Condition Monitoring Using Accelerometer Readings

Cheryl Danner, Kelly Gov, Simon Xu
Stanford University

***Abstract*—In the field of condition monitoring, machine learning techniques are actively being developed to detect and/or predict faults in equipment. In this paper, the authors prototype a process to monitor the condition of rotating machinery using accelerometer readings. Time domain and frequency domain features are used to categorize data samples, with performance compared for three supervised learning algorithms: logistic regression, Naive Bayes, and SVM. Two unsupervised learning algorithms, k-means and mixture of Gaussians, are investigated for use in filtering data initially and identifying machine malfunctions. Logistic regression gives the lowest error rate in tests run on the available dataset, classifying all test samples correctly using time or frequency features. A combination of k-means and mixture of Gaussians algorithms shows potential for filtering data and identifying machine malfunctions.**

I. Introduction

In the industrial and manufacturing world, a vibration analyst's job is to diagnose problems with rotating equipment. Thus, analysts must familiarize themselves with typical spectra of various machinery in order to identify machine malfunction and distinguish bad channels/erroneous readings.

For our project, we plan to use machine learning techniques to develop a few aspects of a condition monitoring algorithm: characterizing typical states for different types of equipment and identifying erroneous readings. Performance on these tasks will be compared using time domain features and frequency domain features.

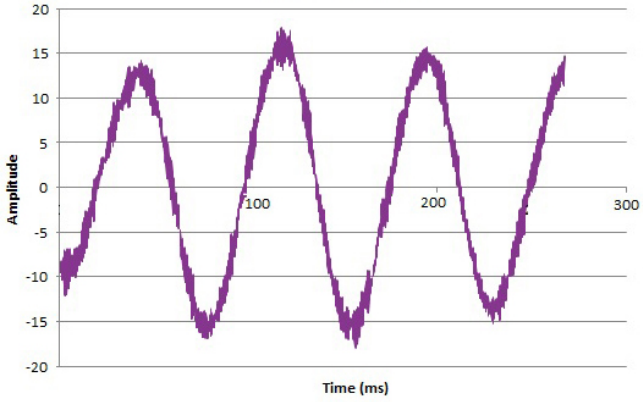
II. Related work

In the new and developing field of condition monitoring and predictive maintenance, many different approaches have been explored by companies and researchers. Some groups use support vector machines for classification [1], others use Bayesian models, and still others use artificial neural networks (ANNs) [2]. Using a combination of techniques can lead to a more accurate solution, especially as machines can be highly variable; not only are the machines different, but they can have different operating conditions. Although there is a lot of interest in using ANNs to do predictive maintenance, it takes a long time to train, and the "black box" nature of the neuron node matrix limits traceability. Since vibration science is highly empirical, having a way to identify the reasoning behind an algorithms decision is very valuable. In this nascent stage, using some other techniques would be helpful for troubleshooting, but as condition monitoring becomes more advanced, ANNs may become a good choice. Outside of machine learning, other alarm techniques set bandpass filters around the frequencies that are indicative of faults [3]. This is good for certain fault patterns, such as the raised harmonics present in mechanical looseness, but not so good for other common problems, such as rolling bearing wear, which are present in subharmonic frequencies. While there are simpler techniques, a trained machine learning algorithm can be the most effective at classifying machinery problems from spectral patterns.

III. Dataset and Features

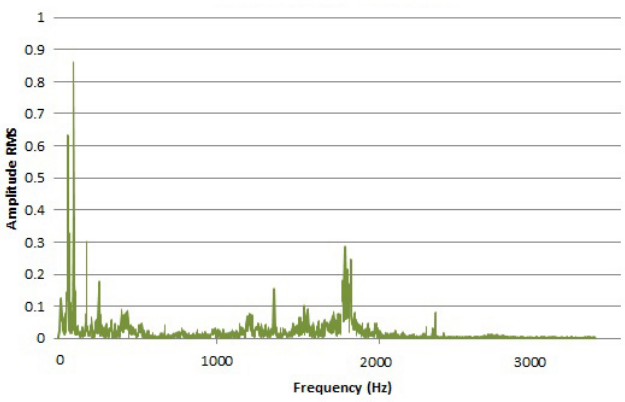
To train and test our models, we have obtained several datasets from different rotating machines, labeled M1, M3, M4, and SN41, which refer to a feed pump, a condensate pump, the motor driving the condensate pump, and a shaker motor, respectively. The data obtained is raw time history data consisting of approximately 4096 time-points per sample. This data has been post-processed into acceleration and velocity in each of the X, Y, and Z directions. We have further applied a Fast Fourier Transform (FFT) to convert the data into the frequency spectrum. Figures 1 shows unprocessed accelerometer measurements, and Figure 2 shows FFT output for the SN 41 machine.

