Smartphone Application for Fault Recognition

Nishchal K. Verma, Sumanik Singh, Jayesh K. Gupta,

Rahul K. Sevakula, Sonal Dixit, Al Salour

Dept. of Electrical Engineering, Indian Institute of Technology, Kanpur, India
Boeing Company, St. Louis, MO, USA
E-Mail: nishchal@iitk.ac.in, sumanik@iitk.ac.in, jayeshkg@iitk.ac.in,
srahulk@iitk.ac.in, dsonal@iitk.ac.in, al.salour@boeing.com

Abstract— **Smart-phones have become an integral part of our daily life. Even though smart-phone app development has boomed, only few general purpose applications exist for feature extraction, feature selection and classification of audio data from a smart-phone. The paper presents detailed theory behind the data mining model used, which has given good results on MATLAB. The application was made to learn different fault states of an industrial air compressor. The application was tested to recognize the fault state in real time as the air compressor was running. It has performed very well with classification accuracies above 93.73%. It is believed that similar application and model with some minor changes in specifications can be used for acoustic pattern recognition in wide range of fields; even in industry.**

Keywords- feature extraction; android; feature classification; feature selection; support vector machines

I. Introduction

Usage of smart-phones has indeed taken a great surge. With the increasing pervasiveness and processing power of the smart-phones, a lot of work which required a separate computer and data acquisition tools, can now be done by a single device - a smart-phone. This certainly saves on skilled man-power required for processing and collection of data, as even an unskilled worker can use a smart-phone and get results. Moreover, earlier one has to transfer the collected data to a separate computer which a smart-phone allows us to bypass.

Smart-phones are still in their nascent development state. Yet with the juggernaut of Moore's law being applicable here too, their processing power can only increase. Hence they have already started displacing the computer in most daily tasks. It only requires a little nudge to see some area of work being done by a computer to be implemented on a smart-phone. Specially, industrial activities have a lot of scope in implementing a number of tasks on a smart-phone. One of them is machine fault-detection.

Although a lot of work has been done on Android in the field of image processing [1], speech and music analysis [2] or even acoustic environmental classification [3] we did not come across any applications for the use of industry. This is quite understandable, as these smart-phones were mainly designed for personal use rather than industry. But the very strength of ease of use can have many industrial applications too, as we show here.

In this paper we present a model that can be used for acoustic pattern recognition. This model uses various feature extraction techniques, Principal Component Analysis (PCA) as feature reduction/selection algorithm and Support Vector Machine (SVM) as the classifier algorithm. All these modules have been implemented on Android which will enable a smartphone to recognize the different classes based on acoustic signals in real time.

The rest of the paper is organized as follows. In Section II, we explain the data mining model used in our application. In Section III, details of implementation on Android platform has been discussed. In Section IV, we describe results. Section V gives the conclusion of our work.

II. Data Mining Model

In the following, we describe the general outline of the theory behind our application. After getting sampled acoustic data, some pre-processing steps are performed for reducing the effect of various kinds of noises, present during recording operation. The preprocessed data is then fed to the feature extraction module.

A. Feature Extraction Module

Feature extraction is a form of dimensionality reduction, where one tries to represent the entire input data in the form of a set of features.

1) Time Domain:

Signals and waveforms are generally present in the time domain. But since time domain representation is a very basic way of representing the data, not much inference can be drawn [4]. Nevertheless, eight statistical parameters are extracted from time domain. They are absolute mean, maximum peak, root mean square, square mean root, variance, kurtosis factor, crest factor, shape factor and skewness of data as exposited in [8]. For details see Table I.

2) Frequency Domain:

Frequency domain representation can convey special characteristics of signal which otherwise are not obvious in the time domain. The frequency domain features show how the signal energy varies as a function of frequency. For studying the properties of the signal in frequency domain, the time domain signal needs to be converted to frequency domain by

TABLE I. TIME DOMAIN FEATURES

Absolute Mean	$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} \lvert x_i \rvert$
Maximum Peak Value	$x_p = \max \lvert x_i \rvert$
Root Mean Square	$x_{rms} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$
Square root value	$x_r = \left(\frac{1}{N}\sum_{i=1}^{N}\sqrt{x_i}\right)^2$
Variance	$var = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$
Kurtosis	$b = \frac{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^4}{s^4}$
Crest Factor	$C_f = \frac{x_p}{x_{rms}}$
Shape Factor	$S_f = \frac{x_{rms}}{\bar{x}}$

using Fast Fourier Transform (FFT). FFT allows us to obtain a spectrum of frequency components which can be divided into 8 equal consecutive segments known as bins. From these bins we can get one feature from each bin, by dividing the individual bins' energy with total spectral energy. This way we get 8 features from frequency domain.

