xi_i\ )\} \qquad \text{- Equation (17)}$$

Where,

C is tuning parameter.

The tuning parameter, C is introduced in Equation (17) offers the user control over the tradeoff by either maximizing the margin or classifying the training set without error. Minimizing Equation (17) with constraint in Equation (16) gives the Generalized Optimal Separating Hyperplane. The equation becomes a Quadratic Programming (QP) problem and it can be solved using the Lagrange multipliers method [68].

$$W(a) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i,j=1}^{n} a_i a_j y_i y_j\ (x_i^T x_j) \qquad \text{- Equation (18)}$$

Where,

α is the LaGrange multiplier

The Quadratic Programming (QP) problem can be solved by finding the LaGrange multipliers, α_i, that maximizes the objective function in Equation 5 [68][73].

2.7.1. Kernel Trick

The algorithm can be generalized to non-linear classification by mapping the input data into a high-dimensional feature space using mapping function, Φ. Modeling a separating hyperplane in the feature space leads to a non-linear decision boundary in the input space. By introducing a kernel function, K in Equation (19), calculation of dot products in a high-dimensional space is avoided [68][73].

$$K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j) \qquad \text{- Equation (19)}$$

Therefore,

$$W(a) = \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i=1,j=1}^{n} a_i a_j y_i y_j \, K(x_i, x_j) \qquad \text{- Equation (20)}$$

Subject to Equation (21),

$$C \geq a_i \geq 0, \sum_{i=1}^{n} a_i y_i = 0 \qquad \text{- Equation (21)}$$

Testing equation, Equation (22) classify new data to class 1 if $i \geq 0$, and to class 2 if $i < 0$ [68].

$$i_F(x) = sign\left[\sum_{i=1}^{l} y_i a_i K(\boldsymbol{x}, x_i) + b\right] \qquad \text{- Equation (22)}$$

Figure 19 illustrates the SVM data flow.

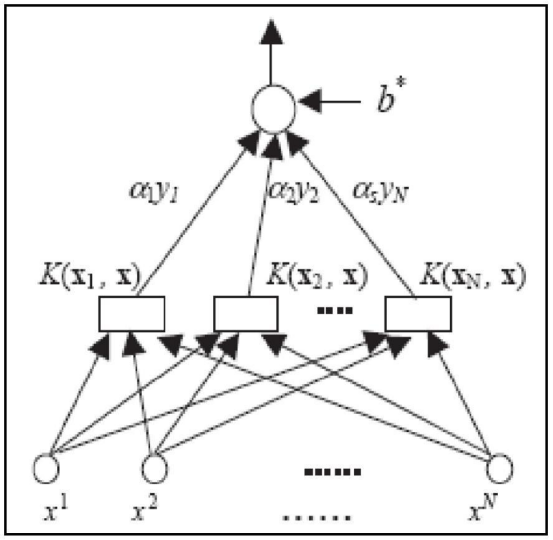


Figure 19: SVM data flow diagram. [80]

The summation is performed only on the SVs not on all training data, because only for SVs do the Lagrange multipliers are not zero. Figure 19 shows the flow from input data point to the final decision value. One of the most common problems in the machine learning field is the classification problem. As SVMs gained wider acceptance in the academic and industrial communities during the late 90s, there arose a need for incremental training algorithms, which would allow application of SVMs in setting where the training set was non-constant, for example, in adaptive control systems or equalization of time-varying communications channels.

In this project, the SVM was used primarily because it offers better result See Appendix A.

2.8. Quadratic Programming

Quadratic programming (QP) is a special type of mathematical optimization problem. It is the problem of optimizing (minimizing or maximizing) a quadratic function of several variables subject to linear constraints on these variables. As it has many applications, quadratic programming is often viewed as a discipline in and of itself [82].

General QP Problem Statement

The general quadratic program can be written as Equation (23) [83][84],

$$minimize\ f(x) = \boldsymbol{c}^T\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q} \qquad \text{- Equation (23)}$$

Subject to Equation (24) and Equation (25),

$$\boldsymbol{Ax} \leq b \qquad \text{- Equation (24)}$$

$$\boldsymbol{x} \leq 0 \qquad \text{- Equation (25)}$$

Where,

$\boldsymbol{c}$ is an n-dimensional row vector describing the coefficients of the linear terms in the objective function, and,

$\mathbf{Q}$ is an $(n \times n)$ symmetric matrix describing the coefficients of the quadratic terms. If a constant term exists it is dropped from the model.

As in linear programming, the decision variables are denoted by the *n*-dimensional column vector **x**, and the constraints are defined by an ($m \times n$) **A** matrix and an *m*-dimensional column vector **b** of right-hand-side coefficients.

2.8.1. Quadratic Programming in Support Vector Machine

In Support Vector Machines (SVMs), QP is used to solved an objective function, Equation (20)

$$W(a) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1,j=1}^{n} a_i a_j y_i y_j \, K(x_i, x_j) \qquad \text{-Equation (20)}$$

Subject to Equation (21),

$$C \geq a_i \geq 0, \sum_{i=1}^{n} a_i y_i = 0 \qquad \text{- Equation (21)}$$

This objective function is solved to find optimal alphas, the weight for a vector which is a support vector.

Typically there are three algorithms for solving quadratic programs, which are the interior point method, active set method and working set method. However, each of these algorithms have their own advantages and disadvantages [10].

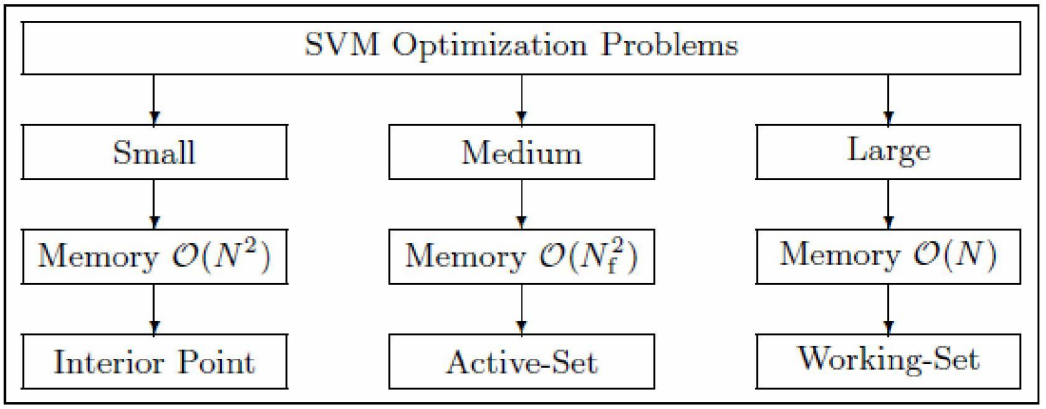


Figure 20: Comparison between three different quadratic programming solvers. [10]

N is the number of vectors.

N_f is the number of free support vectors that is typically only a small fraction of the data set.

The number of N in SVM optimization is a critical point for the method selection. As N may be very large in practical applications, common SVM software packages rely on working-set methods [71][85].

However, this method may show weak results especially if the C parameter is chosen to have high value [86]. Therefore, in this project, the active set method is used as the base algorithm for online SVM.

2.9. Genetic Algorithm

Genetic algorithms (GAs) were invented by John Holland and his colleagues to mimic the evolutionary process of biological organisms. This led to Holland's book titled "Adaption in Natural and Artificial System" published in 1975 [87][88]. GA is a probabilistic search algorithm which can be applied to a variety of combinational optimization problem. In an evolutionary process, natural populations evolve according to principles of natural selection and "survival of the fittest" [89]. Individuals that are more successful in adapting to the environment will have a better chance of surviving and reproducing, and individual that are less fit will be eliminated. This means that the genes from highly fit individuals will spread to increasing number of individuals in each successive generation. Thus, the later individuals are more adapted to the environment [90].

A GA simulates natural processes by generating initial population of individuals and implementing genetic operators in each generation. Each individual in the population is encoded into a chromosome which represents a possible solution for a given problem. Next, the possible solution is evaluated with respect to a given objective function. Highly fit solutions are reproduced by exchanging a part of the genetic information with other highly fit solution; the process is called crossover. This produces new offspring solutions which share characteristics taken from two or more fit parent solutions. Another procedures called mutation is applied after crossover by altering some genes in the solutions. The processes are repeated until a satisfactory solution is found [90][91]. Figure 21 shows the illustration for crossover procedure

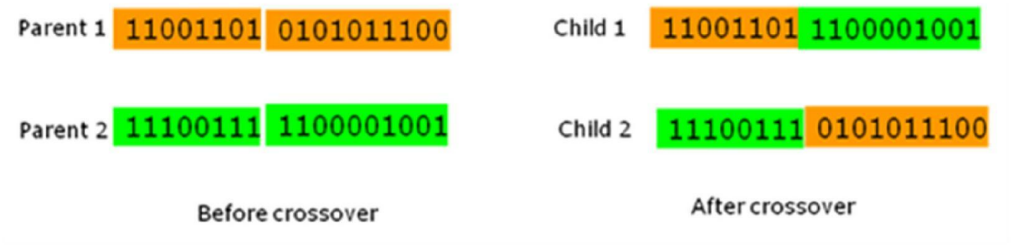


Figure 21: Illustration of GA crossover procedure. [92]

Recently, there are many implementations of Genetic Algorithm in variety domains. Arkandan et al proposed a technique to determine the nature of a crack on the surface of a region using nondestructive testing (NDT) and inverse problem methodology [93]. A GA is implemented and it involved a global search to avoid local minima to. The technique is applied to solve the inverse problem of identifying the position, shape and the orientation of a surface crack. A fine tuning algorithm is combined with the GA to reach the optimum solution.

Another famous used of GA is to find the best machine learning parameters. Kim et al implemented GA to improve Support Vector Machines (SVM) based Intrusion Detection System (IDS). The proposed solution is able to find the optimal parameters for SVM and also “optimal feature set” among the whole feature set. In addition it minimize the number of features that SVM classifier and maximize the detection rates of IDS. Figure 22 shows the flowchart of that proposed method.

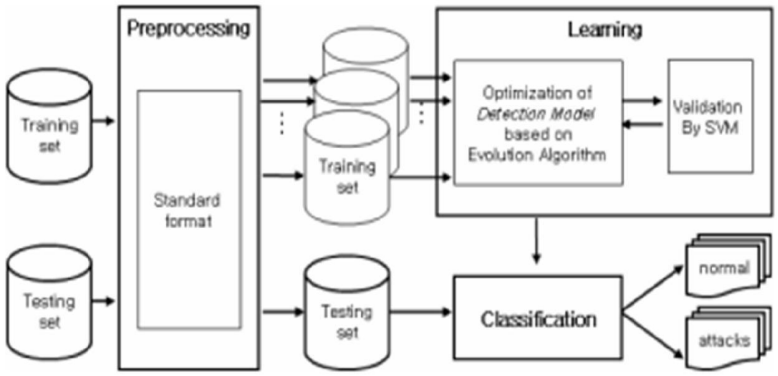


Figure 22: Flowchart of the proposed method for Intrusion Detection System. [94]

In this project, the GA replaced SMO as the QP Solver. This benefits the system as GA is a more versatile approach that can work for any amount of data compared to SMO. However, implementing GA to solve SVM QP problem straightforwardly is not practical as the SVM objective function, Equation 20 may have a huge number of variables. Thus, a strategy to overcome the problem is proposed and discussed in Chapter 3.

CHAPTER 3: METHODOLOGY

This chapter consists of two parts where the first part details the steps in the design of the experimental rig and hardware for generating and acquiring the ultrasonic guided wave signals. The second part of the chapter presents the online Support Vectors Machine , the unsupervised training techniques used here, the genetic algorithm as quadratic programming solver and the multi class classification formation that are used to process the signals

3.1. Experimental Setup

The LRUT test rig system used in this research is designed, constructed and simulated using state of the art laboratory equipment and components. Figure 23 shows the block diagram of the LRUT system [5][57]. The piezoelectric transducers and inflatable collar are purchased from Plant Integrity Ltd. The transducers are highly specialized and are capable of specifically exciting torsional guided waves by manipulating their orientation. The test system consists of 16 piezoelectric transducers, arranged axially in a ring. Tone burst signals are used to excite the transducers and the low bandwidth nature of these signals makes the generation of torsional mode much easier [95].

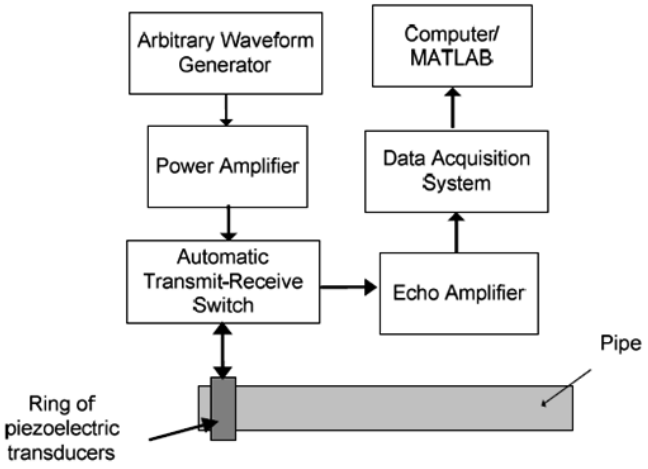


Figure 23: Block diagram of the LRUT system.

Five cycle tone burst signals are created using Agilents 33220A arbitrary waveform generator. The waveforms are created using the waveform editor software and uploaded onto the waveform generator non-volatile memory. The burst frequency is chosen as 10Hz which is the recommended maximum rating specified by the manufacturers of the transducers (Teletest®) [40]. The transducers also require a high voltage excitation signal which was designed specifically to create waves of sufficient amplitude in order to propagate long distances. Finally, the tone burst signals are amplified to a voltage of 200 V peak-to-peak using a power amplifier which was designed and fabricated in house. Figure 24 shows the piezoelectric transducers and inflatable collar.

Figure 24: The piezoelectric transducers and inflatable collar.

3.2. Defect Simulation

An LRUT system was developed and used to test a 1.5m section of a carbon steel pipe which is 140mm in diameter and 5mm in wall thickness .The frequency of the tone burst signals required to excite the transducers for this pipe is experimentally determined to the 20 kHz as this gives the highest back wall echo signal strength [57]. Ideally, the best corrosion simulation approach would gradually corrode a section of the pipe over a long period and periodic measurement could be obtained [96]. However, the approach would require considerable effort, first in fabricating a corrosion simulation rig, and secondly to simulate corrosion at sufficiently high rates. Therefore, in order to prove the concept, a standard corrosion defect (full circumferential defect) will be performed at different depths. Several measurements

will be taken at each depth and all signals will be ordered sequentially in order to simulate a natural corrosion process.

A lathe machine is used to create a complete circumferential corrosion defect with 3mm axial length. 1mm, 2mm, 3mm and 4mm depth circumferential were created. The LRUT system is placed approximately, and measurements are taken after each depth was machined. Figure 25 shows the location of defects on the pipe section. Figure 26 shows the actual image of pipeline and defect.

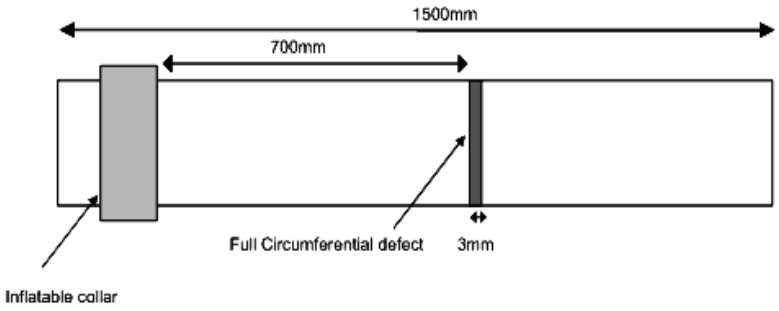


Figure 25: Full circumferential corrosion defect position.

The LRUT system is unique to the research presented in this thesis.

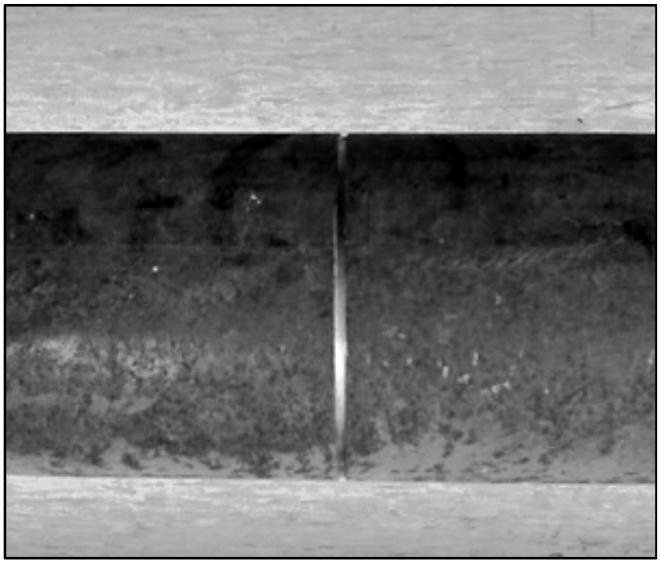


Figure 26: Actual Image of Pipeline and Defect

The guided wave signals are sampled at a rate of 100 kHz and stored for processing. All data processing, including data acquisition, are performed using a MATLAB® program, Appendix C. Figure 27 shows the output waveform of the transducer responses from the initial point of excitation to the arrival of the Back Wall Echo (BWE). The data sampled from the transducer signals must first be gated to remove all redundant information as well as significantly reducing data size. The critical information of the pipe's condition is stored in the time frame between the excitation of the tone burst pulse and the arrival of the BWE.

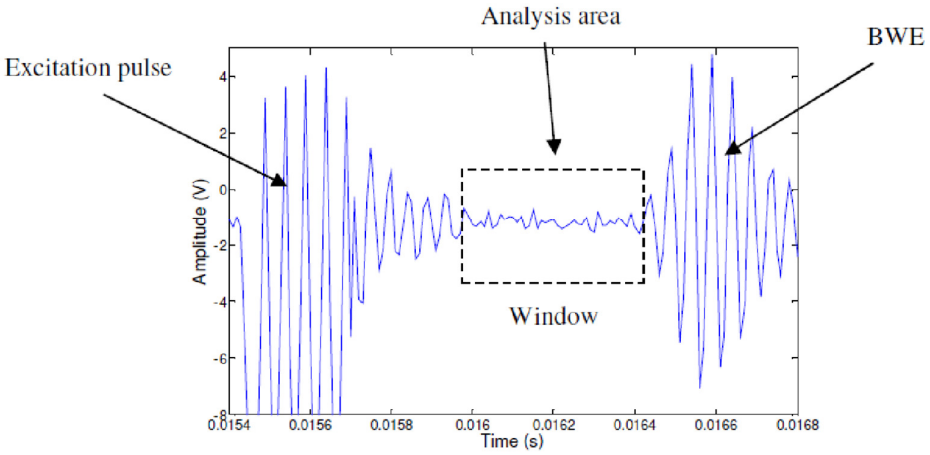


Figure 27: Sampled transducer signal showing position of initial pulse, BWE and critical analysis area.

3.3. Unsupervised Training

One of the objectives of this research is to implement a semi autonomous NDT system. The system will run with minimal human supervision. However, in the standard SVM training phase, the training samples are associated with their label, typically tagged manually by human experts. Therefore, a method using all-to-all distance estimation is proposed. The proposed method will label data automatically and will be used to train the SVM.

In the first step of the proposed methodology, the distances of a sample to all other samples are computed. This step is repeated for each sample data. Once all the distances are calculated, the information is used to form a distance matrix. Next, the smallest distance pair is tagged with same label, and the mean of the samples with the same label is calculated. If both the samples in the pair have no label, both samples are tagged with a new label. If the number of label has reached to a specified number,

both samples are labeled with the closest label sample mean. This step is repeated until all samples are labeled. Figure 28 shows the flow chart of the proposed method.

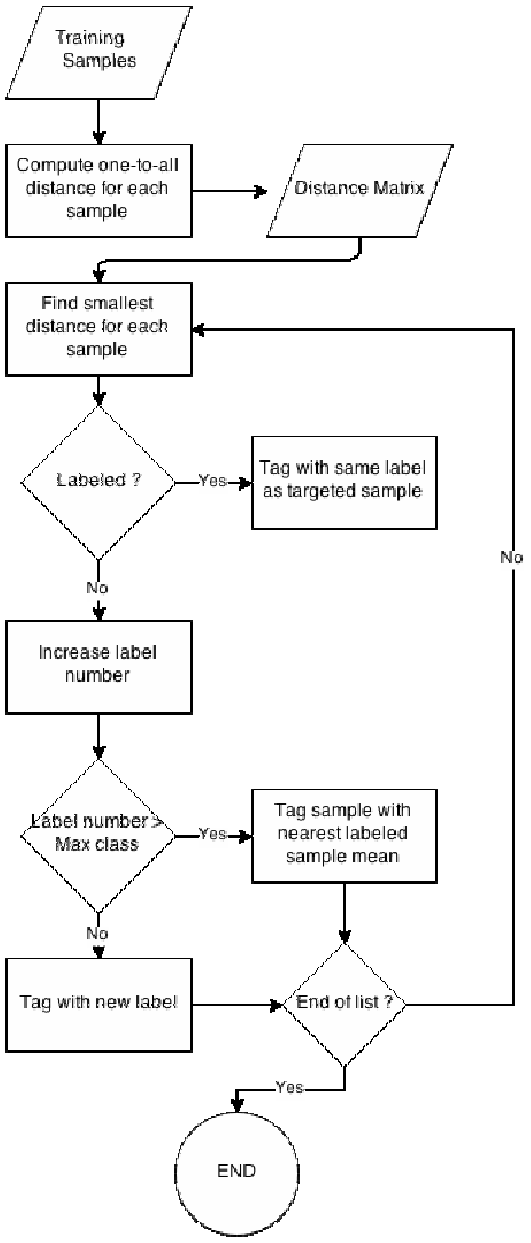


Figure 28: Flow Chart of Proposed All-to-all Clustering Method.

As discussed in the previous paragraph, the first proposed method requires a human expert to specify the number of clusters, thus it is not a fully automate method. A method for number of cluster estimation is therefore proposed. This method is an extension of the previous method. Contrary to previous method, this method increases the number of labels until all the samples are tagged with a label. Then, the mean for all clusters are calculated and are fed back to the algorithm as new samples. These steps are repeated until the number of cluster fall within required number. Figure 29 shows the flow chart of the proposed method.