Fig. 1. Example of waveform data from SN41 machine



Two classes of features were used: time domain features and frequency domain features. In the time domain, RMS, peak, kurtosis, and crest factor were calculated from experimental accelerometer time history data for both acceleration and velocity in each of the X, Y, and Z directions for a total of 24 time domain features. These time domain features were

Fig. 2. Example of frequency spectrum of data from SN41 machine



selected for being relatively straightforward to calculate and their utility in other condition monitoring studies [4] [5].

In the frequency domain, the power spectra for acceleration and velocity in each of the X, Y, and Z directions were calculated with a resolution of 2 Hz resulting in 10,002 features. To reduce the number of frequency domain features, correlation was done using Octave's built-in function 'corr'. Features which exhibited low correlation with machine type were discarded in our model. This proved to be problematic in the implementation of Naive Bayes, as the features that remained were those with strong peaks in the M3 data. Due to the implementation of multinomial Naive Bayes, this biased the algorithm toward predicting M3, and it was unable to function properly. This issue may be avoidable by normalizing the features so that the large magnitude difference in frequency features between different machines is no longer a factor. After correlation analysis and experimentation, we restricted the frequency domain features to the Z-velocity set (1667 features), as this provided sufficient information to create accurate prediction models.

IV. Data Filtering

After starting work with the original dataset (M3 and M4 machines only), we discovered anomalies in the data. These anomalies are unsurprising, as the company is currently still in the development phase of using low cost sensors which can diminish data fidelity. Although it cannot be conclusively determined whether data from a particular measurement was erroneous or not, we used a heuristic to filter the data. If the Z velocity RMS lands within a range of values, the measurement is considered to be correct; values near zero and extremely high values are considered erroneous. When data points classified as incorrect using this RMS filtering method were discarded, performance of algorithms to discriminate across machine types (discussed in Section V) improved significantly. For the M3 and M4 machines, a large percentage of samples were discarded, but the M1 and SN41 machines had most of their data retained, as shown in Table I.

TABLE I
Data Filtering Using Threshold for Z Velocity RMS

	M1	M3	M4	SN41
Total samples	3370	2325	1478	2170
# retained	2940	632	565	1673
# filtered out	880	1693	913	497
% filtered out	26%	73%	62%	23%

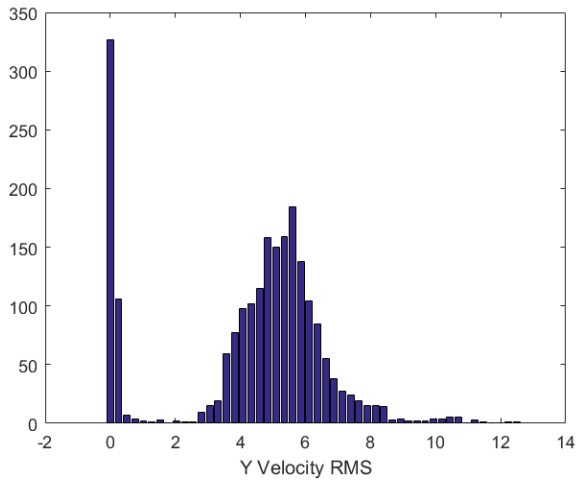
Although it would be possible to have an expert select upper and lower thresholds to implement RMS filtering for each machine, alternative approaches may be able to lower the amount of manual work.

A. Methods

1) K-means: One approach uses k-means to cluster the data and select the cluster that is closest to expected measurement values. Here, X, Y, and Z velocity RMS were used because clusters using these features mapped most closely to the results from the manual RMS filtering method. The expected values are defined by selecting the peak bin from a histogram of the full data set while ignoring the extreme highest and lowest bins. For example, Figure 3 shows the histogram of Y velocity RMS values for the SN41 machine, where the expected Y velocity value was determined to be 5.6 mm/s. For each machine, the k-means algorithm was run 100 times, and the cluster with a centroid closest to the expected values was declared to be correct measurements.

2) Mixture of Gaussians: The same procedure was implemented using mixture of Gaussians as the clustering algorithm instead of k-means. This used the EM algorithm to converge to a solution. Since the converged solution depends on an initial guess, this is best identified by using the best k-means result to create a starting point.