3) *Wavelet Domain*

In order to analyze non stationary signals like the ones we work on, it is better to work on time-frequency domain rather than only time or only frequency domain. In time-frequency domain, you get a rough picture of how frequency spectrum changes with time. Wavelets are a class of powerful tools that helps in analyzing the signal in time-frequency domain. [5, 6].

i) Morlet Wavelet Features [7]: A Morlet wavelet is a cosine signal decaying exponentially on both sides. Mathematically it is defined as:

$$y_{a,b}(t) = e^{\frac{-b^2(t-b)^2}{a^2}} \cos\left(\frac{p(t-b)}{a}\right) \qquad (1)$$

where (1) is the son wavelet obtained by dilating and translating the mother wavelet by *a* and *b* respectively. The signal is first transformed from time domain into the time-frequency domain by convolving it with the Morlet wavelet. Standard Deviation, Wavelet Entropy, Kurtosis factor, Skewness, Variance, Zero crossing rate and Sum of peaks are calculated from the resultant wavelet coefficients as detailed in [8]. This way, 7 more features are obtained.

ii) Discrete Wavelet Transform (DWT): It is another powerful tool in digital signal processing [9]. It decomposes the signal using low pass and high pass filters. There are several families of these filters like Haar, Daubechies[9], Coifflets etc. The signal is convoluted with the filters to obtain two sets of coefficients. The convolutions expression is given by:

$$h[n] = \sum_m f[n-m] * g[m] \qquad (2)$$

where, *h[n]* is convolved output of functions *f* and *g*. The coefficients obtained from low pass filter are known as approximation coefficients and those obtained from high pass filter are known as detail coefficients.

The transform follows a one sided tree like structure as shown in Figure 1. In this figure, X refers to the signal; cA_i and cD_i are the approximation and detail coefficients respectively of ith level. The approximation coefficients obtained are further decomposed to get new set of approximation and detail coefficients. On the other hand, detail coefficients are not decomposed further.

Daubechies-4 (db4) decomposition filter has been used for our model. Both, db4 low pass and db4 high pass filter consists of 8 coefficients each as shown in Table II.

The level of decomposition depends on the signal and the task to be performed. In our model, the signal is decomposed till 6th level which results in six detail coefficients and one approximation coefficient. The detail coefficients at level 1, 2 and 3 have higher frequency content as compared to the detail coefficients at level 4, 5 and 6. The initial decomposition levels have more abrupt variations as compared to the higher decomposition levels. These observations are useful in the extraction of features from DWT as out of the 9 features

TABLE II. DAUBECHIES COEFFICIENTS

S.No.	**Daubechies 4 Low Pass Filter Coefficients**	**Daubechies 4 High Pass Filter Coefficients**
0	-0.0105974018	-0.2303778133
1	0.0328830117	0.7148465706
2	0.0308413818	-0.6308807679
3	-0.1870348117	-0.0279837694
4	-0.0279837694	0.1870348117
5	0.6308807679	0.0308413818
6	0.7148465706	-0.0328830117
7	0.2303778133	-0.0105974018

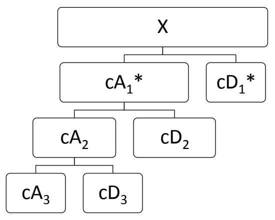

Figure 1. DWT Tree (Level 3 Decomposition)
$*cA_i$: ith Approximation coefficient $*cD_i$: ith Detail coefficient

extracted, three are variances of the detail coefficients at level 1, 2 and 3. For finding the next 3 features, auto-correlation of detail coefficients at level 4, 5 and 6 is performed. Variance of these auto-correlation coefficients at the three levels forms the next three features. Auto-correlation is a mathematical tool for finding repeating patterns, thus capturing similarity within the observations.

$$R_{xx}(l) = \sum_{n=i}^{N-|k|-l} x[n] * x[n-l] \qquad (3)$$

where, l is lag, R_{xx} is the autocorrelation.