This method is one original contribution of this thesis.

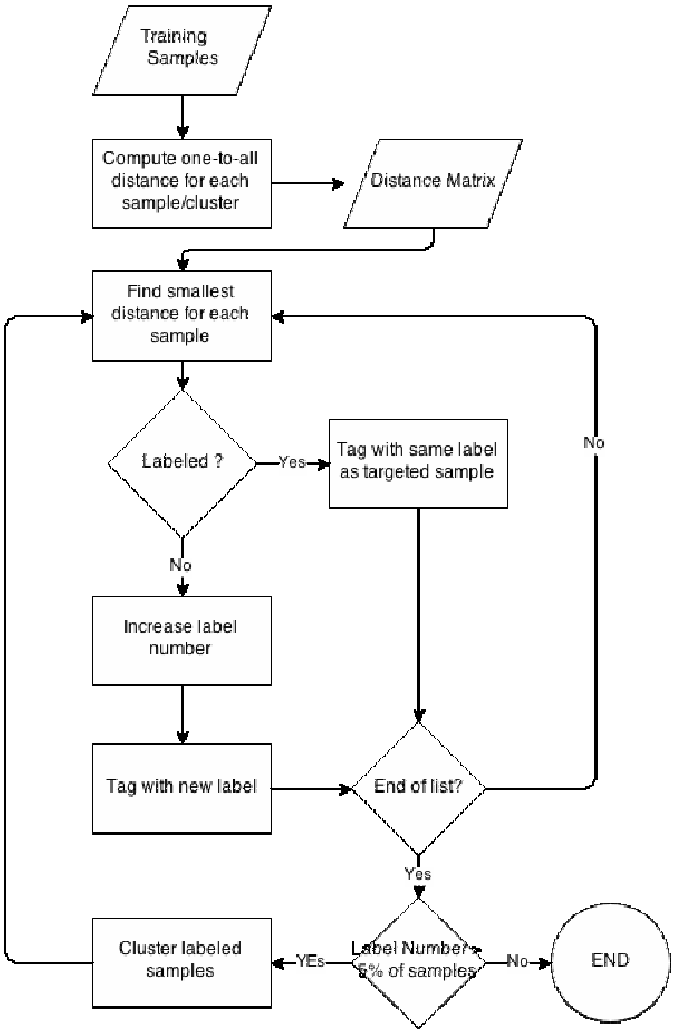


Figure 29: Flow Chart of Proposed All-to-all Clustering Method with Automatic Number of Cluster Estimation.

To demonstrate how this technique works, a 4 data points clustering problem is presented. Data points are set to $D = \{\bar{x}_a, \bar{x}_b, \bar{x}_c, \bar{x}_d, \bar{x}_e\}$, and estimated belongs to 2 cluster $\mathbb{R}^2 \times \{1,2\}$. Data points are plotted in the Figure 30.

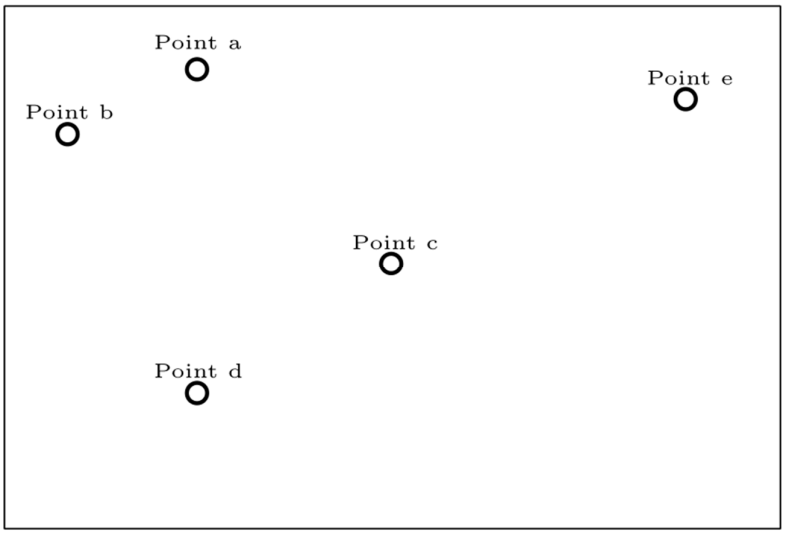


Figure 30: 2D Data points.

Then the distances of data point *a* to all other samples are computed as shown in Figure 31.

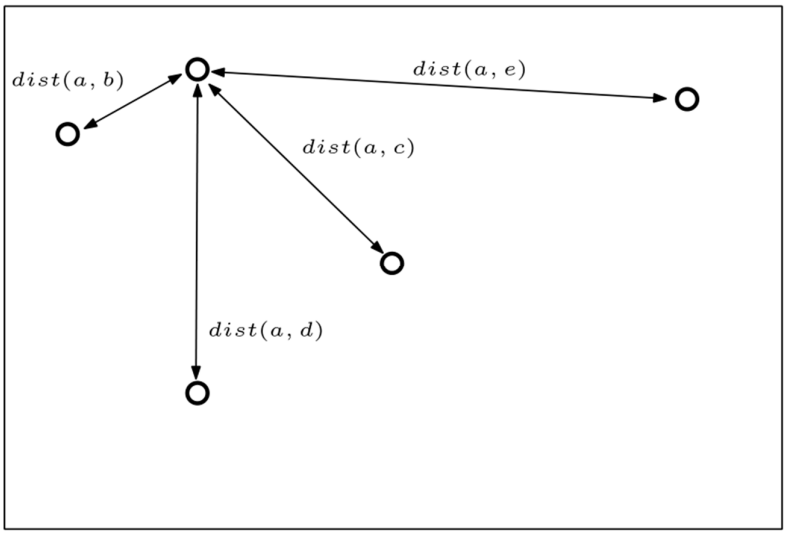


Figure 31: Distance of data points to point *a*.

Previous step is repeated for other datapoints *(point b, point c, point d and point e).* Once all the distances are computed, a distance matrix is constructed as shown in Figure 32.

	Point *a*	**Point *b***	**Point c**	**Point *d***	**Point *e***
Point *a*					
Point *b*	*dist(a,b)*				
Point c	*dist(a,c)*	*dist(b,c)*			
Point *d*	*dist(a,d)*	*dist(b,d)*	*dist(c,d)*		
Point *e*	*dist(a,e)*	*dist(b,e)*	*dist(c,e)*	*dist(d,e)*	

Figure 32: Distance Matrix.

Note that distance for repeated pair is not calculated for example distance *point a to point b* and *point b to point a* (grey cell) as the distances are equal. This will reduce the the computation cost. The total number of distances calculated can be determined by L(L-1)/2 where L is the number of data points. The complete distance matrix is shown in Figure 33.

	Point *a*	**Point *b***	**Point c**	**Point *d***	**Point *e***
Point *a*					
Point *b*	*0.2*				
Point c	*1.2*	*1.4*			
Point *d*	*1.5*	*1.5*	*0.7*		
Point *e*	*2.3*	*2.9*	*2.1*	*3.1*	

Figure 33: Complete Distance Matrix.

In this example, the distance between *point a* and *point b* is the smallest which is 0.2. Next is the distance between *point c* and *point d* which is 0.7. As for now, the class number has reach estimated number, 2. Thus, the mean for all class is computed. Next, distance between *point e* and each of the class mean is computed. Point e belong to *Class 2* as the distance between *point e* and *Class 2* mean is the smallest. Figure 34 shows the distance between *point e* to *Class 1* and *Class 2* means.

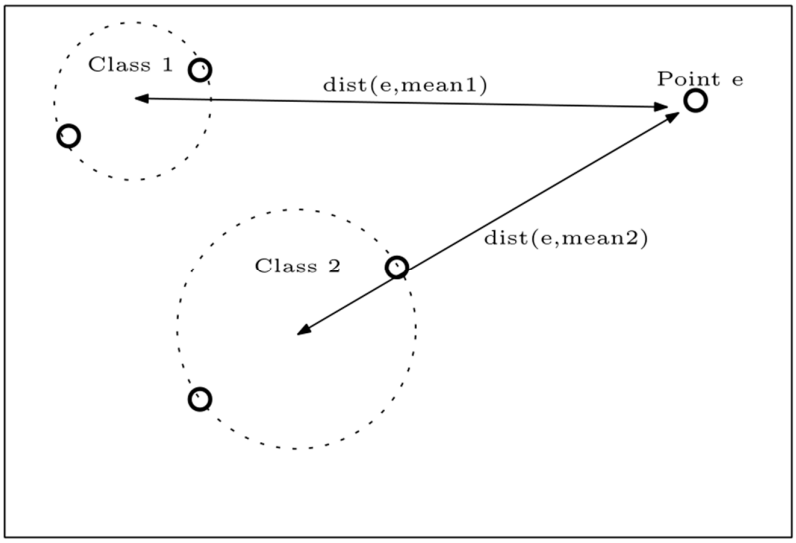


Figure 34: Distance between *point e* to *Class 1* and *Class 2* means.

There are many distance measurement formula available. Although the clustering technique is original but in order to get the best result from the dataset, five formulas are investigated which are, Euclidean distance, Hamming distance, city-block metric, and correlation distance.

1. Euclidean distance

$$d_{st}^2 = (x_s - y_t)(x_s - y_t)'$$

- Equation (22)

2. Hamming distance

$$d_{st} = (\#(x_{sj} \neq y_{tj})/n)$$

- Equation (23)

3. City block metric

$$d_{st} = \sum_{j=1}^{n} \left| x_{sj} - y_{tj} \right|$$

- Equation (24)

4. Correlation distance

$$d_{st} = 1 - \frac{\left(x_s - \bar{x}_s\right)\left(y_t - \bar{y}_t\right)'}{\sqrt{\left(x_s - \bar{x}_s\right)\left(x_s - \bar{x}_s\right)'}\sqrt{\left(y_t - \bar{y}_t\right)\left(y_t - \bar{y}_t\right)'}}$$

- Equation (25)

The result for the unsupervised training technique using each of the distance measurement is presented in Chapter 4.

3.4. Online Support Vector Machine for real time processing

The incremental SVM classifier is an important component of the proposed NDT system. Due to the fact that the proposed system works continuously, the LRUT sensor will produce a tremendous amount of data. Conventional SVM works in batch mode where the algorithm needs time to obtain certain amount of data before a model can be created. In addition, batch mode requires all previous data to be available for creating a model. For example, in the situation which the system has been setup for one year, the system will keep all data from the first day to the current day in order to create a classification model. Due to this reason, a large amount of memory is

required for conventional SVM in real time system, hence it is not feasible for deployment in real time system that need to predict failure. On the other hand, incremental SVM only keep essential data, and unnecessary data are removed. This characteristic of incremental SVM will keep the memory and computation requirements at a minimal level.

Although the total data received at the end of the process is the same, sequence data is supplied continuously to the SVM and computed immediately. Therefore, the active-set technique will be an appropriate method to solve the ill-posed and high precision problem within an acceptable duration. As shown in Figure 20 there are others option of optimizer that may suit the problem. Interior Point Method may offer higher accuracy in processing small data sets [97]. However, it may cost expensive computational time for larger data sets [27]. The Active Set Method can significantly save time cost for medium data sets by introducing the warm start strategy. At the start, the active incremental algorithm will check if it is at initial stage or at incremental stage. At initial stage, the algorithm will buffer the data until it gets the first 4 data pairs with different labels. Then, initial SVs are determined and a hyperplane is constructed. At incremental stage, f classification value for each new data acquired is computed.

$$f(x) = \sum_{i=1}^{n}(y_i\,\alpha_i K(\boldsymbol{x}, x_i)) + b \qquad \text{- Equation (23)}$$

Where,

y is Support Vector decision value or label,

α is weight or Lagrange multiplier,

K is the kernel.

$|f|$ defines the distance between the data point and the hyperplane. The larger the value of $|f|$, the greater the distance of the data from the hyperplane. R is introduced as a new parameter which controls the amount of new data to be included to the SVM model. If the value of $|f|$ is less than the value of R, a new hyperplane is constructed using previous SVs and new data are acquired. Figure 35 shows the flow chart of the proposed incremental SVM.

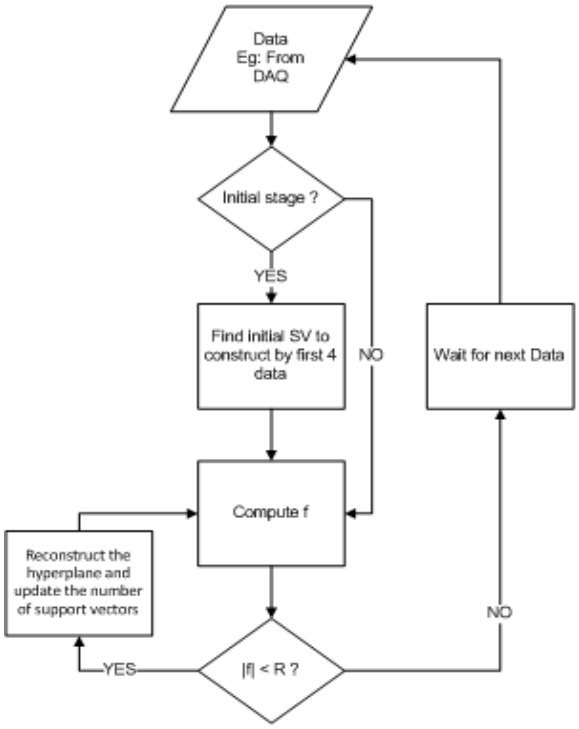


Figure 35: Flowchart of the Proposed Incremental SVM

The algorithm of the proposed system starts by acquiring data from a DAQ (Data Acquisition) module. If the system is at the initial stage, the algorithm will acquire more data until two pairs of different class samples are acquired. Then, the algorithm solves the SVM objective function; constructs the optimal separating hyperplane. Next, the f value is computed for each sample. At the initial stage, all samples should have high $|f|$ values and higher than R. Note that, all samples become Support Vector for the initial hyperplane. Finally, the algorithm waits for the next data from the DAQ. The above description presents a complete cycle of the initial stage.

In the following stage, the f value for all acquired data are computed. If the $|f|$ value is less than the R value, the data is discarded. If $|f|$ value is higher or equal to R value, the new sample and the existing SVs are used to construct a new hyperplane. The algorithm works in an infinite loop conforming to the proposed NDT system specification. See Figure 35 for the flowchart of the algorithm of the proposed Incremental SVM. This is another original contribution of this project.

Each time a hyperplane is constructed, the algorithm solves the QP problem. Conventional SVM uses the QP solver for a large amount of data but only once per task. On the other hand, incremental SVM uses QP solver on a small amount of data but persistently. Accordingly, QP for both applications have different criteria and selecting a suitable QP solver is crucial for incremental SVM.

Appendix B shows the proposed method updates the optimal hyperplane for each data captured.

The training set will be acquired sequentially (one by one) by the SVM. In contrast, the conventional batch SVM needs to buffer the sampled data until all are captured. Conventional SVM will need to retrain all the data at any point in time so that the size of the processed data set is increased. Thus, batch data implementation is less applicable due to its expensive computational cost. In other words, conventional SVM needs to retrain all the data at every point of time while receiving new input data.

3.5. Genetic Algorithm (GA) as Quadratic Programming Solver

In this research, GA is used to optimize the SVM objective function Equation (20), which is a Quadratic Programming (QP) problem, to find the α.

$$W(a) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1,j=1}^{n} a_i a_j y_i y_j \, K(x_i, x_j) \qquad \text{- Equation (20)}$$

Subject to Equation (21),

$$C \geq a_i \geq 0, \sum_{i=1}^{n} a_i y_i = 0 \qquad \text{- Equation (21)}$$

Equation (20) will be the GA fitness functions and Equation (21) are the constraints. For a good SVM model, only a few data points become support vectors (SVs) [98][99]. It is a condition for a good generalization classification model. SVM selects data points which have the greater α values to become SV. Other data points

which are not selected are outliers. Support vectors are data points with non zero α's. Figure 36 shows SVs and outlier vectors.

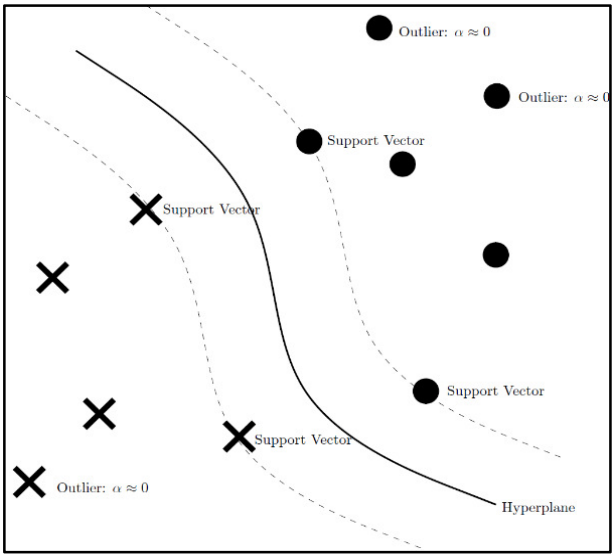


Figure 36: Support Vectors and Outlier Vectors.

The classification problem involves calculating many data points; α is introduced for each data point. Thus, a GA usually takes numerous iterations to find the optimized α for the fitness function.

GA-SVM strategy manipulates SVM conditions by introducing an additional stopping criterion. The new stopping criteria is a point where most of the α lean towards 0. Those α will be removed from the GA constraint. This strategy will reduce the number of iterations as the number of α has been reduced. This is another original contribution of this project. Figure 37 shows the proposed strategy.

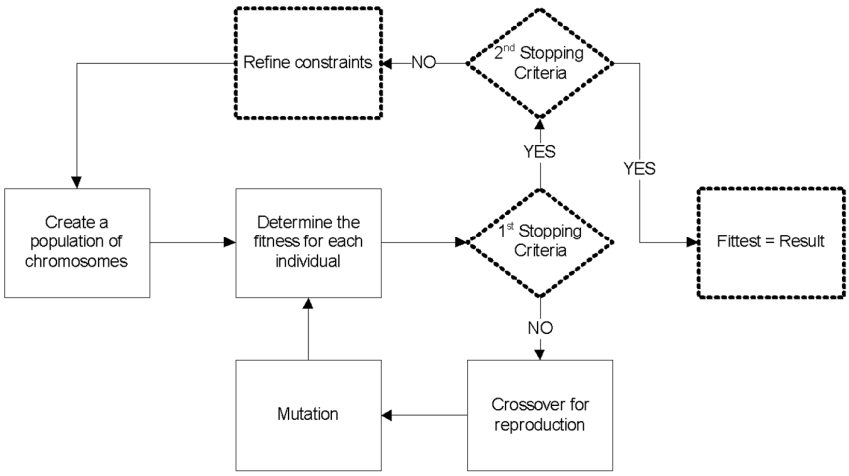


Figure 37: Flowchart of Proposed Strategy for GA as QP solver in SVM.

3.6. Multi-class formation

SVM was originally designed for binary classification problems. However, this project deals with classifying pipeline defect into more than two classes. This part of the thesis details the methodology used in the project to perform multiclass classification.

For this project, one-against-one approach is used as it gives good performances corresponding to the current literature search. For a given multi class problem, the training set is:

$$D = \{(\bar{x}_{1,}y_1), (\bar{x}_{2,}y_2), \ldots, (\bar{x}_{l,}y_l)\}, \mathbb{R}^n \times \{1,2,\ldots,M\} \quad \text{– Equation (24)}$$

M will denote the number of classes. Therefore, M(M-1)/2 binary classifiers are constructed. One classifier for each possible pair of classes. Let $h^{\{p,q\}}$ denote the classifier that classifies class p and class q with $p \neq q$ and {p,q} subset {1,2...M). The following two-class classification problem is solved.

$$h^{p,q}(\bar{x}) = \sum_{i=1}^{|D^{p,q}|} y_i \alpha_i^{p,q} k(\bar{x_i}, \bar{x}) - b^{p,q}$$

– Equation (25)

where,

$D^{p,q} = D^p \, U \, D^q$,

$D^q = \{(\bar{x}, y) | (\bar{x}, y) \in D \wedge y = p\}$

$D^q = \{(\bar{x}, y) | (\bar{x}, y) \in D \wedge y = q\}$

The training set $D^{p,q}$ is the union of two sets which are D^p consists of all the data points in D with class p and the set D^q consists of all the data points in D with class q. Next, all the constructed classifier $h^{\{p,q\}}$ is used to test an unknown point, keeping track how many time the point was assigned to what class label. The class that has the highest count is considered the class for the test point. However, this voting approach has a possibility of a tie. If there is a tie, actual values of the confidence level of the test point for each classifier is interpret. The winner of the tied label class is considered to have the largest sum of confidence values. There are others techniques available for SVM multi-class problems. Hsu and Lin give a comprehensive comparison and concluded that one-against-one is a competitive technique [100].

To demonstrate how this approach work, a 3 class problem is presented. The training samples are set to

$$D = \{(\bar{x}_1, y_1), (\bar{x}_2, y_2), \ldots, (\bar{x}_l, y_l)\}, \mathbb{R}^2 \times \{1,2,3\} \quad \text{– Equation (26)}$$

l is set to 15 (l = 15), as shown in Figure 38.

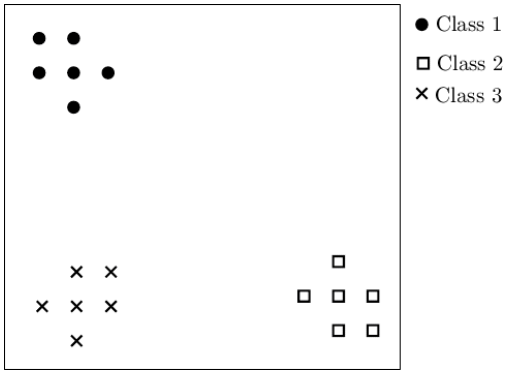


Figure 38: 15 Data-points with 3 Different Classes.

First, ($D^1 \cup D^2$) samples are trained and produce the classifier $h^{\{1,2\}}$ as shown in the Figure 39.