Fig. 3. Histogram of Y velocity RMS values for SN41 machine



B. Results

Since it is not known which data samples are correct versus erroneous, an error rate against the correct labeling cannot

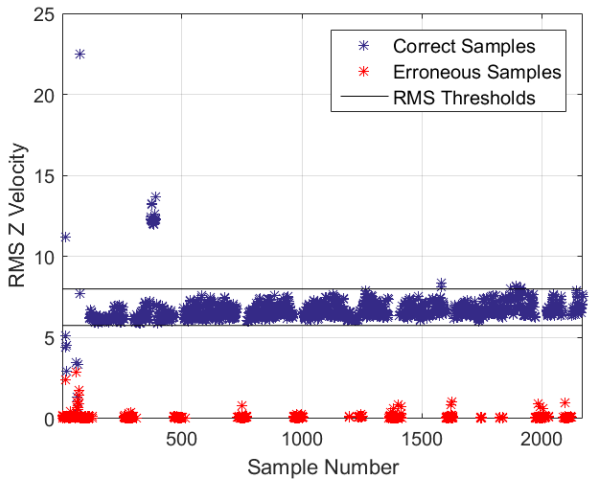
be calculated. Instead, classification results from the k-means and mixture of Gaussians methods were compared against the labeling produced by the RMS filtering method. The resulting error rates in Table II can be interpreted as measuring the algorithms' ability to match results produced by a human. The k-means clustering was better able to achieve this objective, with error rates of less than 2% for M3, M4, and SN41 machines. The automatic filtering was not able to closely match the RMS filtering for the M1 machine, which had a much larger spread of values compared to the other machines. The large discrepancy with the M1 data is mirrored by a large distance from the expected value, so there is at least an indicator of poor performance.

TABLE II
DISAGREEMENT BETWEEN DATA FILTERING ALGORITHMS VERSUS MANUAL RMS FILTERING

	M1	M3	M4	SN41
K-means	23%	1.8%	1.4%	1.8%
Mixture of Gaussians	17%	6.6%	6.2%	0.32%

Figure 4 shows sample Z RMS velocity values over time for k-means with the manually selected upper and lower bounds used for filtering overlaid. Figure 5 displays the same clusters plotted using the X and Y RMS velocity values, the two other features used for clustering. The green dot shows the expected values for correct measurements based on measurement histograms.

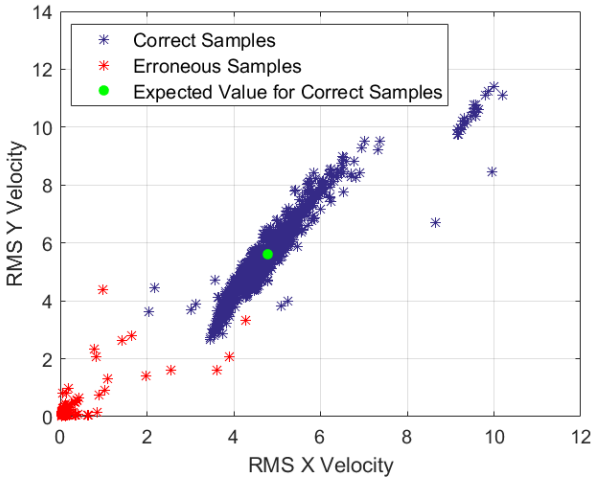
Fig. 4. K-means clustering for the SN41 machine with manually selected RMS thresholds overlaid as black lines



V. CLASSIFICATION

Ideally, a database containing labeled examples of machine performance under different faulty conditions would be available to train learning algorithms. However, such an extensive data set is not currently available for the set of machines being examined. One of the four machines (SN41) was labeled as malfunctioning for 12 data points. Because

Fig. 5. K-means clustering for the SN41 showing X and Y velocity values with a green point representing expected values overlaid



there was insufficient data to classify various fault states, we classified data samples as coming from one of four machines as a proof-of-concept. Additionally, the mixture of Gaussians method was applied to the machine for which data points from a malfunctioning state were available.

A. Methods

Three different learning algorithms were applied to both time domain and frequency domain feature sets. Cross validation used to compare performance of the different algorithms with 70% of the filtered data used for training and 30% of the data held out for testing. Unfortunately, there were only a few measurement points that had the malfunction label, so the size of this training set is less than the number of features.

1) Logistic Regression: One method used to classify data from the four different machines was multinomial logistic regression, implemented using MATLAB's mnrfit and mnrval functions. For the time domain feature set, all features were used, but the frequency domain set, although restricted only Z velocity, needed to be reduced to allow the algorithm to run in a reasonable amount of time. The reduced number of features has the added benefit of reducing the risk of overfitting. Logistic regression was run on the frequency data using 10 features, 20 features, etc., and the resulting test error rates were compared.