Three more features were found by taking the absolute mean of smoothened versions of detail coefficients at level 1, 2 and 3. The following averaging filter is used for smoothing the rapid fluctuations of the detail coefficients:

$$y(k) = \frac{\sum_{l=k-m}^{k+m} |x(l)|}{2m} \qquad (4)$$

where, m decides the degree of smoothness which was 5 in our case.

iii) Wavelet Packet Transform (WPT): One drawback of wavelet transform is its poor frequency resolution in the high frequency region, making it difficult to discriminate between signals having close high-frequency components. Wavelet packets give alternative bases that are formed by taking linear combination of the usual wavelet functions [6]. While DWT involves decomposition of approximation coefficients only, WPT decomposes both - approximation as well as detail coefficients; thus making a complete binary tree as compared to the one-sided tree like structure of DWT. This is shown in Figure 2.

So if we combine all the above features, we will have 286 features in total.

B. Feature selection

Feature selection is a process commonly used in data mining applications for selecting the most relevant features from data. This improves performance of the model by alleviating the curse of dimensionality and speeding up the learning process. We used Principal Component Analysis (PCA) [10] for selecting the best set of features from the given features. We selected PCA because it is simple, effective and has been successful in many fields. PCA basically converts the current feature space to a space of uncorrelated basis functions.

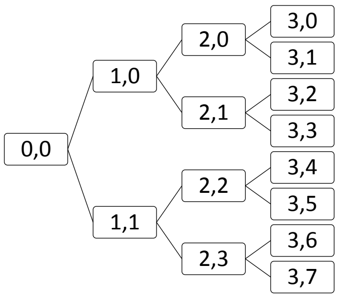

Figure 2. WPT Tree (Level 3 Decomposition)

These basis functions are found from the eigenvectors of the co-variance matrix of input dataset. These eigenvectors form the basis functions of the new space. These are also known as principal components, which are uncorrelated. The eigenvalues indicate the variance in the direction of the eigenvectors and so the principal components are ranked on the basis of these eigenvalues. The input data is transformed to the new space.

Our goal is to get a space that is uncorrelated and which maintains most of the variance. The number of top ranked features (in the new space) is decided according to the percentage of cumulative energy required.

C. Classification with Support Vector Machines and Fuzzy Decision Function

1) Support Vector Machines(SVM)

It is a supervised learning method used for classification. SVM intends to find the optimal hyper plane which maximizes the minimum margin between the two classes. By Statistical Learning Theory, we learn that maximal margin assures lower VC dimension, thus giving good generalization [11]. SVM therefore is popular for giving a good generalized classifier. The objective function for SVM is as follows:

$$\left.\begin{aligned} &\min_{\cdot\xi,w,b} < w.w > + C\sum_{i=1}^{l} \xi_i^{\,2} \\ &\textit{subject to} \quad y_i(< w.x_i > + b) \geq \xi_i, \quad i = 1,.....,l \end{aligned}\right\} (5)$$

As it is a convex problem, it gives global and unique solution. By applying kernel tricks, this can be extended to non-linear classification. We have used radial basis function as the kernel function with appropriate gamma parameter. The standard SVMs were designed for binary classification, but can be applied to multi-class too using one against one, one against all techniques [12] [14].

III. IMPLEMENTATION

All experiments were done on a smartphone having 830MHz ARMv6 processor and 290MB RAM with Android OS v2.3 as the operating system. It is expected to have much better results on the higher end smart phones. The application was used for recognizing the running conditions of air compressor present in the workshop of Electrical Eng. Dept., IIT Kanpur. Specifications of air compressor on which data was recorded are as follows:

- Air pressure range-0-500 lb/m2, 0-35 Kg/cm2
- Induction Motor: 5 HP, 415V, 5Amp, 3ph, 50 Hz, 1440 rpm
- Pressure Switch: Type: PR-15, Range: 100-213 PSI, 7-15 Kg/cm2

Audio recordings were done at 44.1 kHz sampling rate. Dataset consist of 3 classes, 1500 instances (500 for each class). The three classes were for three different conditions of machine namely

a) Healthy condition: As the name suggests, the compressor is healthy.

b) LIV faulty condition: It occurs due to damaged inlet valve which directly affects pressure oscillations.

c) LOV faulty condition: It occurs when outlet valve is damaged, which leads to increased time consumption in the filling of air tank at required pressure level.