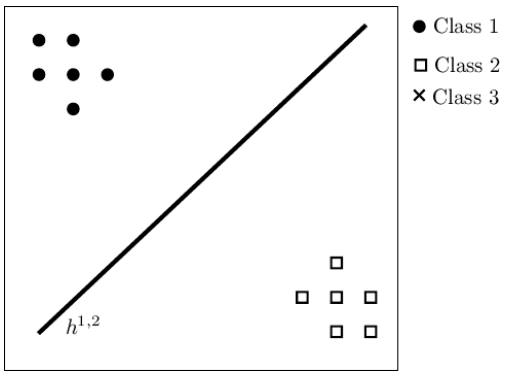


Figure 39: Classifier h{1,2}.

Consequently ($D^1 \cup D^3$) and ($D^2 \cup D^3$) samples are trained to produce the classifier $h^{\{1,3\}}$ and $h^{\{2,3\}}$ as shown in Figure 40 and Figure 41.

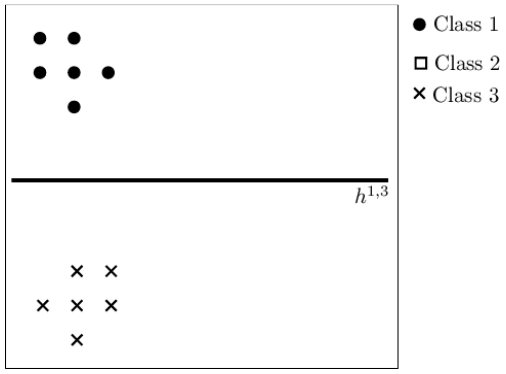


Figure 40: Classifier h{1,3}.

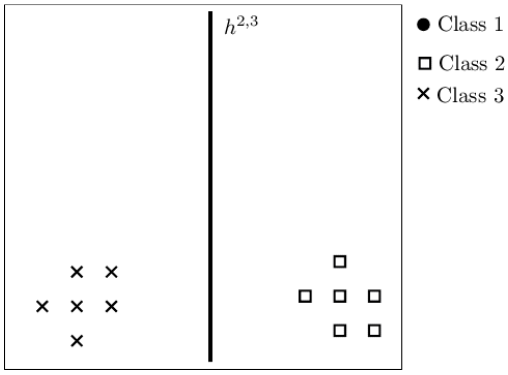


Figure 41: Classifier h{2,3}.

Test data, sample *z* is introduced and tested with all classifiers as shown in Figure 42.

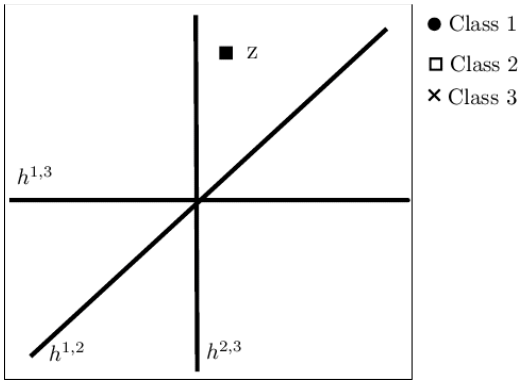


Figure 42: Sample *Z,* Classifiers h{1,2}, h{1,3}, h{2,3}.

Figure 42 shows the result of sample Z on each of the classifier.

Classifier	$h^{1,2}$	$h^{1,3}$	$h^{2,3}$
Result	Class 1	Class 1	Class2

Table 3: Result of sample *Z* on each of the classifiers.

The first two classifiers classified it as *Class 1*, only one classifier classified it to *Class 2* and none to *Class 3*. Thus, using the voting approach, sample Z belongs to *Class 1*.

CHAPTER 4: RESULTS AND DISCUSSIONS

This chapter presents results obtained using the experimental rig and software experiments on the machine learning technique covered in this project. The chapter begins with the outputs obtained from the LRUT system on the experimental rig and the output obtained from the signal processing technique. The next set of results illustrates the output of the proposed unsupervised training technique. Next, the proposed online Support Vector Machine results are presented. Lastly, the final set of results from the multi-class classification is presented.

4.1. The Experimental Setup

The experimental rig is implemented according to the designs specified in Chapter 3 as shown in Figure 43 . The LRUT excitation signal is set to have frequency of 20 kHz, an amplitude of 180 Vpp (Voltage Peak-to-peak) and pulse repetition frequency (PRF) of 10 Hz.

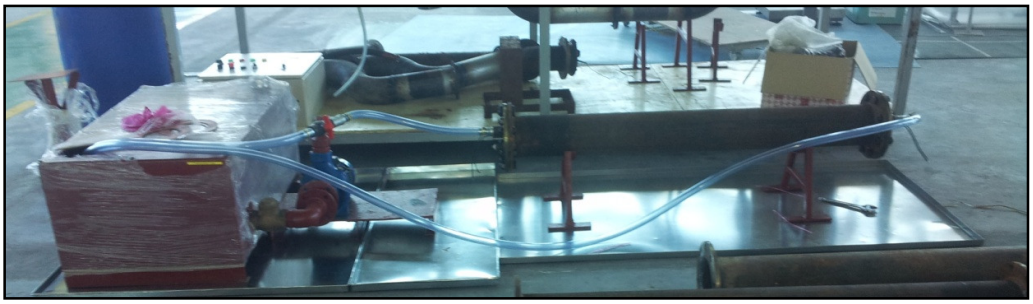


Figure 43: Experimental Rig.

Figure 44 shows the raw signal acquired from the LRUT system. Each excitation pulse is characterized by the huge pulse (-10 V to 5 V) and reappear every 0.1 seconds. This verifies that the PRF is at 10 Hz. ($f = \frac{1}{t} = \frac{1}{0.1} hz$)

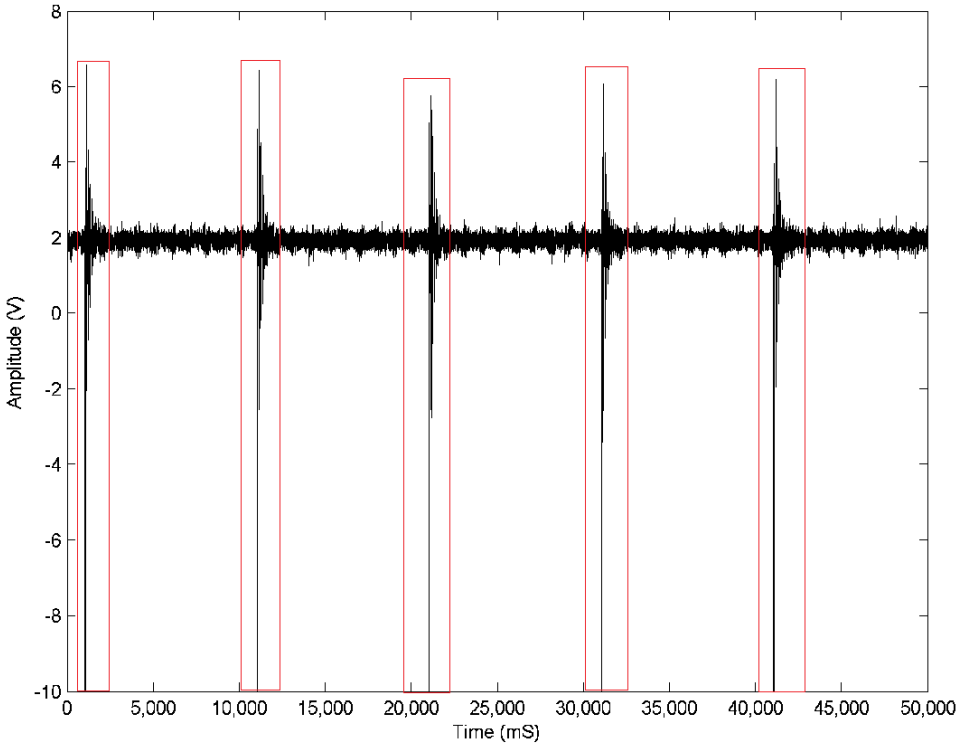


Figure 44: Sampled raw signal acquired form LRUT.

A segment of the raw signal which has been gated is shown by the red boxes in Figure 44. This was done in order to remove all insignificant outs of the signal. The output waveform shows the excitation pulse, analysis area and back wall echo (BWE). The analysis area is the critical area and contains information on the presence of any defects in the pipe. Each of the analysis area is gated out before further processing

since the excitation pulse and BWE are redundant. In other words, it contains no additional information useful for classification.

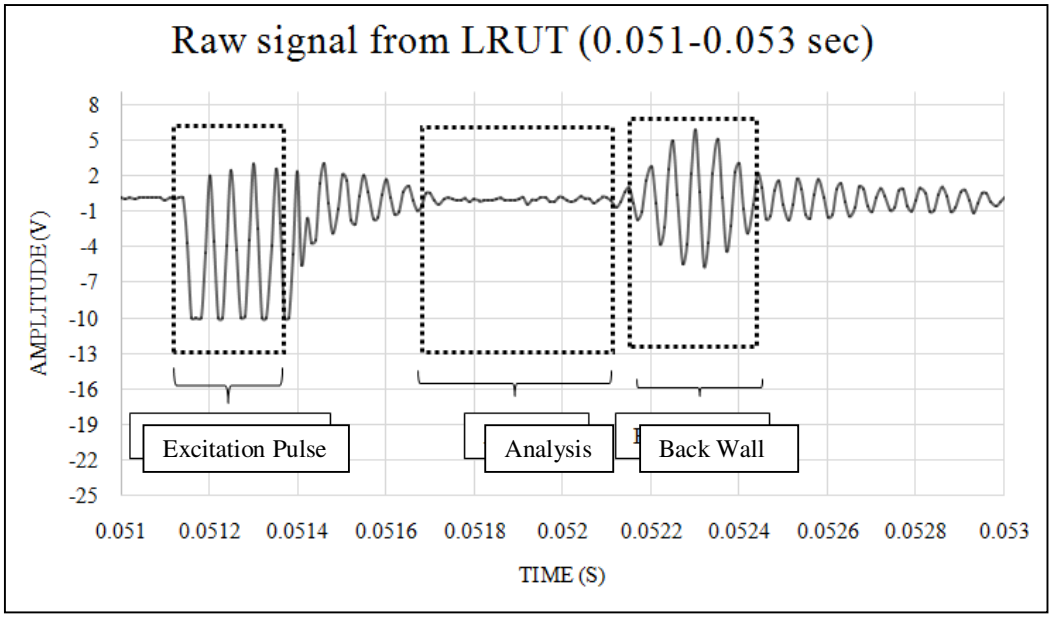


Figure 45: A segment of the raw signal which shows the excitation pulse, analysis area, and back wall echo.

This segment of the signal contains no evidence of guided waves or echoes since the pipe has no defects at this stage. No significant noise is present in the BWE signal.

As stated in Chapter 3 full circumferential (FC) defects are simulated on the test pipe. The FC defects are machined on the pipe using a lathe machine. Depth of 1 mm, 2 mm, 3 mm and 4 mm are labeled PIPE 0, PIPE 1, PIPE 2, PIPE 3 and PIPE 4 accordingly. Next, the signals from the LRUT are acquired for each defect depth.

Figure 46 shows the average signals of 2500 sampled analysis area for each of the defects.

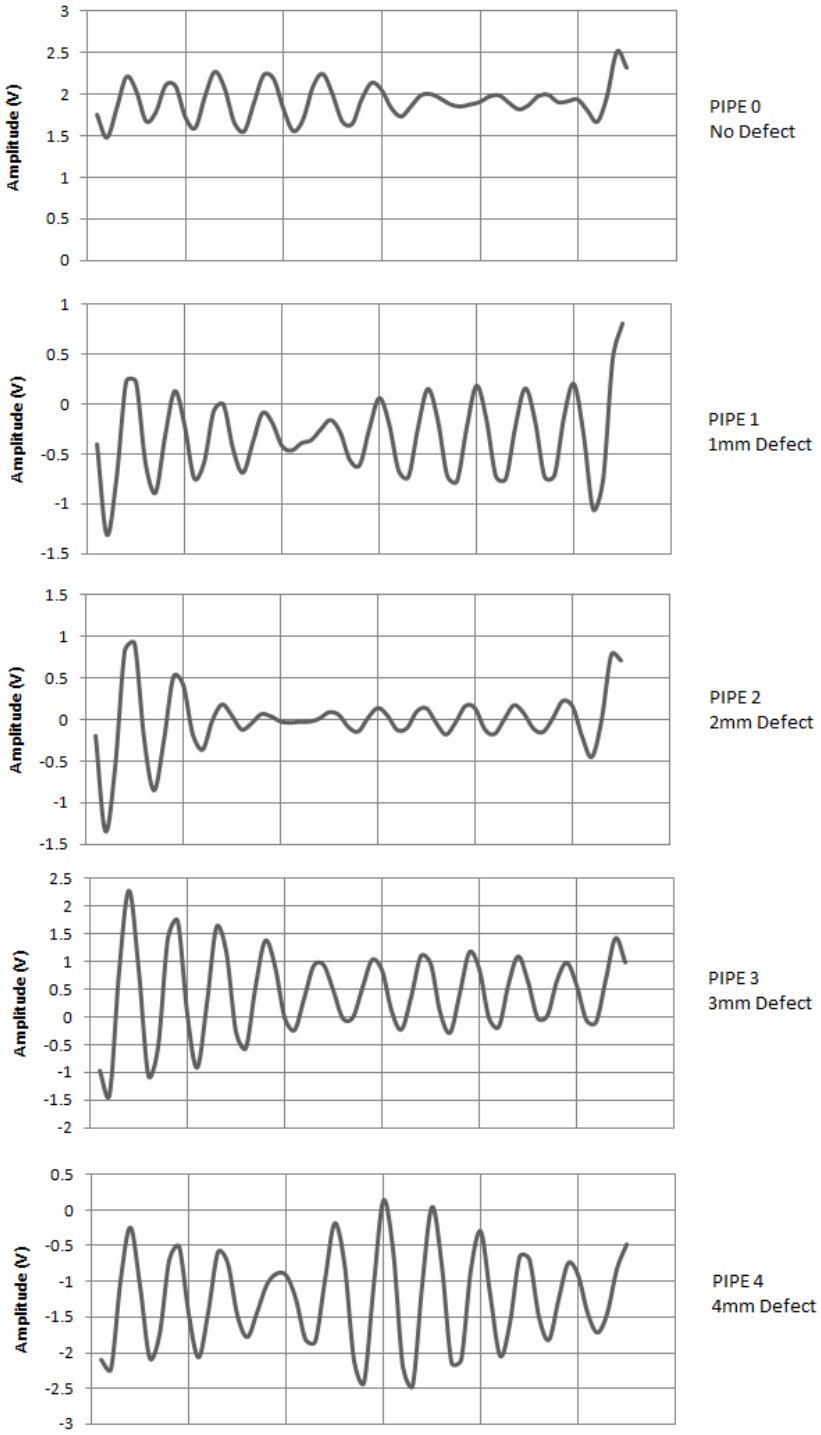


Figure 46: Sampled analysis area signals for each of the defects.

The signal from PIPE 0 shows minimal reflected echo compared to PIPE 1, PIPE 2, PIPE 3 and PIPE 4, hence the echo was due to the reflection of the FC defect. There are amplifications at the start and at the end of PIPE 1, PIPE 2, PIPE 3 and PIPE 4 signals. Reflected echo from PIPE 4 is significant, with a segment of wave similar to BWE wave.

In order to remove any signal fluctuations due to noise and unusual spikes, a rectangular bandpass filter is applied. Figure 47 filter shows the magnitude responses of the filter.

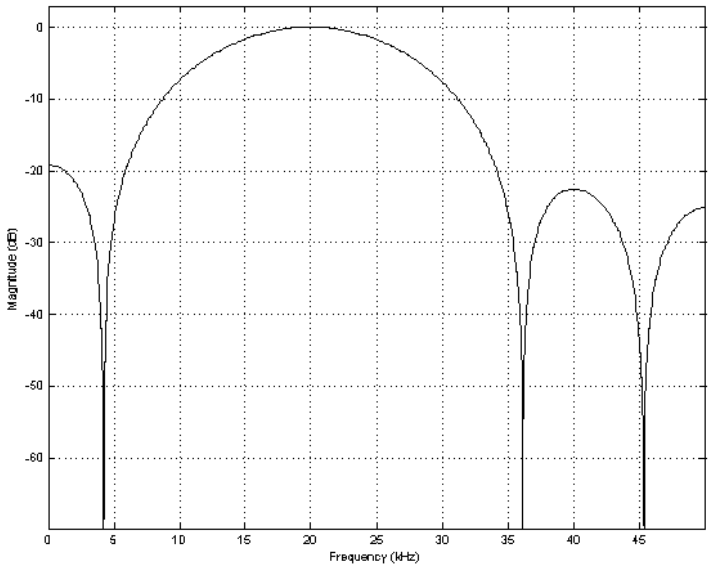


Figure 47: Magnitude response of rectangular bandpass filter.

The order for the rectangular filter is set to 10, first cutoff frequency (fc1) is set to 10000 Hz and second cutoff frequency (fc2) is set to 30000 Hz. Figure 48 shows the average signals of 2500 sampled analysis area for each of the defects.

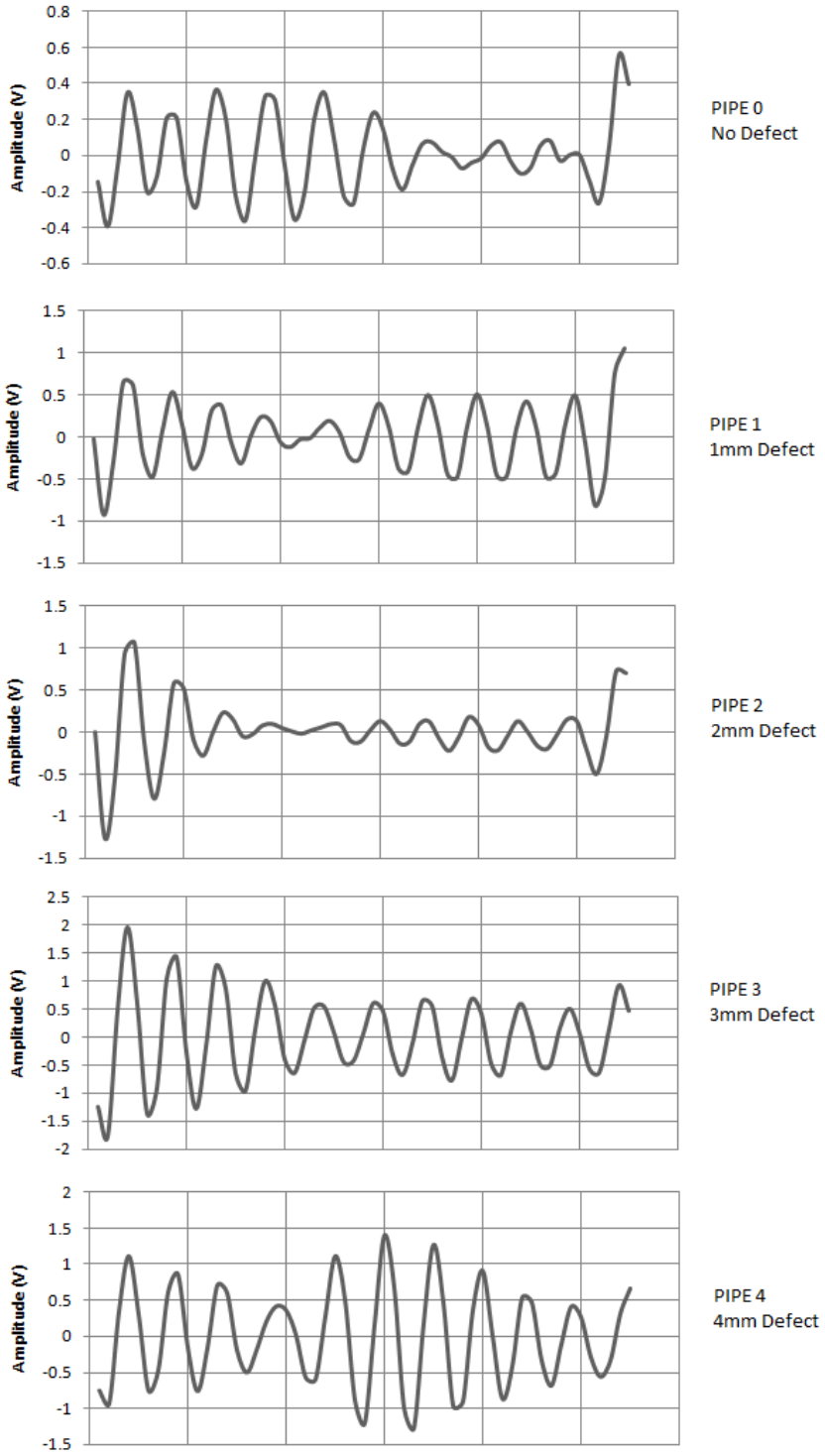


Figure 48: Sampled analysis area filtered signals for each of the defects.

Compared to raw signal, 20000 hz signal is significant in PIPE 0. The echoes in PIPE 1, PIPE 2, PIPE 3 and PIPE 4 is also significant. Less noise appear in PIPE 2 compared to raw signal accordingly. Furthermore, all signal has been normalized, as all signals average is at 0 V hence the filter removed the DC component from the signal.

4.2. Unsupervised Training

The previous section details the acquired signal from a visual point of view in order to inspect the reflected guided wave at different defect levels. This section presents the results of the analysis of the unsupervised training technique stated in Chapter 3. A detailed study is conducted to compare the performance between a commonly used unsupervised training method which is the self organizing map (SOM) and the proposed method. The dataset used is defined first so that the comparison of different techniques is standardized.

The raw samples are gated using a moving window, and then the entire analysis area is separated from other signal components; i.e. the excitation pulse and BWE. The areas of analysis are kept at constant length and contain 56 samples of data. As stated in the previous section, a bandpass filter is applied to the signal to remove

unwanted signals and DC components. The gated samples are labeled using class number from 0 to 4. Table 4 provides the details of the dataset.

Pipe	Defect Depths	Class	Number of training samples	Number of testing samples
PIPE 0	No Defect	0	500	2500
PIPE 1	1 mm	1	500	2500
PIPE 2	2 mm	2	500	2500
PIPE 3	3 mm	3	500	2500
PIPE 4	4 mm	4	500	2500

Table 4: Details of the dataset.

4.2.1. SOM Clustering

The samples produced listed in Table 4 will be used in this section. 100 samples from each class are used. The dataset is not labeled by a human expert but by the SOM will provide labels for every sample. The SOM is set use hex topology and Euclidean distance function. Next, the labels are compared with their respective class manually to produce a confusion map shown in Table 5.