2) Naive Bayes: The multinomial Naive Bayes algorithm was used on the frequency spectrum data in order to classify data by machine type. The "frequency of occurrence" of each "token" was represented by the power at each frequency (value of the feature). From the time domain side, we used a feature set of 24 calculated time domain values for each measurement sample. By calculating the posterior with Bayes' theorem, we can classify machine state by selecting the state with the highest likelihood. This method makes a critical assumption that the data are independent and identically distributed, which is not an accurate assumption in this case.

TABLE III
TEST ERROR RATES FOR THREE LEARNING ALGORITHMS APPLIED TO TIME DOMAIN AND FREQUENCY DOMAIN FEATURE SETS

Learning Algorithm	Time Domain	Frequency Domain
Logistic Regression	0.0%	0.0%*
Naive Bayes	25%	0.37%
SVM	0.19%	0.06%

* 50 frequency domain features used for logistic regression

3) SVM: We classified our data into the four machine categories using the SVM algorithm through the LIBSVM library. We assessed test error rates using a linear kernel with default parameters, followed by higher-degree polynomial kernels and modifications to the cost value and LIBSVM's polynomial kernel coefficient gamma.

B. Results

1) Logistic Regression: When 10 frequency domain features were used, the error rate for logistic regression was 13% but dropped to 0.06% using 40 features and 0% for 50 to 100 features. This result is shown in Figure 6. Using time domain features, logistic regression was again able to correctly classify the test set with no errors.

Using the set of same 50 frequency domain features on pre-filtered data, the error rate was 44%. Using time domain features on pre-filtered data gave an error rate of 7.2%.

2) Naive Bayes: Classifying the test data into four machine categories through the Naive Bayes algorithm with 1667 frequency domain features resulted in an error rate of 0.37%. This result is shown in Figure 7. Using the 24 time domain features, the error rate was much higher, at 25%.

3) SVM: When using the SVM algorithm with the frequency domain features on the full dataset (good, bad, and malfunctioning), the error rate was initially 10.0%. Through optimization of the kernel function and parameters, this was reduced to 2.6%.

When using only good and malfunctioning data (throwing out the bad data), the SVM with a linear kernel and default parameters was able to achieve an error rate of just 0.06%. This result is shown in Figure 8.

4) Mixture of Gaussians: The mixture of Gaussians method was able to identify clusters of nominal and malfunctioning data for the SN41 machine using only two time domain features. Figure 9 shows the resulting Gaussians.

VI. CONCLUSION AND FUTURE WORK

Machine learning can be successfully applied to classification based on accelerometer data using either frequency domain or time domain features by applying a three-step process: 1) sort good data from bad data; 2) categorize data into one of several expected states; 3) identify data that represents malfunction for that state. Good data must be sorted from bad data, as sensor measurements are sometimes noisy and prone to error. A model for good vs bad data can be generated using unsupervised learning methods such as k-means initially, and then that model may be applied to the test data to determine whether the data may be further processed for classification.

Fig. 6. Multinomial classification results for logistic regression on time data

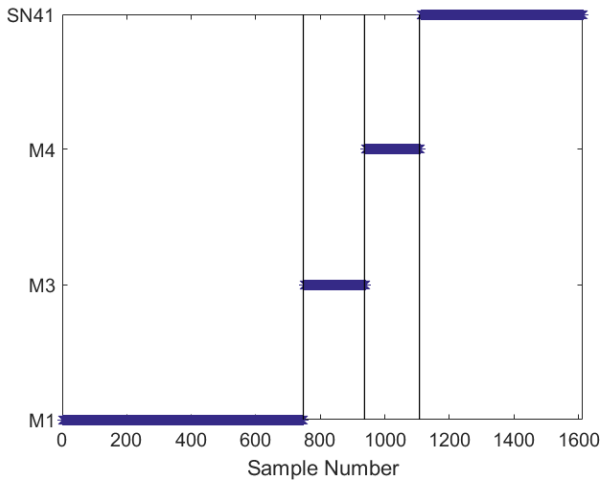
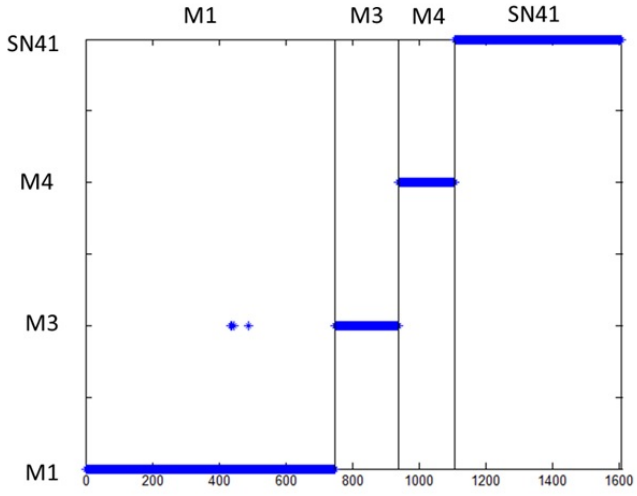