Though we used the pre-recorded audio recordings for building models for classification and for performance analysis, but our application is flexible enough to perform acoustic signal based fault recognition in real time.

For implementing some mathematical functions in Java, Commons Math package: The Apache Commons Mathematics Library [15] is used. This java library consists of various methods for numerical analysis, used in engineering and math.

A. Data Acquisition

Goal of data acquisition is to get raw sampled data of the acoustic signal. We worked on pre-recorded data in wav format. This format was chosen, because it is one of the few formats which do not compress the acoustic data, hence making it easier to get the sampled raw data. The *'.wav'* format stores the data in 16 bit linear PCM format in 2's compliment. By writing appropriate code and using classes namely *"AudioRecorder class"* and *"WavFile class"*, we could get the sampled data from the .wav file for our further processing.

B. Pre-processing

We used a pre-existing module built at our lab, which worked on the sampled data with the main aim of removal of noise and effect of outliers. This was achieved by first down sampling the data by a factor of 2. Then clipping was performed to reduce the size of sample, by selecting the 1 second window with minimum standard deviation. Further the data was filtered by a Butterworth low pass filter to remove high frequency noise and finally the data was normalized between 0 and 1.

C. Feature Extraction

In the time domain, the aforementioned eight statistical parameters were calculated. The algorithms used for calculation of these variables were compatible with the algorithms used in MATLAB. This allowed us to quickly check and compare our results with the corresponding in-built MATLAB functions.

In frequency domain, for calculating Discrete Fourier Transform, Apache Commons Math Library which uses Cooley-Tukey FFT (Fast Fourier Transform) algorithm was used. This is a divide and conquer algorithm that recursively breaks down DFT problem into smaller DFT problems. So it calculated 256 point DFT and divided it into 8 bins of 32 size.

For wavelet domain, we extracted seven features from the Morlet wavelet analysis as mentioned in Section III, nine features from the DWT analysis and 254 features from WPT.

For DWT, autocorrelation is required for calculating 3 features. Autocorrelation was calculated using Wiener-Khinchin theorem because it offers quicker implementation than the brute force method based on the definition of autocorrelation (i.e. going by the expression of autocorrelation). The brute force algorithm has *$O(n^2)$* time complexity and Wiener-Khinchin theorem has *$O(n \log(n))$* time complexity. Wiener-Khinchin theorem calculates autocorrelation using Fourier Transform and Inverse Fourier Transform. Steps involved are first calculating the Fast Fourier Transform of the data i.e.

$$F[k] = \sum_{n=0}^{N-1} x[n] \times e^{-i2\pi \frac{k}{N} n} \tag{6}$$

then multiplying the result with its conjugate,

$$S[n] = F[n] * \overline{F[n]} \tag{7}$$

and finally, finding the inverse Fast Fourier Transform.

$$R[n] = \frac{1}{N} \sum_{k=0}^{N-1} S[k] \times e^{i2\pi \frac{k}{N} n} \tag{8}$$

At each level of decomposition, convolution of the signal with the filters is followed by dyadic decimation (or down-sampling). Here down sampling refers to considering even indexed elements and leaving other elements. Down sampling is done to reduce size of signal in our application. As size of signal was a very critical issue otherwise it would have caused the heap size memory error.

In case of WPT, we decompose signal upto level 7 because decomposition after 7th level would cause redundancy. Moreover higher decomposition level would have created java heap size error. In level 7 decomposition total number of nodes in the tree will be 255 and energy will be extracted from all nodes except the input node. Therefore a total of 254 features are extracted from WPT. Each node is also called a packet. Thus there were 127 packets each of approximation and detail coefficients. We used the Daubechies-4 filter for analysis which is same as the one used for DWT. Energy calculated at each node is given by

$$E_i = \sum_{n_i} x[n]^2 \tag{9}$$

where, E_i is energy at ith node and n_i is number of points belonging to the ith node.

D. Feature Selection

As mentioned in Section II, we applied principal component analysis on the features extracted from earlier recorded files. Training set of 1200 samples was considered where each sample corresponds to one wav-file recording. Though the application can be programmed to select any number of features required, we decided to work on selecting 23 feature set from the data as previous experiments done in our labs had shown it to be the best compromise and contains more than 99.9% of energy. The application writes the selected features in comma separated format along with the eigen matrix, scaling factors and mean values to their respective files on an SD-card. This was done so as to allow for cross-validation of data and using the same parameters on test data. The results are shown in Table IV.