		Manual Label				
		PIPE 0	*PIPE 1*	*PIPE 2*	*PIPE 3*	*PIPE 4*
Machine Label	*PIPE 0*	*99*	*0*	*94*	*0*	*0*
	PIPE 1	1	97	6	0	0
	PIPE 2	0	0	0	0	0
	PIPE 3	0	3	0	72	0
	PIPE 4	0	0	0	28	100

Table 5: Confusion map between SOM label and expert label.

The confusion map shows that the SOM accurately labeled PIPE 4 in one class. Only one sample from PIPE 0 is incorrectly labeled. However, 28 of PIPE 3 samples fell in PIPE 4 class which is incorrect. 94 of PIPE 2 samples are labeled as PIPE 1 and the remaining 6 samples in PIPE 1. In short, SOM is unable to label PIPE 2 correctly. This type of error is not tolerable in critical applications hence new approach is investigated and implemented.

4.2.2. All to all Clustering

The experiment is repeated using proposed unsupervised training technique presented in Chapter 3. Five distance formulas are used which are, Euclidean distance, hamming distance, city-block metric, and correlation distance. Table 6 shows the confusion map produced by proposed technique using Euclidean distance, Equation (22).

$$d_{st}^2 = (x_s - y_t)(x_s - y_t)' \qquad \text{- Equation (22)}$$

		Manual Label				
		PIPE 0	PIPE 1	PIPE 2	PIPE 3	PIPE 4
Machine Label	PIPE 0	98	0	0	0	0
	PIPE 1	1	100	0	0	0
	PIPE 2	0	0	100	0	0
	PIPE 3	1	0	0	99	0
	PIPE 4	0	0	0	1	100

Table 6: Confusion map between proposed technique (Euclidean) and real label.

The proposed technique is completely described in Chapter 3.

The proposed technique accurately clusters PIPE 1, PIPE 2 and PIPE 4 samples. Two samples from PIPE 0 are cluster to PIPE 1 class and PIPE 4 class which are incorrect. Only one sample from PIPE 3 is incorrectly clustered to PIPE 4 class. In conclusion, proposed method using Euclidean distance successfully clusters the dataset with minimal error.

Next, the proposed method is experimented with using the hamming distance. Table 7 shows the confusion map produced by the proposed technique using hamming distance, Equation (23).

$$d_{st} = (\#(x_{sj} \neq y_{tj})/n)$$

- Equation (23)

		Manual Label				
		PIPE 0	PIPE 1	PIPE 2	PIPE 3	PIPE 4
Machine Label	PIPE 0	100	100	100	100	100
	PIPE 1	0	0	0	0	0
	PIPE 2	0	0	0	0	0
	PIPE 3	0	0	0	0	0
	PIPE 4	0	0	0	0	0

Table 7: Confusion map between proposed technique (hamming) and real label.

Overall, the proposed method using hamming distance is unable to cluster the dataset. All samples fall into one cluster.

Then, the proposed method is experimented using cityblock distance measure. Table 8 shows the confusion map produced by proposed technique using cityblock distance measure, Equation (24).

$$d_{st} = \sum_{j=1}^{n} \left| x_{sj} - y_{tj} \right|$$

- Equation (24)

		Manual Label				
		PIPE 0	PIPE 1	PIPE 2	PIPE 3	PIPE 4
Machine Label	PIPE 0	99	0	0	0	0
	PIPE 1	0	100	2	0	0
	PIPE 2	0	0	98	0	0
	PIPE 3	1	0	0	99	0
	PIPE 4	0	0	0	1	100

Table 8: Confusion map between proposed technique (cityblock) and real label.

Next, the proposed method is experimented using correlation distance. Table 9 shows the confusion map produced by proposed technique using correlation distance, Equation (25).

$$d_{st} = 1 - \frac{\left(x_s - \bar{x}_s\right)\left(y_t - \bar{y}_t\right)'}{\sqrt{\left(x_s - \bar{x}_s\right)\left(x_s - \bar{x}_s\right)'}\sqrt{\left(y_t - \bar{y}_t\right)\left(y_t - \bar{y}_t\right)'}}$$

- Equation (25)

The proposed technique accurately clusters PIPE 1 and PIPE 4 samples. For PIPE 0, one sample is clustered to PIPE 3 class which is incorrect. Only two samples from PIPE 2 are clustered to PIPE 1 class which is also incorrect. Only one sample

from PIPE 3 is incorrectly clustered to PIPE 4 class. To sum up, proposed method using cityblock distance measure successfully clusters the dataset with minimal error.

		Manual Label				
		PIPE 0	*PIPE 1*	*PIPE 2*	*PIPE 3*	*PIPE 4*
Machine Label	*PIPE 0*	98	0	0	0	0
	PIPE 1	1	100	100	0	0
	PIPE 2	0	0	0	0	0
	PIPE 3	1	0	0	99	0
	PIPE 4	0	0	0	1	100

Table 9: Confusion map between proposed technique (correlation) and real label.

For this experiment, the proposed technique accurately clusters PIPE 4 into one class. Only one sample is incorrect for PIPE 3 and two data samples are incorrect for PIPE 0. However, the proposed technique cannot distinguish data sample between PIPE 1 and PIPE 2, thus it clusters them into same class. In short, the proposed method using correlation distance decreases the class number from 5 to 4 classes.

Table 10 shows the summary of the experiments

	SOM	**Euclidean**	**Hamming**	**Cityblock**	**Correlation**
PIPE 0	99%	98%	100%	99%	98%
PIPE 1	97%	100%	0%	100%	100%
PIPE 2	0%	100%	0%	98%	0
PIPE 3	72%	99%	0%	99%	99%
PIPE 4	100%	100%	0%	100%	100%
Average	73.6%	99.4%	20.0%	99.2%	79.4%

Table 10: Summary of results using SOM and proposed all-to-all clustering method.

Next, the label produced in previous experiment was used to train the SVM. The model produced by the SVM was then tested. As the LRUT system will be installed in remote areas where noise is anticipated, white Gaussian noises at different SNR levels were introduced to the dataset. The noise set from 0 to 50 dB in increment of one. First, a model was produced using SOM labeling. Figure 49 shows the accuracies at different levels of noises.

In normal consequent, it is not feasible to compare between human and machine label as the human label is agreed to be 100% accurate. However, the intention of this experiment is to show that the proposed system has comparable accuracy with the human expert.

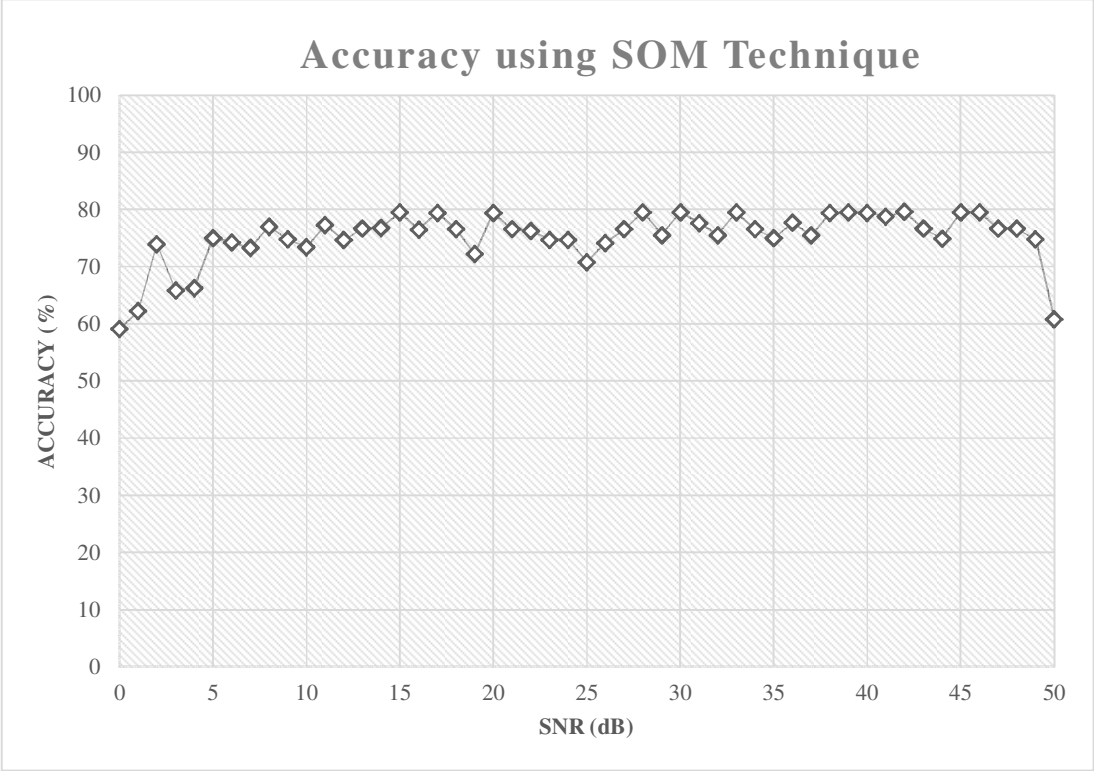


Figure 49: Accuracy at different level of noise for model with SOM label.

The accuracies were consistent between 60% and 80% as shown in Figure 49. Next, the SVM model was produced by using labels from the proposed method utilizing Euclidean distance. Figure 50 shows the accuracy at different levels of noise for the proposed method using Euclidean distance

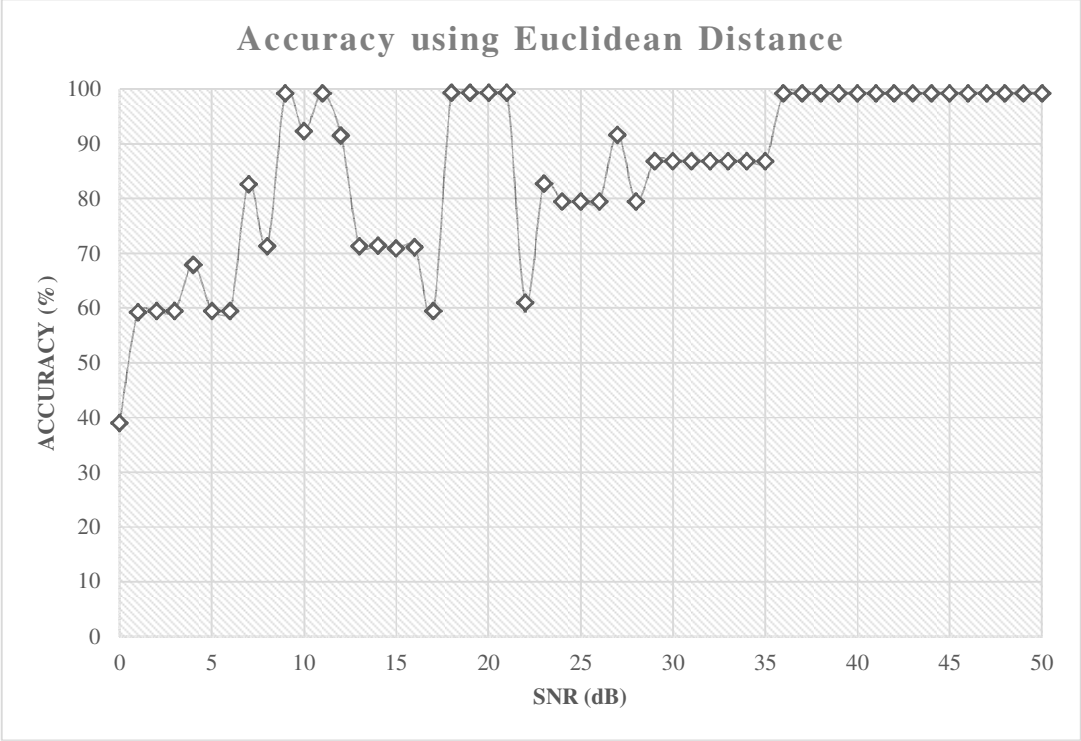


Figure 50: Accuracy at different level of noise for model labeled with the proposed method label. (Euclidean Distance))

The results show high accuracies at high SNR, 35 dB and above. However, the accuracies were not consistent at low SNR because at these SNR levels the noisy signals may have altered the original signal extensively. Next, a SVM model is produced by using labels from the proposed method utilizing hamming distance. Figure 51 shows the accuracy at different levels of noise for the proposed method using hamming distance.

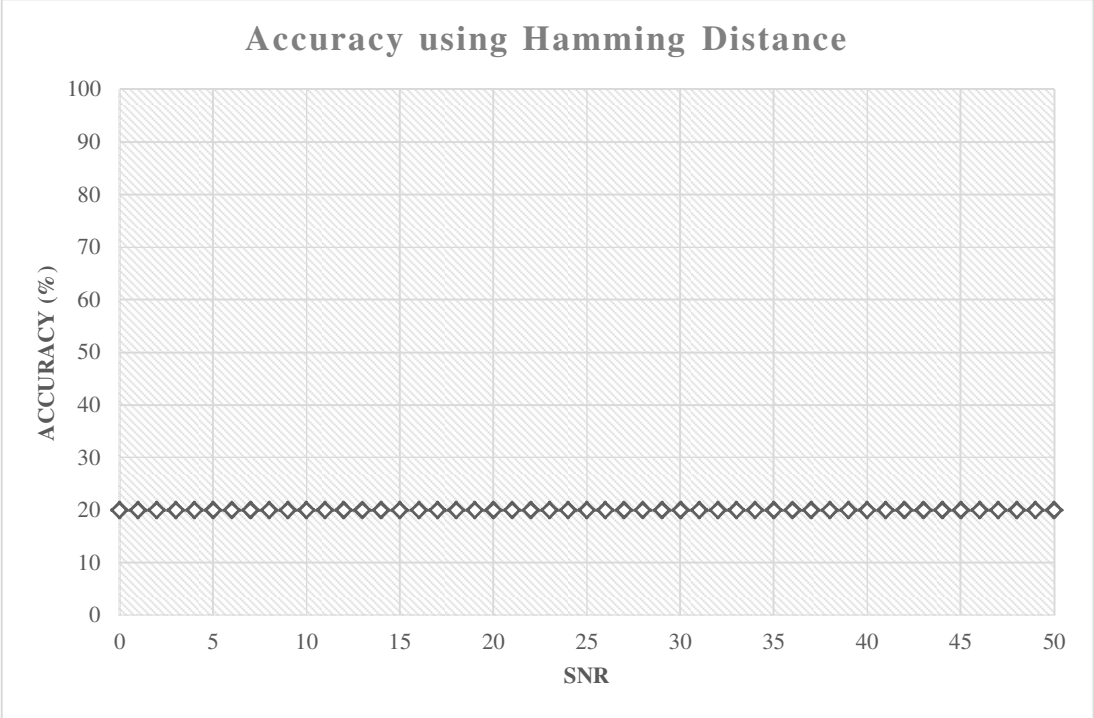


Figure 51: Accuracy at different level of noise for model with proposed method label. (hamming).

Overall, the proposed method using hamming distance has low accuracies for all SNR levels. This is due to the fact that the hamming distance has labeled the signals into one cluster, PIPE 0, refer Table 7. Next, an SVM model is produced by using the labels from the proposed method utilizing cityblock distance measure. Figure 52: Accuracy at different level of noise for model labeled with proposed method label. (Cityblock distance measure). Figure 52 shows the accuracies at different level of noise for the proposed method using cityblock distance measure.

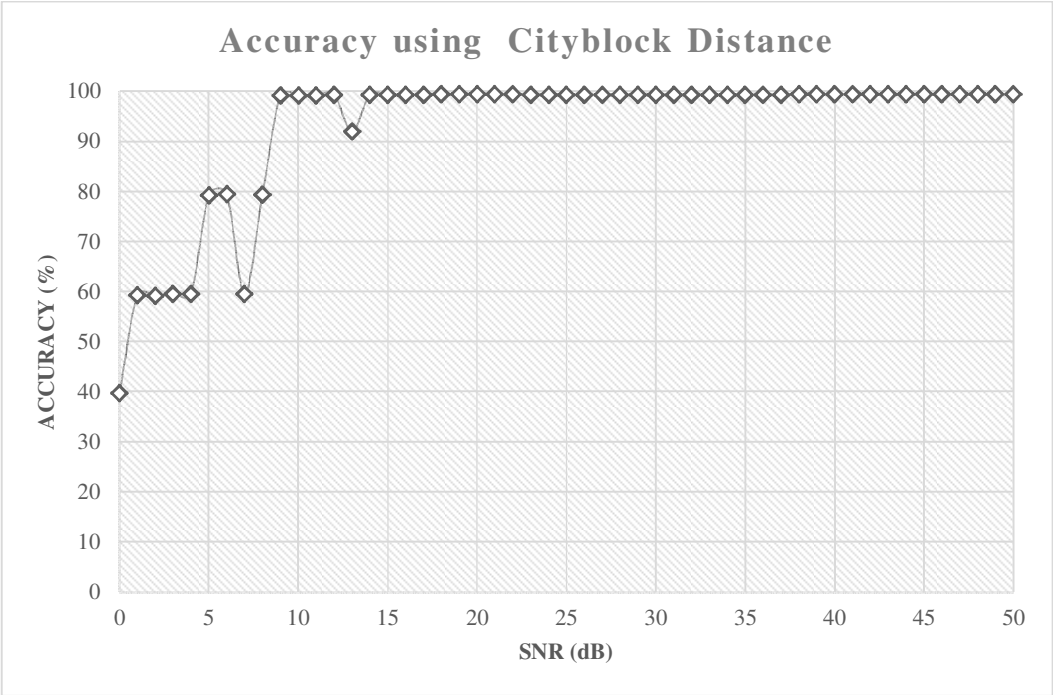


Figure 52: Accuracy at different level of noise for model labeled with proposed method label. (Cityblock distance measure).

The results show low accuracies at low SNR, 8 dB and below. The accuracies were consistently high at 10dB and above. Finally, a SVM model is produced by using labels from the proposed method utilizing correlation distance. Figure 53 shows the accuracy at different level of noise for proposed method using correlation distance.

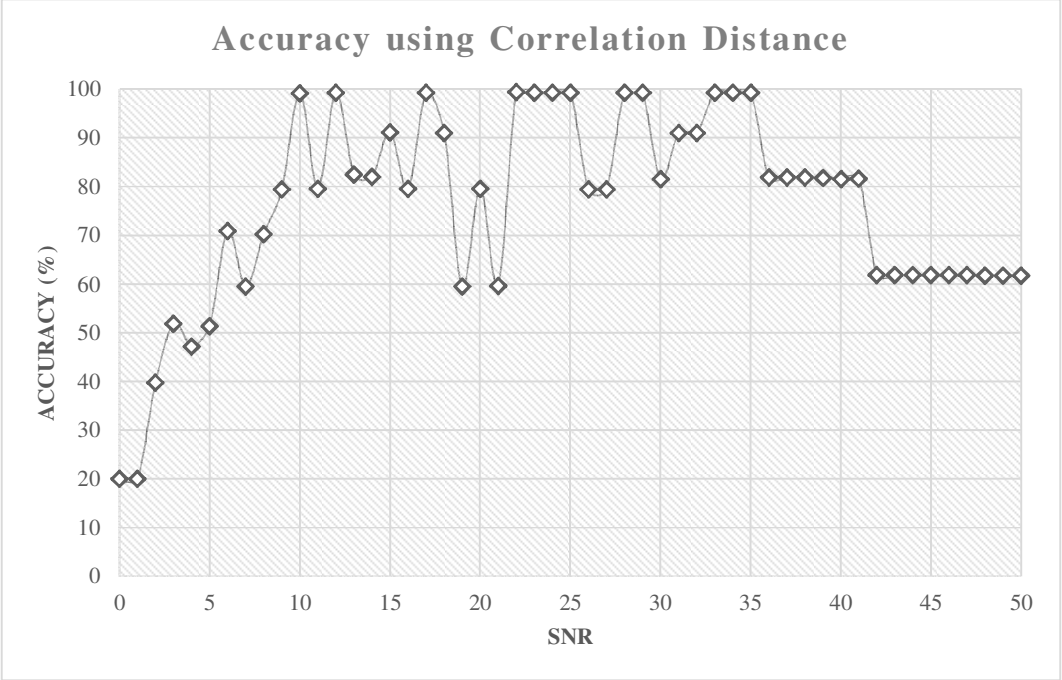


Figure 53: Accuracy at different level of noise for model with proposed method label. (Correlation distance).

The results show accuracy less than 60 % at low SNR, below 5 dB. The accuracies are inconsistent above 10 dB to 50 dB. In addition even at high SNR, above 35, the accuracies drop to 60 %. These may be due to the fact that the number of cluster produced by the proposed system is not consistent with human expert, refer Table 9.

The result of the experiment demonstrates that the performance of the proposed method is promising using cityblock distance kernel. Compared to the SOM, the proposed method gives a higher accuracy consistently. The proposed method using cityblock distance also shows better performance when noise is introduced to the dataset. Generated labels were used in all further experiments.

4.3. Online Support Vector Machine

In this section, a detailed study is conducted to compare the performance between the batch trained SVM and the incremental trained SVM. The dataset introduced in the previous section is used. Each data is paired with plus 1mm data: 0mm with 1mm, 1mm with 2mm, 2mm with 3mm and 3mm with 4mm. All the labels are produced using proposed all-to-all clustering technique in previous section. The reason for this pairing is to simulate the steps of the pipe corrosion. SVM-C parameter is set to 1. The gamma parameter for Radial Basis Function (RBF) kernel is set to 0.01786 which is equivalent to 1/56. The degree for the Polynomial is set to 2. As for the classification accuracy comparison between SVM models, the value of parameter *R* for incremental training is varied from 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 to 1.0 sequentially. For execution time evaluation, the number of training data is varied ranging from 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 120, 140, 160, 180 to 200. Then the algorithm will carry out the classification on a set of 400 testing data based on the hyperplane which is constructed at the training phase. The simulation of the SVM models is carried out in MATLAB and the 'tic-toc' function is used to measure the time.

4.3.1. Results

Parameter *R* is used to decide whether the data point is significant to construct the hyperplane. In Equation (23) the larger the value of checkpoint, the more data points are passed to the Active Set algorithm to redefine the hyperplane, refer Figure 35: Flowchart of the Proposed Incremental SVM in Chapter 3.

	Incremental				
	R = 0.1	*R = 0.2*	*R = 0.3*	*R = 0.4*	*R = 0.5*
0 mm vs. 1 mm	50	50	50	50	50
1 mm vs. 2 mm	50	50	50	50	50
2 mm vs. 3 mm	50	50	50	50	50
3 mm vs. 4 mm	50	50	50	50	50

	Incremental					Batch
	R = 0.6	*R = 0.7*	*R = 0.8*	*R = 0.9*	*R = 1.0*	
0 mm vs. 1 mm	50	50	50	50	50	99.75
1 mm vs. 2 mm	50	50	50	50	50	99.5
2 mm vs. 3 mm	50	50	50	50	50	99.25
3 mm vs. 4 mm	50	50	50	50	50	99.37

Table 11: Accuracy of SVM using Linear kernel across the testing range of parameter R.

	Incremental				
	R = 0.1	*R = 0.2*	*R = 0.3*	*R = 0.4*	*R = 0.5*
0 mm vs. 1 mm	50	50	50	50	50
1 mm vs. 2 mm	50	50	50	50	50
2 mm vs. 3 mm	50	50	50	50	50
3 mm vs. 4 mm	50	50	50	50	50

	Incremental					Batch
	R = 0.6	*R = 0.7*	*R = 0.8*	*R = 0.9*	*R = 1.0*	
0 mm vs. 1 mm	50	50	50	50	50	99.75
1 mm vs. 2 mm	50	50	50	50	50	99.5
2 mm vs. 3 mm	50	50	50	50	50	99.375
3 mm vs. 4 mm	50	50	50	50	50	99.5

Table 12: Accuracy of SVM using Radial Basis kernel (RBF) across the testing range of parameter R.

	Incremental				
	R = 0.1	*R = 0.2*	*R = 0.3*	*R = 0.4*	*R = 0.5*
0 mm vs. 1 mm	89.875	91.125	93.875	93.875	93.875
1 mm vs. 2 mm	99.75	99.75	99.625	99.75	99.75
2 mm vs. 3 mm	99	99.125	99.125	99.125	92.625
3 mm vs. 4 mm	99	99	99	99	99

	Incremental					Batch
	R = 0.6	*R = 0.7*	*R = 0.8*	*R = 0.9*	*R = 1.0*	
0 mm vs. 1 mm	93.875	93.875	99.875	99.875	99.875	100
1 mm vs. 2 mm	99.75	99.75	99.75	99.75	99.75	99.75
2 mm vs. 3 mm	92.625	92.625	92.625	92.625	92.625	99.5
3 mm vs. 4 mm	99	99	99	99	99	99

Table 13: Accuracy of SVM using Polymonial kernel across the range of parameter R.

Table 11, Table 12 and Table 13 show the classification accuracy of the SVM for each kernel. SVM with batch training shows high accuracy with all 3 kernels. On the other hand, the SVM with incremental training using linear and radial basis function kernel shows a poor result with 50% accuracy. This indicates that the SVM has classified all datapoints into one class, as the samples sizes for each class are equal. However, SVM with Polynomial kernel shows high accuracy for all datasets.

The execution time for 4 different solver namely, Matlab built in QP solver, Quadprog (Batch), Active Set (Batch), and Sequence Active Set (Incremental) is measured. Only the Polynomial kernel is used in this experiment as it shows

comparable result in the previous experiment. Dataset '0mm vs 1mm' is used in all experiments as it has the lowest average accuracy found in previous experiments. All tests are conducted in Matlab 2011a Window 7 running on Pentium D 3.40 Ghz with 2 GB RAM.

Figure 54 depicts the execution time for solvers using the Polynomial kernel against increasing number of training data. The execution time of Incremental (sequence) Active Set is lower than 'QuadProg' by approximately a factor of 4.

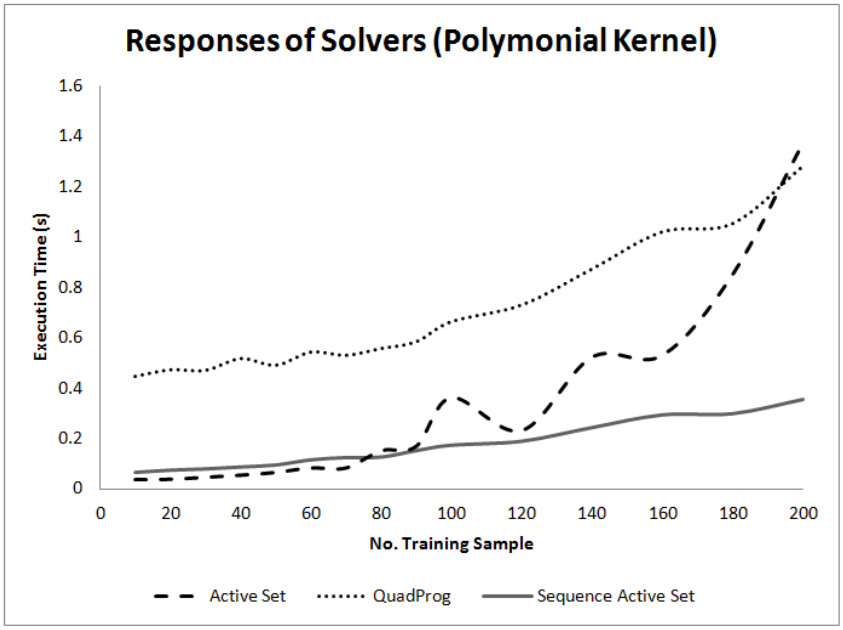


Figure 54: Execution Time Taken by Solvers in Solving the Problem Corresponds to Number of Training Data (Polynomial Kernel).

The experiments are repeated using the proposed and other incremental SVM methods. Method A is proposed by Gert Cauwenberghs [101], which uses Adiabatic Incremental approach. This method offers decremental function where it is able to

"unlearn" some particular training samples. However, the proposed NDT application does not need the function. Method B applies full-retrain with Simple SVM approach and was proposed by S.V.N. Vishwanathan [102]. Simple SVM is a suitable method to apply full-retrain approach as it works by adding a point at a time to the current dataset. Figure 55 demonstrates the response of solvers using three different incremental SVM methods

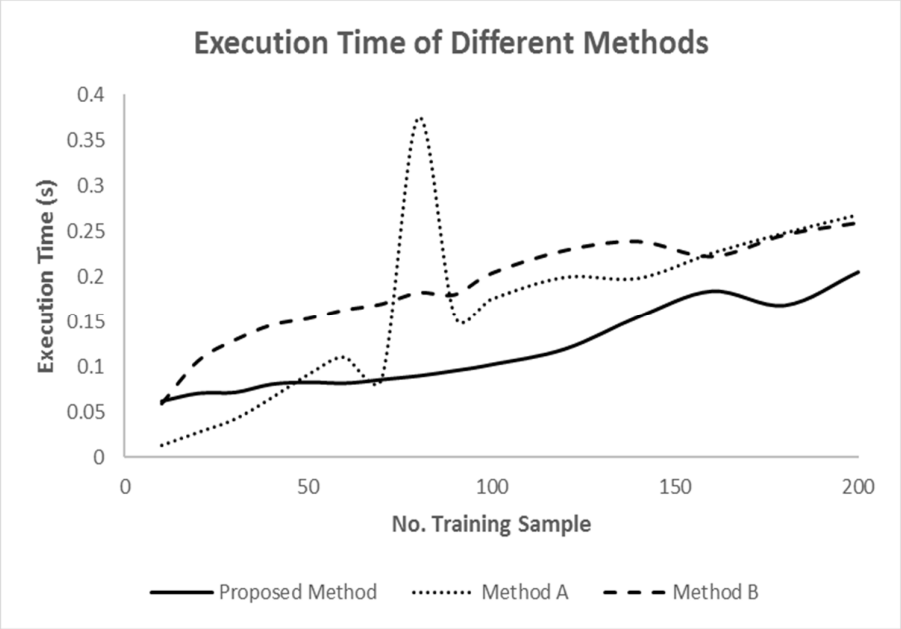


Figure 55: Execution Time Taken by three different incremental SVM methods.

The results show that the proposed technique has lower execution time in average as compared to the other methods. Method A shows uneven values through the process, methods with inconsistent performance are not practical for a critical application like the proposed system. Method B constantly has higher execution time as compared to the proposed method.

Figure 56 shows the accuracies for all three comparing methods. All methods are able to maintain over 95% accuracy after 20 samples

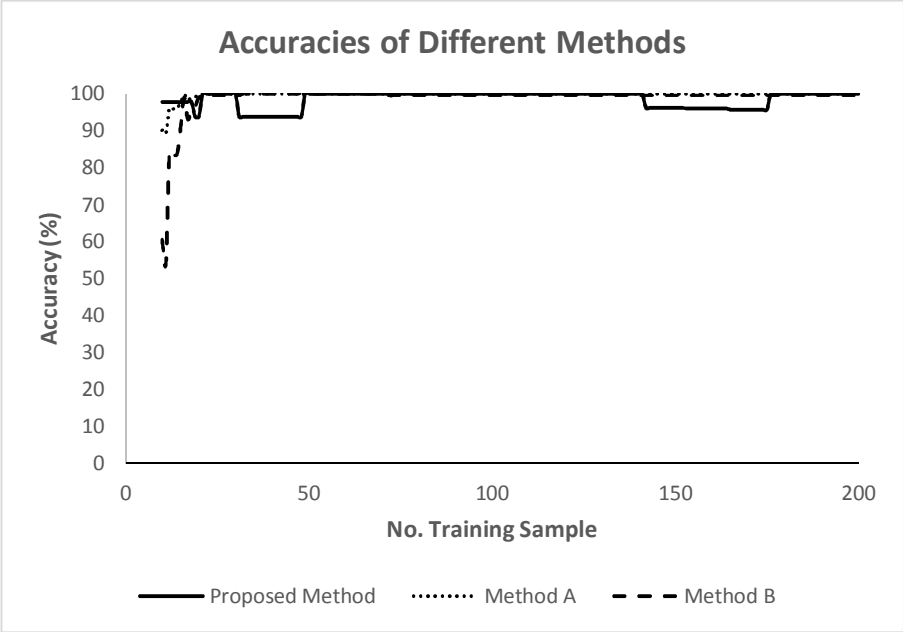


Figure 56: Accuracy of the Proposed Method, Method A and Method B.

An incremental learning SVM based on the active set method is proposed. It is able to handle sequence data collected from the LRUT pipeline defect detection system. The performance of the algorithm is evaluated from two aspects, classification accuracy and execution time. The results of the simulation demonstrate that the performance of the proposed algorithm is promising using a Polynomial kernel in dealing with sequential data. Compared to batch learning which uses the active set method and Matlab Quadprog, the proposed incremental SVM works faster

without compromising accuracy. The proposed method also shows better performance in term of execution time as compared to other incremental SVM methods, while the accuracy is comparable.

4.4. Genetic Algorithm

This section details the results of the proposed approach to solve the Quadratic Programming (QP) problem in SVM using Genetic Algorithms. As discussed in Chapter 3 a new strategy is introduced to reduce the number of generations while maintaining the proposed incremental SVM accuracy. The proposed approach is tested on the dataset acquired from the experimental rig.

The first set of results is obtained by implementing the conventional GA as QP solver in the proposed incremental SVM. Cross over ratio is set to 0.8 which is high because the fitness function is known to have global minimum and population size is set to 30 which is low to reduce computational cost as the fitness function is complex to execute [103][104]. In order to maintain the proposed incremental SVM accuracy the final fitness value must match the value when using Active Set QP Solver. Figure 57 shows the best fitness versus number of generations.

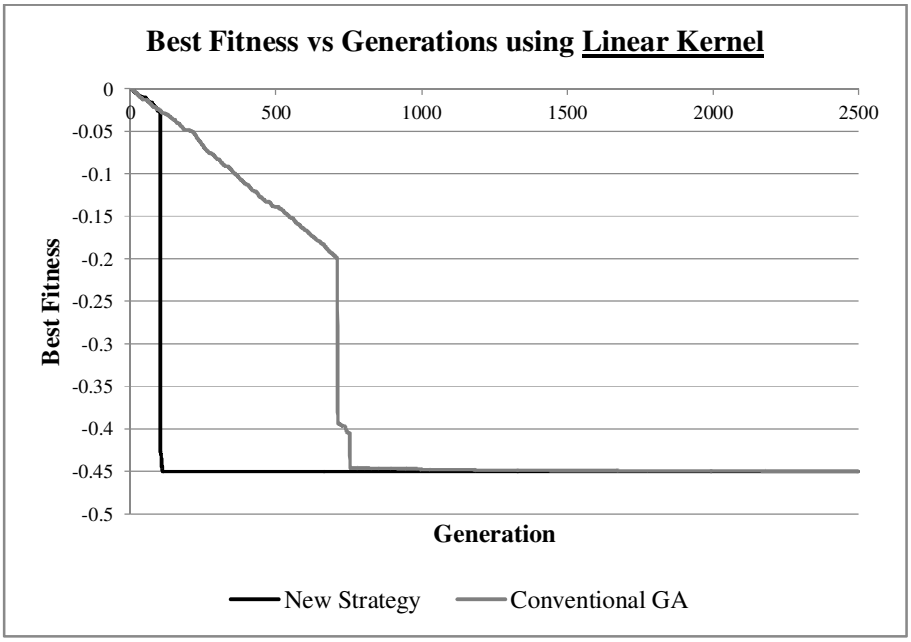


Figure 57: Best fitness versus number of generations (Linear Kernel).

Conventional GA took 1675 generations to find solve the QP while the new strategy only took 111 generations. The final value for the fitness function using linear kernel is -0.45.

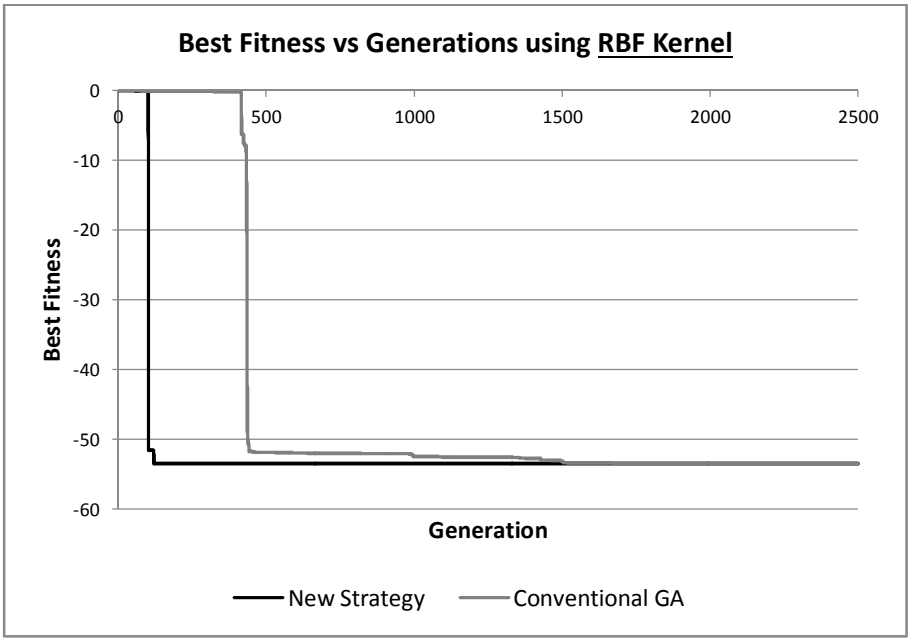


Figure 58: Best fitness versus number of generations (RBF Kernel).

Conventional GA took 1764 generations to find solve the QP while the new strategy only took 121 generations as shown in Figure 58. The final value for the fitness function using RBF kernel is -53.4.

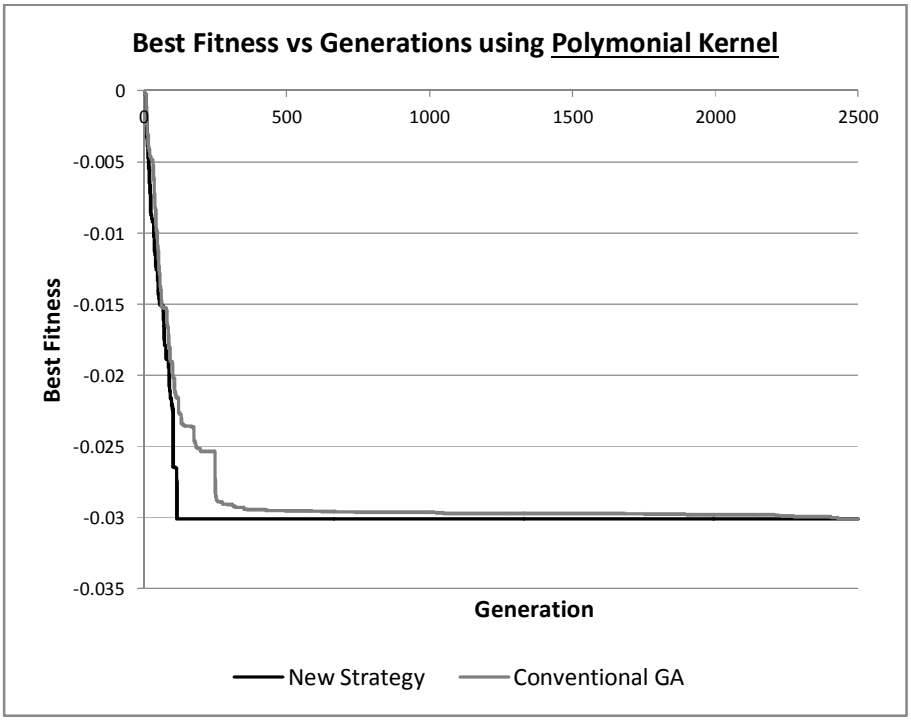


Figure 59: Best fitness versus number of generations (Polymonial Kernel).

Conventional GA took 2429 generations to find solve the QP while the new strategy only took 117 generations as shown in Figure 59. The final value for the fitness function using RBF kernel is -0.03.

Table 14 shows the results summary for the experiments.

Kernel	Number of Generations using conventional GA (A)	Number of Generations using new strategy (B)	Reduction Percentage (A-B)/A × 100	Fitness Value
Linear	1675	111	93.4%	-0.45
RBF	1764	121	93.1%	-53.4
Polymonial	2429	117	95.1%	-0.03

Table 14: Summary of the proposed GA strategy results.

4.5. Proposed Strategy vs Active Set QP Solver

The experiment was repeated using Active Set QP solver to measure and compare the execution time for both solvers. Both solvers were set to use polymonial kernel. Number of samples are varied at 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800 and 2000 samples.

No. Training Data	**Execution Time (s)**	
	Proposed Strategy	**Active Set QP Solver**
200	0.3547	0.0352
400	1.2419	0.8416
600	7.4823	6.5723
800	15.0823	16.5178
1000	27.5343	31.1133
1200	46.9811	57.9225
1400	76.1928	102.6956
1600	127.9230	177.3930
1800	215.2169	291.6643
2000	364.7613	474.5246

Table 15: Execution times for proposed strategy and Active Set QP solver.

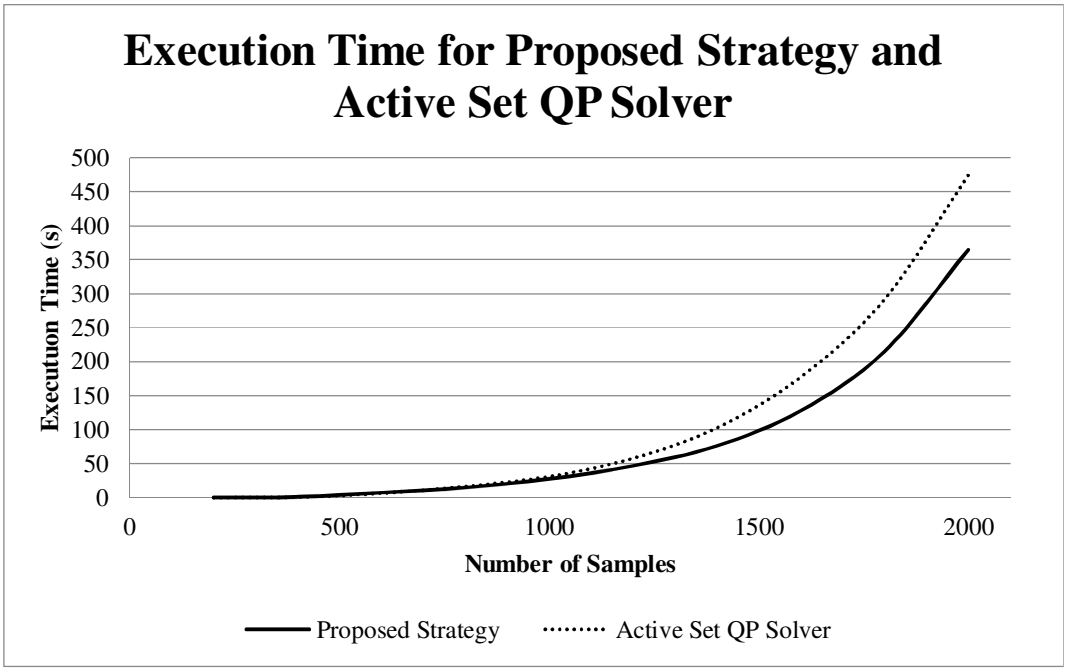


Figure 60: Execution Time for Proposed Strategy and Active Set QP Solver.

The results show that the proposed strategy has lower execution time in average as compared to the Active Set QP Solver. The differences of the execution times are higher as number of samples increased.

In conclusion, the results demonstrate the capability of the proposed method to reduce the execution time.

4.6. Multi-class classification

For evaluating the performance for multiclass classification using one-against-one approach, a set of experiments are carried out using the dataset described in the previous section. one-against-one dataset is constructed which are 0mm vs 1mm, 0mm vs 2mm, 0mm vs 3mm, 0mm vs 4mm, 1mm vs 2mm, 1mm vs 3mm, 1mm vs 4mm, 2mm vs 3mm, 2mm vs 4mm and 3mm vs 4mm. Total number of dataset is 10, this verified the formula for number of binary classifier described in Chapter 3:

$$Number\ of\ binary\ classifer = \frac{M(M-1)}{2},$$

$$where\ M\ is\ number\ of\ classes$$

The gamma parameter for Radial Basis Function (RBF) kernel is set to 0.01786 which is equivalent to 1/(number of features) (1/56) [105][106]. The degree for Polynomial is set to 2. The number and percentage of Support Vectors (SVs) are evaluated.