TABLE III. FEATURE EXTRACTION PROCESSING TIME

File	Size	Time
Healthy/Reading1.wav	1.12 MB	12 s
Lov/Reading1.wav	1.14 MB	12 s
Liv/Reading1.wav	1.14 MB	12 s
Healthy/Reading2.wav	1.14 MB	12 s
Lov/Reading1.wav	1.14 MB	12 s
Liv/Reading2.wav	1.14 MB	12 s

E. Feature Classification

For SVM classification the LIB-SVM java library was used, which implements a SMO type algorithm for classifying data into different classes [16].

The SVM module works in the following modes:

i. Parameter Search: In this mode the application cross validates for finding the best possible cost and gamma parameters. The application also gives an option where user can manually enter the parameter values.

ii. Training: In this mode, the application builds an SVM model for later prediction.

iii. Classification: In this mode, the application uses the stored classifier model for predicting the classes of an unknown test data file and writes the result to a file.

LIB-SVM library was integrated in our android application by writing our own interface to the main library. The user-interface consisted of three buttons for the respective modes: Cross Validation, Training and Classification and two edit boxes for manually entering the cost and gamma parameters.

IV. RESULTS

The simple user interface of our application is shown in Figure 3. The button “Extract Features”, “Select Features”, “Search best Parameters” would perform Feature Extraction, Feature Selection and Parameter search respectively, when clicked. Next page of the application gives options for training the feature selection model and classifier model.

The processing time for reading and extracting features from single audio recordings are given in Table III. Table IV gives the processing time for training, i.e. building models for PCA (Feature selection), Parameter search and SVM classifier (classifier model). As can be seen in the table, feature selection gets an input dataset having 286 features which it then converts to a dataset of 23 features. SVM classifiers which include parameter search as well, gets their input with 23 features (after applying PCA). Though the processing times for training are quite large, this needs to be done only once. While performing acoustic recognition on real time, the testing time for a single recording was 58 seconds. As we worked on a low end smartphone, the training times will be significantly reduced on higher end phones which have higher processing capability. We cross-validated all our results with the corresponding prebuilt modules made in our lab on MATLAB. The results were found to be identical.

For classification purposes, we worked on data after feature selection i.e. samples having 23 features. Further details of the classification module are as follows:

1) Parameter Search: Using the LIBSVM’s train function, we built a cross validation module which checked for the best Cost and Gamma parameters for training. Unfortunately the validation time was long, so we had to trim the range under which it searched for the parameters between -4 and +4. Even then, the results were unsatisfactory as can be seen from Table IV.

2) Training: We worked on the features selected by PCA. Using the LIB-SVM’s algorithm, the model is built and stored as model file for prediction which is finally stored in SD card for future use.

3) Testing: For a test audio file, the file is read and features are extracted to a feature vector. The feature vector is mean-centered using the mean values calculated during training, which is then multiplied by the transform matrix generated from PCA. The transformed matrix, which has 23 features, is passed to classifier (LIBSVM) module which performs classification using the already stored model.

   We performed 5 fold cross validation to the 1500 sample database, i.e. 1200 samples for training and 300 samples for testing. The performance results w.r.t. recognizing LOV fault and LIV fault were very encouraging as shown in Table V. The average accuracy was 93.73 % for LOV data and 99.87 % accuracy for LIV data.

TABLE IV. MODEL BUILDING DETAILS

Operation Performed	Number of Samples	Number of Attributes	Time
Feature Selection	1200	286	471 s
Parameter Search	1200	23	10851 s
Classifier model building	1200	23	71 s

TABLE V. ACCURACY RESULTS OF APPLICATION

Cross-validation Data Set	LOV Accuracy	LIV Accuracy
1	92.33 %	100 %
2	94.33 %	100 %
3	97.33 %	100 %
4	94.67 %	100 %
5	90.00 %	99.67 %