	Number of Support Vectors (SVs)	**SVs Percentage**
0mm vs 1mm	16	8.00%
0mm vs 2mm	22	11.00%
0mm vs 3mm	11	5.50%
0mm vs 4mm	10	5.00%
1mm vs 2mm	17	8.50%
1mm vs 3mm	15	7.50%
1mm vs 4mm	11	5.50%
2mm vs 3mm	11	5.50%
2mm vs 4mm	12	6.00%
3mm vs 4mm	12	6.00%
Total number of unique SV	82	

Table 16: Number and percentage of Support Vector for each classifier using linear kernel.

	Number of Support Vectors (SVs)	**SVs Percentage**
0mm vs 1mm	45	22.50%
0mm vs 2mm	47	23.50%
0mm vs 3mm	17	8.50%
0mm vs 4mm	18	9.00%
1mm vs 2mm	56	28.00%
1mm vs 3mm	26	13.00%
1mm vs 4mm	25	12.50%
2mm vs 3mm	24	12.00%
2mm vs 4mm	20	10.00%
3mm vs 4mm	24	12.00%
Total number of unique SV	156	

Table 17: Number and percentage of Support Vector for each classifier using RBF kernel.

	Number of Support Vectors (SVs)	**SVs Percentage**
0mm vs 1mm	39	19.50%
0mm vs 2mm	47	23.50%
0mm vs 3mm	11	5.50%
0mm vs 4mm	11	5.50%
1mm vs 2mm	45	22.50%
1mm vs 3mm	15	7.50%
1mm vs 4mm	18	9.00%
2mm vs 3mm	12	6.00%
2mm vs 4mm	14	7.00%
3mm vs 4mm	14	7.00%
Total number of unique SV	137	

Table 18: Number and percentage of Support Vector for each classifier using Polymonial kernel.

Table 16, Table 17 and Table 18 show the number and percentage of support vector for each binary classifier. The total number of unique support vectors reported in the tables represent the number of training data which corresponds to at least one support vector of a binary problem. For one-against-one approach, one training data may be a support vector of different binary classifiers. Note that all of binary classifiers have less than 50% of support vectors which indicate the classifiers have good generalization performance [107][108].

As the proposed system will implement incremental learning SVM and the results from previous section show that the polymonial kernel offer satisfactory performance, only 4 classifiers are found to be relevant which are 0mm vs 1mm, 1mm vs 2mm, 2mm vs 3mm and 3mm vs 4mm using the polymonial kernel. Table 19 shows the numbers and percentages of support vectors for relevant classifiers.

	Number of Support Vectors (SVs)	**SVs Percentage**
0mm vs 1mm	39	19.50%
1mm vs 2mm	45	22.50%
2mm vs 3mm	12	6.00%
3mm vs 4mm	14	7.00%

Table 19: Number and percentage of Support Vector for each classifier using Polymonial kernel (Incremental Order).

A set of experiments implementing multiclass classification using one-against-one approach on the dataset was presented. The results of the simulations demonstrate that the performance of the proposed approach is promising. In addition, the proposed approach also shows good generalization performance.

4.7. Graphical User Interface (GUI)

The proposed system was incorporated in a GUI. The GUI was designed to have minimal user intervention.

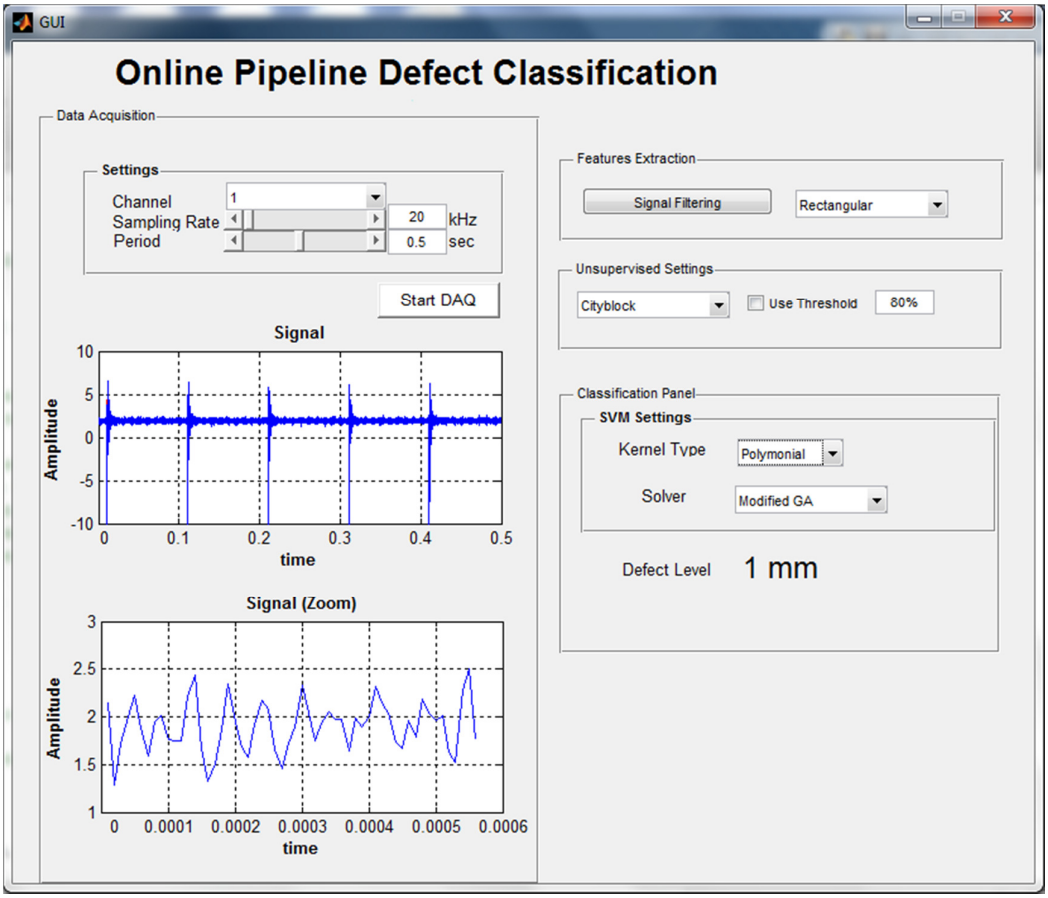


Figure 61: Graphical User Interface (GUI) for the proposed system

Figure 61 shows the GUI for the proposed system. User need to configure the DAQ setting according to the DAQ specification and then click the "Start DAQ" button. The signal acquired will be showed in real time. The GUI will alert the user if it founds the signal is in new defect class.

4.8. Performance against Benchmark Dataset

The framework of the system was tested using a benchmark dataset. “Activity Recognition from Single Chest-Mounted Accelerometer Data Set” dataset was used as the benchmark dataset as it is times-series similar to the pipeline dataset.[109] The dataset consist of data from an accelerometer wear by 15 samples performing 7 activities. The activities are:

1. Working at Computer.
2. Standing Up, Walking and Going Up/Down stairs.
3. Standing.
4. Walking.
5. Going Up/Down Stairs.
6. Walking and Talking with Someone.
7. Talking while Standing.

Similar activities are merged together; activity 2 with activity 5 and activity 4 with activity 7. The dataset was split into 80% training dataset and 20% testing dataset. The dataset was tested using Random Forest Algorithm. Random Forest is an ensemble classifier where it forms many classification trees, then each tree votes for a class and finally it will choose the classification with the highest vote. Next the dataset was tested using conventional SVM with human label. Finally the dataset was tested using proposed framework where labels are generated using all-to-all clustering technique and trained using proposed incremental SVM. Table 20 shows the performance in term of accuracy for the classifiers.

	Random Forest	SVM	Proposed Framework
Stairs	89.80%	90.10%	85.00%
Walking	95.90%	96.00%	90.50%
Talking	92.90%	92.90%	80.30%
Standing	88.80%	85.80%	75.30%
Working	97.70%	95.20%	95.70%
AVERAGE	93.02%	92.00%	85.36%

Table 20: Summary of accuracies using different classifiers.

The result shows that all classifiers have high average accuracies which are above 80%. Random Forest has highest average accuracy, 93.02%. SVM has highest accuracies on “Stairs”, “Walking” and “Talking” classes but not at “Standing” and “Working” classes. Proposed Incremental SVM has lower accuracy in most of the classes except “Working” class which is higher than SVM. The result also shows the proposed framework using Incremental SVM has below 80% accuracy in “Working” class; however it is coherent with other classifier where all of them have lower accuracy for the class.

In conclusion the proposed framework has a comparable result in term of accuracy on the benchmark dataset.

CHAPTER 5: CONCLUSION

5.1. Summary and Conclusion

In this thesis, a method which incorporates machine learning techniques into an oil and gas pipeline defect classification system is presented. Traditionally, pipeline inspection is performed at predetermined times using techniques such as pigging which requires a human expert to perform measurements and interpret results. Thus, the condition of the pipe between these predetermined testing periods is unknown. The proposed system is able to work continuously in real-time with minimal expert supervision.

An experimental rig implementing LRUT that is capable of detecting and classifying pipe defects is designed and constructed. LRUT is used mainly due to the ability to detect corrosion under insulation and able to inspect long lengths of pipe from a single location. The LRUT system is designed and constructed for 5.5 inch pipe diameters. The test system used 16 piezoelectric transducers, arranged axially in a ring and mounted on the pipe using a robust collar assembly. Experiments are conducted on test pipes with full circumferential defects. The defect depths for each pipe are varied at 1mm, 2mm, 3mm and 4mm. A test pipe with no defects is also used as the control sample. Visual inspection of the raw signal acquired from the LRUT showed good correlation between the signal frequency, amplitude and different defect depths. Thus, the experimental rig is able to detect full circumferential defects.

In order to achieve the objectives of the project, three new techniques were proposed. These techniques were developed specifically for the project. First is an unsupervised algorithm for assigning labels to training data with minimal human intervention was developed. The term unsupervised here means that the technique requires minimal human intervention. As a result, a technique using all-to-all distance estimation is proposed (page 49). The proposed method is able to label data automatically which will be used to train the SVM. Five selected distance formulations were investigated; Euclidean distance, hamming distance, city-block metric and correlation distance were investigated to get the best performance. The performance of the technique using city-block metric on data from the experimental rig shows good result. In addition, the technique is compared with state of the art unsupervised learning algorithm which is the SOM and results show that the proposed technique surpasses SOM accuracy (page 75).

The second new technique proposed and developed is an incremental SVM classifier (page 56). Since the proposed system works continuously and produces a tremendous amount of data, a technique is developed to take the data in one at a time without the need to re-train previous data. Conventional SVM works in batch mode and requires all previous historical data to be available for creating a model. On the contrary, incremental SVM only maintains relevant data, thus the memory and computation requirements are at a minimal level. The performance of the algorithm on data acquired from the experimental rig show good results as the proposed technique is able to train the data incrementally. The proposed technique is also compared with incremental SVM proposed by other research groups, the performance

was found to be better in term of execution time while the accuracy is comparable (page 87).

The third new technique proposed is an approach to solve the Quadratic Programming (QP) problem in SVM using a Genetic Algorithm (GA). The traditional approach requires the GA to generate a large number of generations in order to solve the QP problem, thus it is not feasible for the proposed system. A new approach is proposed to reduce the number of generations while maintaining the SVM accuracy (page 60). This proposed approach manipulates the SVM conditions by introducing an additional stopping criteria where most of the α that lean towards 0 are removed. This makes the GA objective function less complex to solve. Experiment using data acquire from the test rig confirms that the proposed method successfully solve the QP problem. In addition, the proposed method show significant reduction in term of number of generations compared to conventional approach (page 93).

The SVM is a binary classifier which classifies data into two classes. However, the project requires the proposed system to classify the pipeline defects into more than two classes. Therefore multiclass classification using one-against-one approach is investigated. The results using data acquired from the test rig show that the performance of the proposed approach is promising. In addition, the one-against-one approach shows good generalization performance.

5.2. Future Work

Although the results from the experiments have demonstrated the effectiveness of the proposed system, it can be further developed in a number of ways. The potential future efforts can be viewed from the angle of improvements to the proposed system.

5.2.1. Parallel processing

In terms of speed for the SVM training phase, running a single processor at a time might prove to be expensive in computational cost. For example, in the training phase using the GA discussed in Chapter 3, each "individual" is computed one at a time. This can be very costly for the SVM because of the complexity of the corresponding fitness functions. As such, parallelization can be employed, assigning each of the "individual" to separate processors. The aim would be to improve the speed of the SVM training phase for identical datasets which should produced identical model.

5.2.2. Feature Extraction

The proposed system utilized raw signal data that may lead to high dimensional dataset. For example, the proposed system used 56-dimension data. This may lead to high memory requirement. However this can be improved by making the system analyzing extracted data features only. For example, the statistical characteristics of the signal such as standard deviation, kurtosis and skewness may be

used as the input for the classifier. Implementing signal processing technique such as Principal

Component Analysis (PCA) and Fast Fourier Transform (FFT) analysis may also reduce the data dimension. The aim of this potential improvement is to reduce the memory requirement for the proposed system; however accuracy would have to be maintained.

REFERENCES

[1] S. Thomas and R. a. Dawe, “Review of ways to transport natural gas energy from countries which do not need the gas for domestic use,” *Energy*, vol. 28, pp. 1461–1477, 2003.

[2] M. Edgar, M. Udaeta, J. Luiz, D. O. Bernal, L. Claudio, R. Galvão, J. Aquiles, and B. Grimoni, “Natural Gas Virtual-Pipeline for Alternative Energy Distribution,” 2012.

[3] Association of Oil Pipelines (AOPL), “Report on Shifts in Petroleum Transportation,” 2011.

[4] D. Furchtgott-roth, “Pipelines are safest for transportation of oil and gas,” *Manhattan Institute for Policy Research*, pp. 1–10, 2013.

[5] L. H. Lee, R. Rajkumar, L. H. Lo, C. H. Wan, and D. Isa, “Oil and gas pipeline failure prediction system using long range ultrasonic transducers and Euclidean-Support Vector Machines classification approach,” *Expert Systems with Applications*, vol. 40, no. 6, pp. 1925–1934, May 2013.

[6] C. Campos-Castellanos, Y. Gharaibeh, P. Mudge, and V. Kappatos, “The application of long range ultrasonic testing (LRUT) for examination of hard to access areas on railway tracks,” in *5th IET Conference on Railway Condition Monitoring and Non-Destructive Testing (RCM 2011)*, 2011, pp. 8B3–8B3.

[7] R. Caruana and A. Niculescu-Mizil, “An empirical comparison of supervised learning algorithms,” in *Proceedings of the 23rd international conference on Machine learning - ICML ’06*, 2006, pp. 161–168.

[8] E. Byvatov, U. Fechner, J. Sadowski, and G. Schneider, “Comparison of Support Vector Machine and Artificial Neural Network Systems for

Drug/Nondrug Classification," *Journal of Chemical Information and Computer Sciences*, vol. 43, no. 6, pp. 1882–1889, Jan. 2003.

[9] L. Vanajakshi and L. R. Rilett, "A comparison of the performance of artificial neural networks and support vector machines for the prediction of traffic speed," in *IEEE Intelligent Vehicles Symposium*, 2004, pp. 194–199.

[10] M. Vogt and V. Kecman, "Active-Set Methods for Support Vector Machines," in *Support Vector Machines: Theory and Applications*, vol. 158, no. 10, 2005, pp. 133–158.

[11] E. R. S. N. Y. Krylov, A. A. Bokserman, "The Oil Industry of The Former Soviet Union," *Addison Wesley, Massachusetts*, vol. 2nd Editio. 1998.

[12] J. a Beavers, J. a Beavers, N. G. Thompson, N. G. Thompson, C. C. Technologies, and C. C. Technologies, "External Corrosion of Oil and Natural Gas Pipelines," *Environments*, vol. 13, pp. 1015–1025, 2006.

[13] H. Gholami and M. J. Azizpour, "Stress Corrosion Crack Identification with Direct Assessment Method in Pipeline Downstream from a Compressor Station," *International Journal of Chemical, Nuclear, Metallurgical and Materials Engineering*, vol. 8, no. 12, pp. 1216–1219, 2014.

[14] P. R. Roberge, *Corrosion Inspection and Monitoring*. John Wiley & Sons, 2006, pp. 1–383.

[15] R. Bickerstaff, M. Vaughn, G. Stoker, M. . Hassard, and M. Garrett, "Review of Sensor Technologies for In-line Inspection of Natural Gas Pipelines," *Sandia National …*, pp. 1–10, 2002.

[16] P. R. Management, “Pipeline Risk Management,” *Pipeline and Hazardous Materials Safety Administration (PHMSA)*. [Online]. Available: http://primis.phmsa.dot.gov/. [Accessed: 02-Feb-2015].

[17] Koch, “Corrosion costs and preventive strategies in the United States. US Federal Highway Administration,” *Materials Performance*, vol. 41, no. 7 (cost of corrosion supplement), p. 12, 2002.

[18] “Gas line explodes in West Virginia; homes burn, freeway damaged - U.S. News,” *NBC News*, 2012. [Online]. Available: http://usnews.nbcnews.com/_news/2012/12/11/15845530-gas-line-explodes-in-west-virginia-homes-burn-freeway-damaged?lite. [Accessed: 02-Feb-2015].

[19] “NTSB releases details about Sissonville pipeline explosion investigation,” *MetroNews*, 2013. [Online]. Available: http://wvmetronews.com/2013/06/07/ntsb-releases-details-about-sissonville-pipeline-explosion-investigation/. [Accessed: 02-Feb-2015].

[20] J. a. Hammerschmidt, J. J. Goglia, and C. J. Carmody, “Natural Gas Pipeline Rupture and Fire Near Carlsbad , New Mexico,” pp. 1–57, 2003.

[21] M. S. Liew, E. S. Lim, K. L. Na, and N. F. M. Sidek, “Parametric Study on the Factors of External Corrosion of Offshore Pipelines in Malaysia 2 Fundamentals of Study,” *Recent Advances in Engineering*, pp. 260–265, 2013.

[22] W. Loy, “State and BP are still far from leak claim decision,” *Alaska Dispatch News*, 2012. [Online]. Available: http://www.adn.com/article/20120727/state-and-bp-are-still-far-leak-claim-decision. [Accessed: 02-Feb-2015].

[23] J. Roach, “Alaska oil spill fuels concerns over Arctic wildlife, future drilling,” *National Geographic News*, 2006. [Online]. Available:

http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Alaska+Oil+Spill+Fuels+Concerns+Over+Arctic+wildlife,+Future+Drilling#0. [Accessed: 02-Feb-2015].

[24] D. Joling, "BP agrees to $25M penalty for 2006 Alaska spills," *BusinessWeek*, 2011. [Online]. Available: http://www.businessweek.com/ap/financialnews/D9N087103.htm. [Accessed: 02-Feb-2015].

[25] "Map of northern Alaska showing location of Arctic National Wildlife Refuge, ANWR-en:1002 area, and the National Petroleum Reserve-Alaska (NPRA)." [Online]. Available: http://energy.usgs.gov/images/alaska/NPRA_F1lg.gif.

[26] "Gas pipeline explosion," *BorneoPost Online*, 2014. [Online]. Available: http://www.theborneopost.com/2014/06/11/gas-pipeline-explosion/. [Accessed: 02-Feb-2015].

[27] "Updated photos Petronas Bintulu-Kimanis gas pipeline explosion in Lawas," *Bintulu Weekly*, 2014. [Online]. Available: http://www.bintulu.org/2014/06/10/updated-photos-petronas-bintulu-kimanis-gas-pipeline-explosion-in-lawas.php. [Accessed: 02-Feb-2015].

[28] I. O. Fadeyibi, D. T. Omosebi, P. I. Jewo, and S. a Ademiluyi, "Mass Burns Disaster in Abule-egba, Lagos, Nigeria from a Petroleum Pipeline Explosion Fire.," *Annals of burns and fire disasters*, vol. 22, no. 2, pp. 97–103, Jun. 2009.

[29] P. L. Metropolo and a. E. P. Brown, "Natural gas pipeline accident consequence analysis," *Process Safety Progress*, vol. 23, no. 4, pp. 307–310, Dec. 2004.

[30] G. a. Papadakis, “Major hazard pipelines: A comparative study of onshore transmission accidents,” *Journal of Loss Prevention in the Process Industries*, vol. 12, no. 1, pp. 91–107, Jan. 1999.

[31] S. H. Gamidi, “Non Destructive Testing Of Structures,” *Master’s Thesis, Indian Institute of Technology, Bombay …*, 2009.

[32] R. Wirahadikusumah, D. M. Abraham, T. Iseley, and R. K. Prasanth, “Assessment technologies for sewer system rehabilitation,” *Automation in Construction*, vol. 7, pp. 259–270, 1998.

[33] H. Schempf, E. Mutschler, and B. Chemel, “Pipe inspection and repair system,” *US Patent*, 2004.

[34] R. Ales, W. Glime, and J. Hull, “Ultrasonic testing of fitting assembly,” *US Patent …*, 2009.

[35] “Ultrasonic thickness gauge,” *Ultrasonics*, 1970. [Online]. Available: http://www.hadcoservices.com/site27.html. [Accessed: 02-Feb-2015].

[36] D. N. Alleyne and P. Cawley, “Optimization of lamb wave inspection techniques,” *NDT & E International*, vol. 25, pp. 11–22, 1992.

[37] P. Cawley and M. Lowe, “Practical long range guided wave inspection-applications to pipes and rail’,” *Materials Evaluation*, pp. 66–74, 2003.

[38] R. B. Inspection, “Long Range Ultrasonic,” *NDT - Guided Wave, Teletest*. [Online]. Available: http://www.twi-global.com/technologies/ndt/advanced-ndt/long-range-ultrasonic-testing/. [Accessed: 02-Feb-2015].

[39] S. Lebsack, “Non-Invasive Inspection Method for Unpiggable Pipeline Sections,” *Pipeline and Gas Journal*, vol. 143, pp. 58–64, 2009.

[40] Plan Integrity Limited, “Teletest Focus Goes Permanent.” [Online]. Available: www.plantintegrity.co.uk.

[41] P. Wilcox, M. Lowe, and P. Cawley, “The effect of dispersion on long-range inspection using ultrasonic guided waves,” *NDT & E International*, vol. 34, pp. 1–9, 2001.

[42] “Guided wave testing - Wikipedia, the free encyclopedia.” [Online]. Available: http://en.wikipedia.org/wiki/Guided_wave_testing. [Accessed: 02-Feb-2015].

[43] T. R. Hay, *Low Cost Steel Bridge Pile Inspection Technology*. 2009.

[44] C. Ennaceur, P. Mudge, and T. Gan, “Long Range Inspection of Engineering Assets Using Guided Ultrasonic Waves,” *17th World Conference on Non-Destructive Testing*, 2008.

[45] V. Desai, M. Pal, M. Banjare, and C. Nancharaiah, “Use Of Long Range Ultrasonic Testing (LRUT) Technique For Health Assessment Of Critical Piping In LPG Service In A Petroleum Refinery,” *212.8.206.21*.

[46] L. Zhang, “A Multi-channel Transmitter System for NDT Pipeline Testing using Guided Waves Ultrasound,” in *5th International CANDU In-Service Inspection Workshop*, 2014.

[47] P. Catton, P. Mudge, and W. Balachandran, “Advances in defect characterisation using long-range ultrasonic testing of pipes,” *Insight: Non-Destructive Testing and Condition Monitoring*, vol. 50, pp. 480–484, 2008.

[48] L. Zhang, W. Luo, and J. L. Rose, “Ultrasonic guided wave focusing beyond welds in a pipeline,” *AIP Conference Proceedings*, vol. 820 I, pp. 877–884, 2006.

[49] A. Wilkinson, "Long range inspection and condition monitoring of rails using guided waves," *51st Annual Conference of the British Institute of Non-Destructive Testing*, pp. 1–11, 2012.

[50] T. Stepinski, "Structural Health Monitoring of Piping in Nuclear Power Plants - A Review of Efficiency of Existing Methods," *Structural Health Monitoring*, 2011.

[51] W. N. Heo, H. T. Lim, T. G. Kim, and M. S. Choi, "Approach for the Realtime Received Signal Processing in Magnetostrictive Long-Range Ultrasonic Testing," 2012.

[52] P. Rizzo, I. Bartoli, A. Marzani, and F. Lanza di Scalea, "Defect Classification in Pipes by Neural Networks Using Multiple Guided Ultrasonic Wave Features Extracted After Wavelet Processing," *Journal of Pressure Vessel Technology*, vol. 127, p. 294, 2005.

[53] H. G. Ramos, T. Rocha, J. Král, D. Pasadas, and A. L. Ribeiro, "An SVM approach with electromagnetic methods to assess metal plate thickness," *Measurement: Journal of the International Measurement Confederation*, vol. 54, pp. 201–206, 2014.

[54] X.-G. Z. X.-G. Zhang, J.-J. X. J.-J. Xu, and G.-Y. G. G.-Y. Ge, "Defects recognition on X-ray images for weld inspection using SVM," *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*, vol. 6, 2004.

[55] A. Bernieri and G. Betta, "A novel biaxial probe implementing multifrequency excitation and SVM processing for NDT," … *(I2MTC), 2013 IEEE …*, 2013.

[56] C. Wan, A. Mita, and T. Kume, “An automatic pipeline monitoring system using sound information,” *Structural Control and Health Monitoring*, vol. 17, pp. 83–97, 2010.

[57] D. Isa and R. Rajkumar, “Pipeline Defect Prediction Using Support Vector Machines,” University of Nottingham, 2009.

[58] a. Bernieri, L. Ferrigno, M. Laracca, and M. Molinara, “Crack Shape Reconstruction in Eddy Current Testing Using Machine Learning Systems for Regression,” *IEEE Transactions on Instrumentation and Measurement*, vol. 57, 2008.

[59] M. Steinbach, G. Karypis, V. Kumar, and Others, “A comparison of document clustering techniques,” *KDD workshop on text mining*, vol. 400, pp. 525–526, 2000.

[60] G. H. Ball and D. J. Hall, “A clustering technique for summarizing multivariate data.,” *Behavioral science*, vol. 12, pp. 153–155, 1967.

[61] U. Maulik and S. Bandyopadhyay, “Performance evaluation of some clustering algorithms and validity indices,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 1650–1654, 2002.

[62] I. Ylakoski and I. Ylakoski, “Unsupervised classification of ultrasonic NDT data,” *Proceedings of the SPIE---The International Society for Optical Engineering*, vol. 2345, pp. 182–186, 1994.

[63] T. Kohonen, “Self-organitation and Associative Memory.,” *Self-Organization and Associative Memory, 100 …*, 1984.

[64] P. O. Box, F.- Hut, J. Vesanto, and J. Himberg, *SOM Toolbox for Matlab 5 Juha Vesanto , Johan Himberg , SOM Toolbox Team Helsinki University of*

Technology Report A57 April 2000 ISBN 951-22-4951-0 ISSN 1456-2243 Libella Oy Espoo 2000 SOM Toolbox for Matlab 5. 2000, pp. 0–59.

[65] "Kohonen Self Organizing Maps," *mnemstudio.org*. [Online]. Available: http://mnemstudio.org/neural-networks-kohonen-self-organizing-maps.htm. [Accessed: 02-Feb-2015].

[66] B. Conan-guez and F. Rossi, "Fast Algorithm and Implementation of Dissimilarity Self-Organizing Maps arXiv : 0709 . 3461v1 [cs . NE] 21 Sep 2007," *Neural Networks*, 2006.

[67] E. Berglund and J. Sitte, "The parameter-less SOM algorithm," *Proc. ANZIIS*, pp. 159–164, 2003.

[68] L. Hamel, *Knowledge Discovery With Support Vector Machines*. 2009.

[69] V. Vapnik, E. Levin, and Y. Le Cun, "Measuring the VC-Dimension of a Learning Machine," *Neural Computation*, vol. 6, pp. 851–876, 1994.

[70] O. Bousquet, S. Boucheron, and G. Lugosi, *Statistical Learning Theory*. 2004.

[71] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, vol. 16. 2002.

[72] C. J. C. Burges, ". ' ' 1-43 () A Tutorial on Support Vector Machines for Pattern Recognition," *Data mining and knowledge discovery*, vol. 43, 1998.

[73] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel based learning methods*. 2000, p. 190.

[74] M. Khelil, M. Boudraa, a Kechida, and R. Drai, "Classification of defects by the SVM method and the principal component analysis (PCA)," *World Acad. Sci. Eng. Technol*, vol. 9. pp. 226–231, 2005.

[75] J. Yin, Q. Yang, and J. Pan, "Sensor-based abnormal human-activity detection," *Knowledge and Data Engineering, …*, 2008.

[76] O. Chapelle, "Support vector machines for histogram-based image classification," *Neural Networks, IEEE …*, vol. 10, pp. 1055–64, 1999.

[77] D. C. D. Chen, H. Bourlard, and J.-P. Thiran, "Text identification in complex background using SVM," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 2, 2001.

[78] I. Goethals, K. Pelckmans, J. Suykens, and B. De Moor, "Identification of MIMO Hammerstein models using least squares support vector machines," *Automatica*, 2005.

[79] C. Mu, R. Zhang, and C. Sun, "LS-SVM predictive control based on PSO for nonlinear systems," *Control Theory & Applications*, 2010.

[80] a J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[81] N. Akram, N. Shafiabady, and D. Isa, "Using Neural Networks as Pipeline Defect Classifiers," *IT Convergence and Security (…*, 2013.

[82] T. Bartz-Beielstein, "How experimental algorithmics can benefit from Mayo's extensions to Neyman-Pearson theory of testing," *Synthese*, vol. 163, pp. 385–396, 2008.

[83] W. Sun and Y. Yuan, *Optimization theory and methods: nonlinear programming*. 2006.

[84] P. a. Jensen and J. F. Bard, "Operations research models and methods," p. 266, 2003.

[85] J. Nocedal and S. J. Wright, *Numerical Optimization*. 1999, p. 625.

[86] M. Vogt, K. Spreitzer, and V. Kecman, "Identification of a high efficiency boiler by Support Vector Machines without bias term," *System Identification*, 2003.

[87] C. L. Karr and L. M. Freeman, *Industrial applications of genetic algorithms*. 1999, p. 350.

[88] J. H. Holland, *Adaptation in Natural and Artificial Systems: An introductory analysis with applications to biology, control, and artificial intelligence.*, vol. Ann Arbor. 1975, pp. 1–228.

[89] R. Eberhart and Y. Shi, "Comparison between Genetic Algorithms and Particle Swarm Optimization," *Lecture Notes in Computer Science: Evolutionary Programming VII*, vol. 1447, pp. 611–616, 1998.

[90] D. Beasley, D. R. Bull, and R. R. Martin, "An Overview of Genetic Algorithms : Part 1, Fundamentals," *University Computing*, vol. 2, pp. 1–16, 1993.

[91] S. Forrest, "Forrest 1993 - Genetic Algorithms- Principles of natural selection applied to computation.pdf," *Science*, 1993.

[92] S. Sivanandam and S. Deepa, "Genetic Algorithm Optimization Problems," *Bentley Applied Research - Bentley Communities*, 2008. [Online]. Available: http://link.springer.com/content/pdf/10.1007/978-3-540-73190-0_7.pdf. [Accessed: 03-Feb-2015].

[93] A. A. Arkadan, S. Member, T. Sareen, and S. Member, "Genetic Algorithms for Nondestructive Testing in Crack Identification," *IEEE Transactions on Magnetics*, vol. 30, pp. 4320–4322, 1994.

[94] D. S. K. D. S. Kim, H.-N. N. H.-N. Nguyen, and J. S. P. J. S. Park, "Genetic algorithm to improve SVM based network intrusion detection system," *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)*, vol. 2, 2005.

[95] D. N. Alleyne, M. J. S. Lowe, and P. Cawley, "The Reflection of Guided Waves From Circumferential Notches in Pipes," *Journal of Applied Mechanics*, vol. 65, p. 635, 1998.

[96] W. Zhu, J. L. Rose, J. N. Barshinger, and V. S. Agarwala, "Ultrasonic Guided Wave NDT for Hidden Corrosion Detection," *Research in Nondestructive Evaluation*, vol. 10, pp. 205–225, 1998.

[97] K. Woodsend, "Using interior point methods for large-scale support vector machine training," *Philosophy*, p. 162, 2009.

[98] D. M. J. Tax and R. P. W. Duin, "Support Vector Data Description," *Machine Learning*, vol. 54, pp. 45–66, 2004.

[99] R. Koggalage and S. Halgamuge, "Reducing the number of training samples for fast support vector machine classification," *Neural Information Processing-Letters and Reviews*, vol. 2, pp. 57–65, 2004.

[100] C. Hsu and C. Lin, "A comparison of methods for multiclass support vector machines," *Neural Networks, IEEE Transactions on*, vol. 13, pp. 415–425, 2002.

[101] G. Cauwenberghs and T. Poggio, "Incremental and Decremental Support Vector Machine Learning," *Learning*, vol. 13, p. 409, 2001.

[102] S. Vishwanathan, A. Smola, and M. Murty, "SimpleSVM," 2003.

[103] R. L. Haupt and S. E. Haupt, “Optimum Population Size and Mutation Rate for a Simple Real Genetic Algorithm that Optimizes Array Factors,” *Applied Computational Electromagnetics Society Newsletter*, vol. 15, pp. 94–102, 2000.

[104] a H. Wright, “Genetic algorithms for real parameter optimization,” *Foundations of Genetic Agorithms*, pp. 205–220, 1990.

[105] C. Chang and C. Lin, “LIBSVM : A Library for Support Vector Machines,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, pp. 1–39, 2011.

[106] D. Morariu and L. Vintan, “A Better Correlation of the SVM Kernel ’ s Parameters,” *Proceedings of the 5th RoEduNet IEEE International Conference, ISBN (13)*, pp. 978–973, 2006.

[107] D. Mattera, F. Palmieri, and S. Haykin, “Simple and robust methods for support vector expansions.,” *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, vol. 10, pp. 1038–1047, 1999.

[108] D. Mattera, F. Palmieri, and S. Haykin, “An explicit algorithm for training support vector machines,” *IEEE Signal Processing Letters*, vol. 6, 1999.

[109] Casale, P., Pujol, O., & Radeva, P. (2012). Personalization and user verification in wearable systems using biometric walking patterns. Personal and Ubiquitous Computing, 16(5), 563-580.

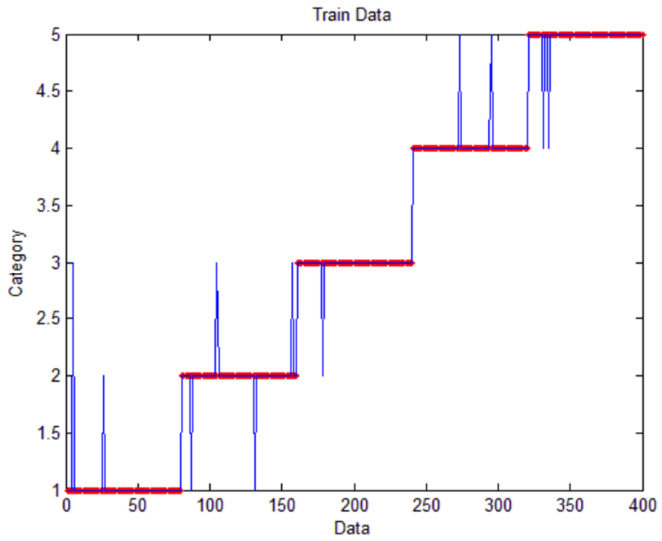
APPENDICES

Appendix A: Classification performance using Neural Network (NN).

Train data contained 400 data samples and test data contained 100 data samples. The data is labeled into 5 classes according to the defect depth. Table below shows the Mean Squared Error (MSE) that were achieved from five runs.

Different Runs	First Run	Second Run	Third Run	Fourth Run	Fifth Run
Train MSE	0.0340	0.0272	0.0258	0.0296	0.0249
Test MSE	0	0.0100	0.0100	0.0100	0.0100

Figure below shows the graph for the NN classification using training data. Blue line represents the class by NN and red line represents the correct label.



This work has been published in:

Akram, N. A., Shafiabady, N., & Isa, D. (2013, December). Using Neural Networks as Pipeline Defect Classifiers. In *IT Convergence and Security (ICITCS), 2013 International Conference on* (pp. 1-4). IEEE.

Appendix B: Artificial data experiment on Incremental Support Vector Machine

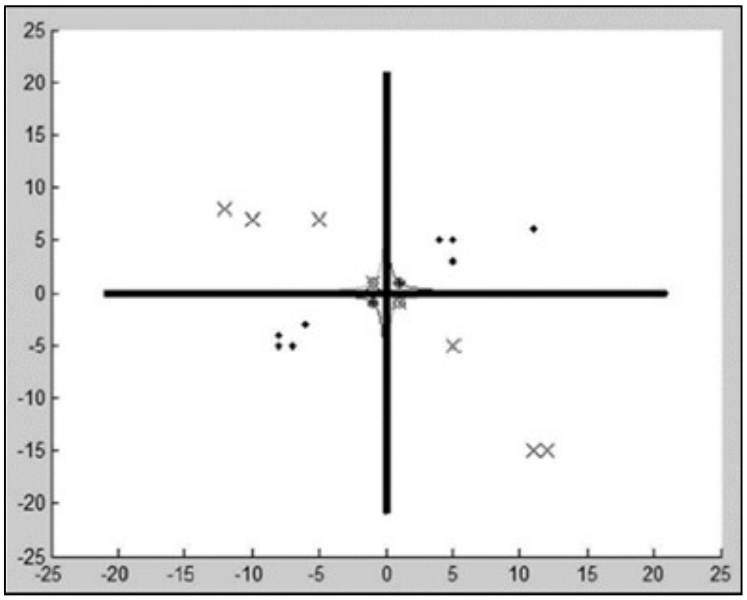
Exclusive-Or (XOR) problem dataset is chosen to investigate on how the proposed incremental training SVM updates the optimal hyperplane for each data captured. The experimental data is a two-dimensional data as shown in Table 3. Polymonial kernel is used as the XOR problem cannot be solved using a linear hyperplane.

Table below shows the XOR dataset.

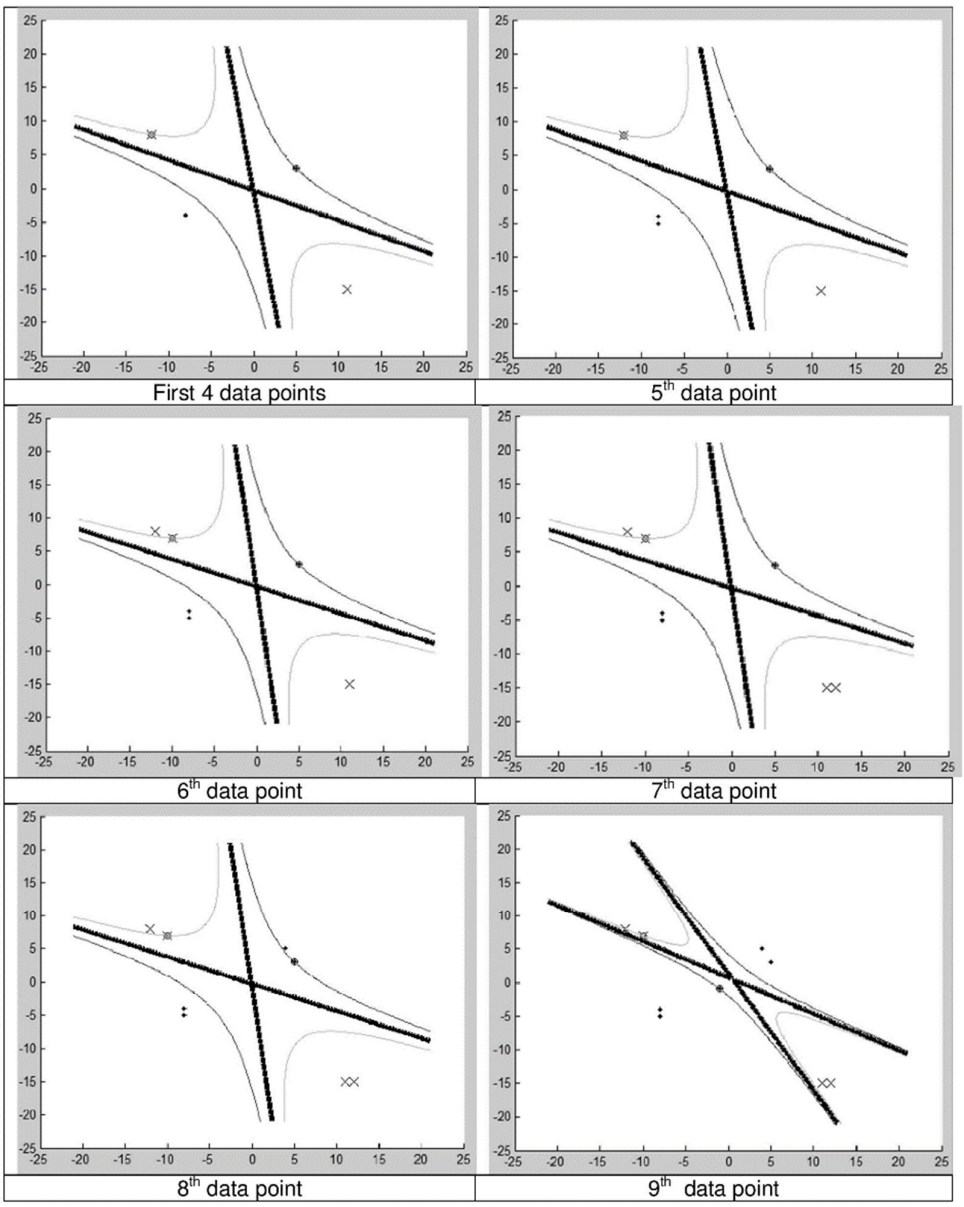
Data #	X_1	X_2	**D**
1	-8	-4	-1
2	-12	8	1
3	11	-15	1
4	5	3	-1
5	-8	-5	-1
6	-10	7	1
7	12	-15	1
8	4	5	-1
9	-1	-1	-1
10	-5	7	1

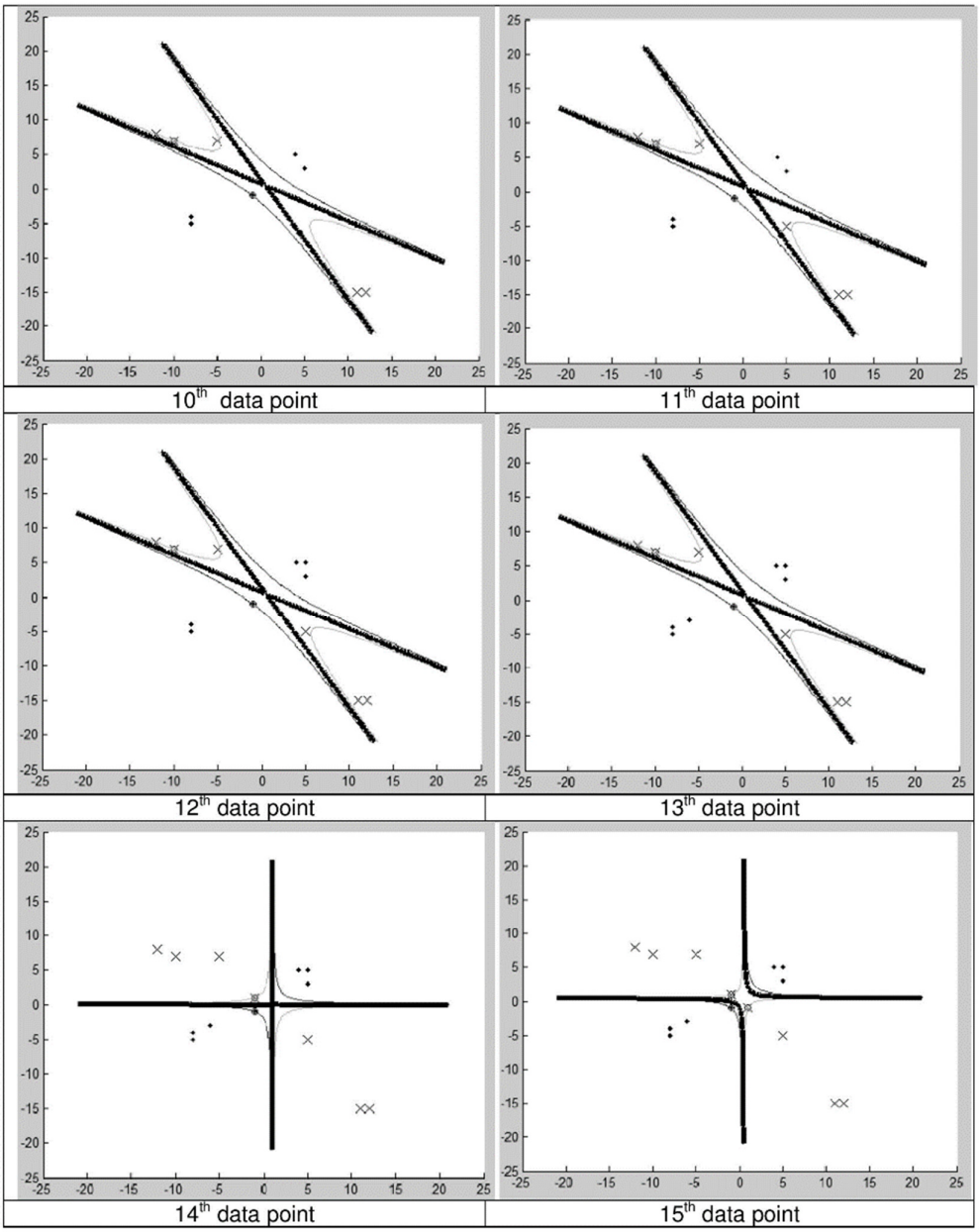
11	5	-5	1
12	5	5	-1
13	-6	-3	-1
14	-1	1	1
15	1	-1	1
16	11	6	-1
17	-7	-5	-1
18	-5	7	1
19	1	1	-1
20	5	-5	1