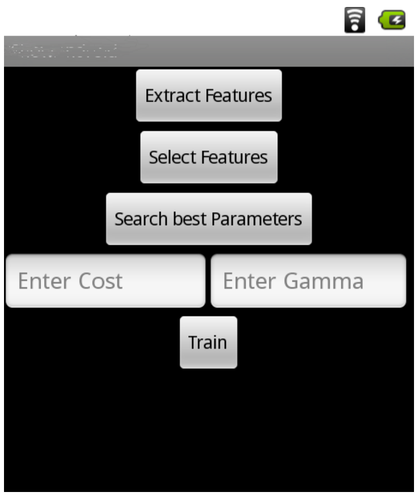

Figure 3. Model Building Interface

V. Conclusion

This paper described an Android implementation of a general purpose audio data-mining application using apache-commons math and LIBSVM libraries. The application running on smartphone is complete in itself, as it gives facility to perform all operations i.e. from data acquisition to training of classifier model. It performed very well with accuracies of 93.73% and 99.87% in recognizing the faulty states of air compressor. The application with some small changes in specifications can be extended and used for acoustic pattern recognition in many applications. In the future, PCA can be implemented on the native layer of Android to speed up the process of analysis. Also we would like to check the performance by using different feature selection algorithms like Independent Component Analysis and algorithms using Mutual Information amongst features.

Acknowledgment

We would like to express or sincere gratitude towards Boeing for their funding support for this research work through research grant (EE/BOEING/20100220), at IIT Kanpur.

.

References

[1] Charalampos Doukas, Ilias Maglogiannis. A fast mobile Face Recognition System for Android OS Based on Eigenfaces Decomposition. In Artificial Intelligence Applications and Innovations, IFIP Advances in Information and Communication Technology, Volume 339, pp. 295-302, ISBN 978-3-642-16238-1, 10.1007/978-3-642-16239-8_39.

[2] J. Saunders, "Real-time discrimination of broadcast speech/music," Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on , vol.2, no., pp.993-996 vol. 2, 7-10 May 1996

[3] L. Ma, B. Milner, and ***B***.Smith. Acoustic environment classification. ACM Transactions on Speech and Language Processing (TSLP), 3(2):1–22, 2006

[4] Q. He, F. Kong, R. Yan,"Subspace-based gearbox condition monitoring by kernel principal component analysis" Mechanical Systems and Signal Processing, 21 (4) (2007), pp. 1755–1772

[5] I. Daubechies, Ten lectures on wavelets, CBMS-NSF conference series in applied mathematics (1992). SIAM Ed.

[6] B. Liu, S.-F. Ling, Q.F. Meng, Machinery diagnostics based on wavelet packets, *JVC/*Journal *of* Vibration *and* Control 3 *(*1997*)* 5–17.

[7] Jin Ming, "Feature Extraction based on Morlet wavelet and its application for mechanical fault diagnosis," Journal of sound and vibration, Volume 234, 2000, pp. 135-148.

[8] Qingbo He, Ruqiang Yan, Fanrang Kong, Ruxu Du, Machine condition monitoring using principal component representations, Mechanical Systems and Signal Processing, Volume 23, Issue 2, February 2009

[9] S.K. Goumas, M.E. Zervakis, G.S. Stavrakakis, "Classification of washing machines vibration signals using discrete wavelet analysis for feature extraction," Instrumentation and Measurement, IEEE Transactions on , vol.51, no.3, pp.497-508, Jun 2002

[10] I. T. Jolliffe. Principal Component Analysis. New York:Springer Verlag, 1986

[11] C. Cortes, V.Vapnik , "Support-vector networks", Machine Learning, Volume 20,Issue 3, 1995, pp. 273-297

[12] A. Mathur, G.M. Foody,, "Multiclass and Binary SVM Classification: Implications for Training and Classification Users," Geoscience and Remote Sensing Letters, IEEE , vol.5, no.2, pp.241-245, April 2008

[13] N.K.Verma, A.Roy, A.Salour, "An optimized fault diagnosis method for reciprocating air compressors based on SVM," System Engineering and Technology (ICSET), 2011 IEEE International Conference on 27-28 June 2011, pp.65-69.

[14] Achmad Widodo, Bo-Suk Yang, Support vector machine in machine condition monitoring and fault diagnosis, Mechanical Systems and Signal Processing, Volume 21, Issue 6, August 2007, Pages 2560-2574

[15] Commons Math Developers, Apache Commons Math, Release 2.2 Available: http://commons.apache.org/math

[16] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.

Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.