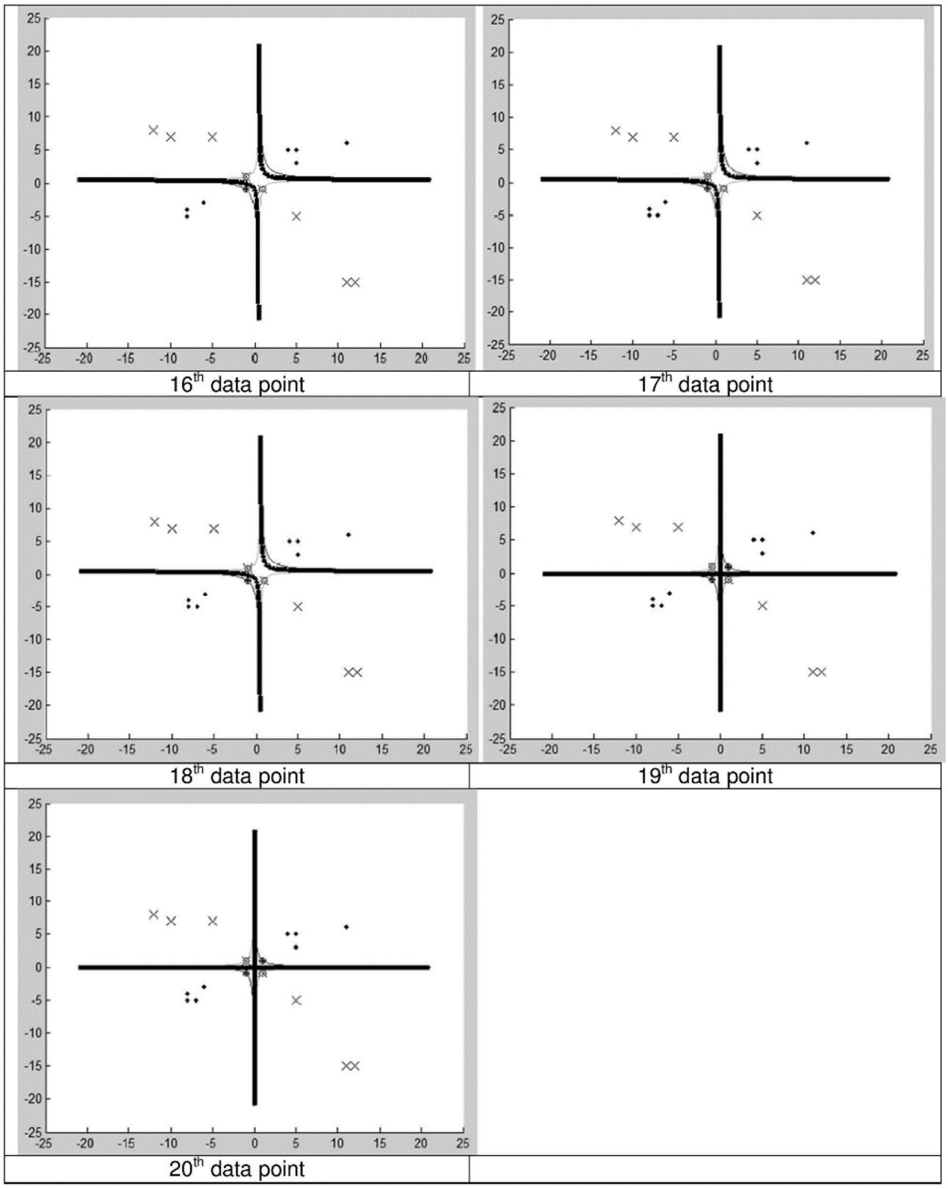
Figure below shows the hyperplane constructed by batch training SVM.



Figures below shows the change of hyperplane for each data captured.




10th data point
11th data point
12th data point
13th data point
14th data point
15th data point

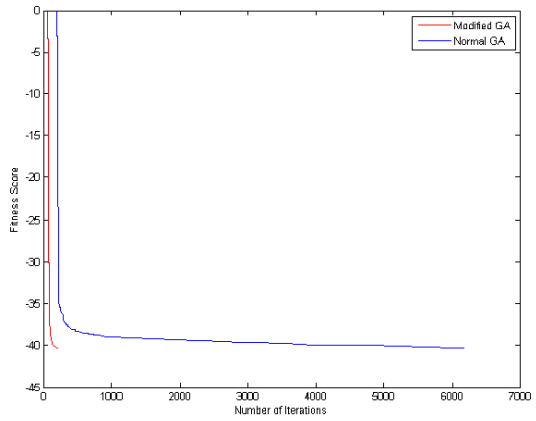


As a conclusion, the results shows that proposed incremental training SVM technique is able to produces a matching final hyperplane as batch training SVM. Thus, the incremental training SVM sequence data implementation is empirically proven to match the accuracy of the conventional batch training SVM.

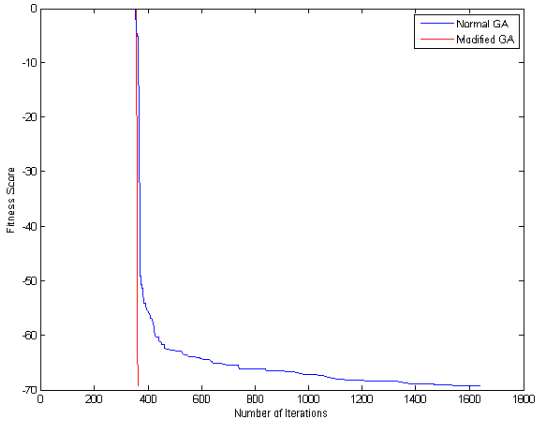
This work has been published in:

Akram, Nik Ahmad, et al. "Active incremental Support Vector Machine for oil and gas pipeline defects prediction system using long range ultrasonic transducers." *Ultrasonics* 54.6 (2014): 1534-1544.

Appendix C: Result on Standard Dataset using Proposed Method



Fitness score vs iterations for Normal GA and Modified GA SVM for URINARY dataset



Fitness score vs iterations for Normal GA and Modified GA SVM for XOR dataset.

Appendix D: Matlab Codes

All to All Clustering

File List:

1. Multipass.m

```
function [AllMean, Label] = Multipass(inputData)

%% Find Correlation
% tic

CorrelationMatrice = pdist2(inputData,inputData,'cityblock');

% toc
%% Remove and replace redundant matrice's entries
UnsortMatrice = CorrelationMatrice;

% Replace diagonal with big value
UnsortMatrice(eye(size(UnsortMatrice))~=0)=999;
% tic
%% Sort Matrice by Row

% Find Minimun Correlation for each element
[value, index] =min(UnsortMatrice,[],2);

% Concatenate matrices and add index number.
tempSortedMatrices = [value (1:length(index))' index];

% Sort According Row #1, Ascending
SortedMatrices = sortrows(tempSortedMatrices, 1);
% toc
%% Label the elements
% tic
%Initial Label Number
Label = zeros(length(index),1);
LabelNumber = 1;

MaxD = max(SortedMatrices (:,1));

for k = 1:length(index)
  if SortedMatrices (k,1) > 1 *MaxD
                Label(SortedMatrices (k,2)) = LabelNumber;
            Label(SortedMatrices (k,3)) = LabelNumber;
            LabelNumber = LabelNumber + 1;
  else

  switch Label(SortedMatrices (k,2))
```

```
    case 0
       switch Label(SortedMatrices (k,3))
          case 0
             Label(SortedMatrices (k,2)) = LabelNumber;
             Label(SortedMatrices (k,3)) = LabelNumber;
             LabelNumber = LabelNumber + 1;

          otherwise
             Label(SortedMatrices (k,2)) = Label(SortedMatrices (k,3));
       end

    otherwise
       Label(SortedMatrices (k,3)) = Label(SortedMatrices (k,2));
  end
%     SortedMatrices (k,1)
  end

end


% toc
%% Compute Mean For Each Cluster

for  L = 1:LabelNumber-1

   index = Label == L;
   ClusterMean = mean(inputData(index,:));
   AllMean(L,:) = ClusterMean;

end

end
```

2. LabelCluster.m

```
function [FinalLabel] = LabelCluster(inputData,iteration)
%% Find size of input Data
LabelArray = [];
[sizeM, TEMP ] = size(inputData);


%% Use Multipass.m to find Label and Mean

Mean = inputData;
for longrun = 1:iteration


   [Mean, Label] = Multipass(Mean);

   save testsave.mat;
```

```
   if longrun == 1
      tempLabel = Label;
   else

   for q = 1:sizeM
      tempLabel(q,1)= Label(tempLabel(q));
   end

   end



end
FinalLabel = tempLabel;
```

3. AddNoise.m

```
Result = [];
for SNR = 0:1:50

   xs = awgn(x,SNR,'measured',100000);
   xts = awgn(xt,SNR,'measured',10000);
   ML = LabelCluster(xs,3);
   pecahan(y,ML);
   yA = yAdjust(y,ML);
   pecahan(y,yA);
   model=svmtrain(yA,xs,'-q');
   [predicted_label, accuracy, decision_values] = svmpredict(yt,xts,model);

   temp1 = [SNR accuracy(1)];
   Result = [Result;temp1];

end
```

Incremental SVM

File List:

IncrementalSVM.m

```
function [accuracy,resultime] = inlineSVM(Y,X)

%% Load Data

% load('xyrand.mat');
% load('data.mat');
gamma = 0.0179
%
%
% X = [x(1:100, :);x(101:200, :)];
% % X = Xrand;
% xt = [xt(1:400, :);xt(401:800, :)];
%
% % yt = [y(1:100, :);y(101:200, :)];
% yt = [yt(1:400, :);yt(401:800, :)];
%
% Y = [ones(100,1);-1*ones(100,1)]; %data.y ;
% % Y = Yrand;
% D2 = [ones(400,1);-1*ones(400,1)];
resultime = [];
%% Make Sure get 1 plus 2 different class for first itteration

lenY = length(Y);

i = 1;
counter1 = 0;
counter2 = 0;
while( ~(counter1 > 1 && counter2 > 1 && ( counter1+counter2 > 3)) )

   switch Y(i)
      case -1
         counter1 = counter1+1;
      case 1
         counter2 = counter2+1;
   end

i = i + 1;
end
tic
%% Find sv
index = [1:i-1]';
[alphay,svind,b] = getSV(X(index,:),Y(index));

index = index(svind);

% Evaluate next datapoint
for j = i:lenY
   j

 K = (gamma * X(svind,:) * X(i,:)').^2 ;   %%% Kernel Poly
%  K = (X(svind,:) * X(i,:)') ;   %%% Kernel Linear
%  K = exp(- gamma * pdist2(X(svind,:),X(i,:))) ;   %%% Kernel RBF

f = alphay(svind)' * K + b;

%% Rules:)
```

```
if abs(f) < 9999
   index = [index;i];
   i = i + 1;
   [alphay,svind,b] = getSV(X(index,:),Y(index));
   index = index(svind);
else
%   index = index(svind);
   i = i + 1;
end



timeout = toc;
tempresult = [j,timeout];
resultime = [resultime;tempresult];

end

%% Accuracy Test

% K_test = (X(svind,:) * xt') ; %kernel linear
K_test = (gamma * X(svind,:) * X').^2 ; %kernel poly
%    K_test =  exp(- gamma * pdist2(X(svind,:),X)) ; %kernel RBf
f_test = alphay(svind)' * K_test + b;

f_test = f_test';

len = length(f_test);
len2 = length(Y);

result = zeros(len,1);
accuracyresult = zeros(len2,1);

for m = 1:len
   if f_test(m,1) < 0
      result(m,1) = -1;
   else
      result(m,1) = 1;
   end
end

result
for n = 1:len2
   if isequal(result(n,1),Y(n,1))
      accuracyresult(n,1) = 1;
   else
      accuracyresult(n,1) = 0;
   end
end

accuracy = length(find(accuracyresult == 1))/len2 * 100
```

2. findLambdaPK.m

```
function [lambda,pk,XnP] = findlambdapk(W,X,H,f,A,b)

   l = length(W);

   K = [H,A(W,:)';
   A(W,:),zeros(l,l)];
   Lx = pinv(K)*[-f;b(W,:)];

   [XnP] =  Lx(1:length(X))
   lambda = Lx(length(X)+1:length(Lx));

   pk = XnP - X';

end
```

3. findAkw.m

```
function [ak,W] = findakW(WAll,A,pk,W,b,X)

Wa = WAll;
Wa(W,:) = [];

Atemp = A;
Atemp(W,:) = [];


btemp = b;
btemp(W,:) = [];


rule1 = Atemp*pk;
ruleout = find(rule1 <0 );

Atemp(ruleout,:) = [];
btemp(ruleout,:) = [];
Wa(ruleout,:) = [];


Otemp = ((btemp-Atemp*X')./(Atemp*pk));
Otemp(Otemp < 0 )= [];
aksub =  min(Otemp);

ak = min([1,aksub]);

   if ak < 1
      g = Atemp*(X'+(ak*pk)) == btemp;
      rule2 = g==1;              % looking for the constraint that bind witht ak
      Wa = Wa(rule2,:);
```

```
    W = [W; Wa];
  end
end
```

4. qp.m

```
function [x0] = qp(H,f,A,b)


%H = [2 0; 0 2];
%f = [-8; -6];
%A = [-1 0; 0 -1; 1 1];
%b = [0; 0; 5];

WAll = (1:length(A))';      % find  index of constraints
W = [1:WAll]';              % Initial working set

%W = [1:2]';
x0 = zeros(1,length(f));    % Initial point

it = 0;

while ( 1 )

  it = it + 1;

  [lambda,pk,XnP] = findlambdapk(W,x0,H,f,A,b);  % call function to find
lambda and pk

  x0; W; pk; lambda;


  if( abs(pk) == 0 )

    if isempty(lambda)
      break
    end

    if min(lambda) >= 0
      break
    else

      [~,k] = sort(lambda);

      W(k(1)) = [];
      x0 = XnP';

    end

  else

    [ak,W] = findakW(WAll,A,pk,W,b,x0);   % call function to find ak
    ak;
```

```
      x0 = x0+(ak*pk)';

   end
end
```

5. getSV.m

```
function [alphay,svind,b] = getSV(X,D)

gamma = 0.0179;
n = numel(D);

A = [D';
   diag(-ones(n,1));
   diag(ones(n,1))];



b = [0;
   zeros(n,1);
   ones(n,1);];



fixcon= 0.0001;
C = 1;
Dd = diag(D);


% K = (X*X');
  K=(gamma*X*X').^2; %% poly kernel
% K = exp(- gamma * pdist2(X,X));

opts = optimoptions('quadprog','Algorithm','active-set','Display','off');

alpha = quadprog(Dd*K*Dd, - ones(n,1), ...
              [], [], ...
              D', 0, ...
              zeros(n,1), C * ones(n,1), ...
              [], opts) ;

%  alpha = qp(Dd*K*Dd, -ones(n,1), A, b)'; %%% CHANGE qp function with
your own QP... i use same syntax with quadprog see doc quadprog in matlab


svind = find(alpha > fixcon * C) ;
alphay = Dd * alpha ;



 r = 1 - alphay' * K * Dd ;
 act = ismember(1:n, svind) ;
 pos = D' > 0 ;
```

```
 maxb = min([+r(pos & act),  -r(~pos & ~act)]) ;
 minb = max([-r(~pos & act), +r(pos & ~act)]) ;

   b = mean([minb maxb]) ;

end
```

Genetic Algorithm as QP Solver in SVM

File List:

1. GAasQPSVM.m

```
clear all;
global bestGenmin;
global bestGenmean;
bestGenmin = [];
bestGenmean = [];
load 0vs1mm.mat;

D = Y;

gamma = 0.0178571429;

C = 10;


n = numel(D);
Dd = diag(D);
fixcon = 0.001;

% K=(1+X*X').^2;
% K=X*X';
 K =  exp(- gamma * pdist2(X,X)) ; %kernel RBf

H=Dd*K*Dd;

f= - ones(n,1);


fx = @(x)param_fitness(x,f,H);

LB = ones(n,1) * 1e-6;
UB = C * ones(n,1);

nvars = n;    % Number of variables


% options = gaoptimset('MutationFcn',@mutationadaptfeasible);
% options = gaoptimset(options,'CrossoverFcn',@crossoverintermediate);
% % options = gaoptimset(options,'CreationFcn', @gacreationlinearfeasible);
```

```
options = gaoptimset('PlotFcns',{@gaplotbestf2},'Display','iter');
% options = gaoptimset(options,'PopulationSize',30);
% X0 = ones(1,n)* 0.5; % Start point (row vector)
% options = gaoptimset(options,'InitialPopulation',X0);
% options = gaoptimset(options,'OutputFcns',@gaoutputgen);
options = gaoptimset(options,'Generations',100,'StallGenLimit', 100);
tic
[x,fval,exitflag,output,population,scores]  =
ga(fx,nvars,[],[],D',0,LB,UB,[],options);


index = x > 0;

nvars2 =  length(index);
lenTo = sum(index);

LB2 = ones(lenTo,1) * 1e-10;k
UB2 = C * ones(lenTo,1);


fx2 = @(x)param_fitness2(x,f,H,index,nvars);
ConstraintFunction = @(x)simple_constraint(x,index,nvars,D);

X0 = x(index); % Start point (row vector)
options = gaoptimset('InitialPopulation',X0);
options = gaoptimset(options,'PlotFcns',{@gaplotbestf2},'Display','iter');
% options = gaoptimset(options,'TolCon',1e-13,'TolFun',1e-13);
[x2,fval2,exitflag2,output2,population2,scores2]  =
ga(fx2,lenTo,[],[],[],[],LB2,UB2,ConstraintFunction ,options);
toc

alpha = zeros(1,nvars);
alpha(index) = x2;

alpha = alpha';

svind = find(alpha > 0) ;
model.alpha = alpha ;
alphay = Dd * alpha ;

r = 1 - alphay' * K * Dd ;
act = ismember(1:n, svind) ;
pos = D' > 0 ;

maxb = min([+r(pos & act),  -r(~pos & ~act)]) ;
minb = max([-r(~pos & act), +r(pos & ~act)]) ;

b = mean([minb maxb]) ;

%  K_test = (1 + X(svind,:) * XT').^2 ; %kernel poly
%  K_test = (X(svind,:) * XT') ; %kernel linear
 K_test =  exp(- gamma * pdist2(X(svind,:),XT)) ; %kernel RBf
f_test = alphay(svind)' * K_test + b;

f_test = f_test';
```

```
len = length(f_test);
len2 = length(YT);

result = zeros(len,1);
accuracyresult = zeros(len2,1);

for m = 1:len
  if f_test(m,1) < 0
     result(m,1) = -1;
  else
     result(m,1) = 1;
  end
end

for t = 1:len2
  if isequal(result(t,1),YT(t,1))
     accuracyresult(t,1) = 1;
  else
     accuracyresult(t,1) = 0;
  end
end

accuracy = length(find(accuracyresult == 1))/len2 * 100;
```

2. paramFitness.m

```
function y = param_fitness(xk,f,H)
y = 0.5*xk*H*xk' + xk*f;
```

3. plotIt.m

```
n = numel(D);
Dd = diag(D);
fixcon = 0.001;

K=(1+X*X').^2;
H=Dd*K*Dd;

f= - ones(n,1);

svind = find(alpha > fixcon * C) ;
% model.alpha = alpha ;
alphay = Dd * alpha ;
```

```
 r = 1 - alphay' * K * Dd ;
 act = ismember(1:n, svind) ;
 pos = D' > 0 ;

 maxb = min([+r(pos & act),  -r(~pos & ~act)]) ;
 minb = max([-r(~pos & act), +r(pos & ~act)]) ;

   b = mean([minb maxb]) ;

ur = linspace(-8,8,256) ;
[u,v] = meshgrid(ur) ;
X_dense = [u(:)' ; v(:)'] ;

X = X';

K_dense = (1+  X(:,svind)' * X_dense).^2 ;   %%% polymonial kernel
%  K_dense = (X(:,svind)' * X_dense) ;
f_dense = alphay(svind)' * K_dense + b ;
f_dense = reshape(f_dense, size(u,1),size(u,2)) ;

clf;
hold on ;
imagesc(ur,ur,f_dense) ; colormap jet ; hold on ;
[c,hz] = contour(ur,ur,f_dense,[0 0]) ;
hg  = plot(X(1,D>0), X(2,D>0), 'g.', 'markersize', 15) ;
hr  = plot(X(1,D<0), X(2,D<0), 'r.', 'markersize', 15) ;
hko = plot(X(1,svind), X(2,svind), 'ko', 'markersize', 5) ;

[c,hm] = contour(ur,ur,f_dense,[-1 -1]) ;
set(hm,'color', 'r', 'linestyle', '--') ;
[c,hp] = contour(ur,ur,f_dense,[+1 +1]) ;
set(hp,'color', 'g', 'linestyle', '-.') ;
[c,hz] = contour(ur,ur,f_dense,[0 0]) ;
set(hz,'color', 'k', 'linewidth', 4) ;

legend([hg hr hko hz ], ...
      'Positive Vector', 'Negative Vector', 'Support Vector', ...
      'Hyperplane', ...
      'location', 'northeastoutside') ;

% colorbar('location','southoutside')
```