Fig. 7. Multinomial classification results using Naive Bayes on frequency data



For machine type classification (as a stand-in for machine state based on limited available data as proof-of-concept), logistic regression with a subset of available features is extremely accurate on both time and frequency domain data. SVM also provides high accuracy on both feature sets. Naive Bayes works well on the frequency domain feature set, but performs poorly on time domain data. For our purposes, we recommend using logistic regression.

Once the machine type (state) is known, unsupervised learning methods such as mixture of Gaussians may be applied to find data that represents a malfunction.

A drawback of this approach is that a new test sample that is matched against these pre-defined Gaussians may be

Fig. 8. Multinomial classification results using SVM on frequency data

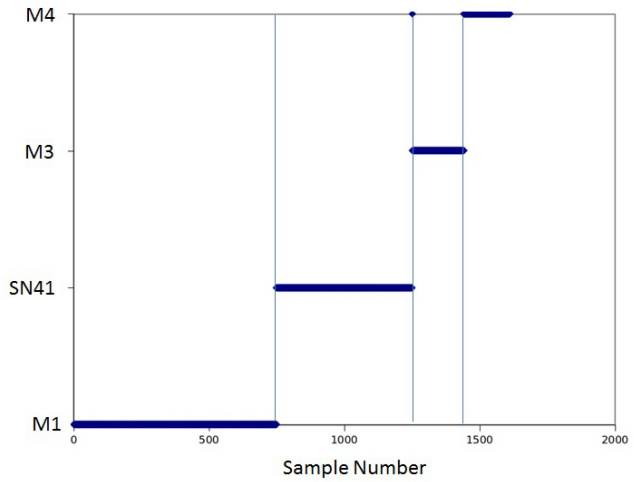
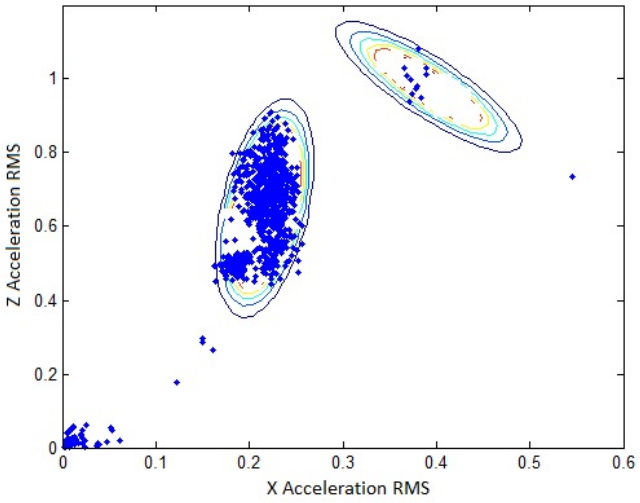


Fig. 9. SN41 data points plotted for two time domain features with PDF contours overlaid and surrounding nominal and malfunctioning data points



classified as "good" instead of "malfunctioning" because the characteristics of this new type of malfunction may be far from those of the malfunction in the dataset from which the Gaussians were computed. An approach to minimize this type of error would be to create a single Gaussian over only the good data. A threshold could be set (for example, two or three standard deviations from the mean) such that if a new sample is below that threshold, it warrants further analyst attention as a potential malfunction.

Because machines can fail in a finite number of ways, it is theoretically possible to collect a dataset that includes samples in all malfunction states. Using this full dataset, supervised learning may be employed to create a definitive model against which to classify future samples. This is a recommended next step to a party with access to such a dataset.

ACKNOWLEDGMENT

The authors would like to thank Petasense, for providing the data that was used to develop this prototype process.

REFERENCES

[1] A. Widodo and B.-S. Yang, "Support vector machine in machine condition monitoring and fault diagnosis," *Mechanical Systems and Signal Processing*, vol. 21, no. 6, pp. 2560–2574, 2007.

[2] A. K. Nandi, C. Liu, and M. L. D. Wong, "Intelligent Vibration Signal Processing for Condition Monitoring," pp. 1–15.

[3] G. A. Hassaan, "Frequency Spectrum Filtering For Machinery Fault Diagnostics," vol. 3, no. 8, pp. 200–203, 2014.

[4] A. K. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mechanical Systems and Signal Processing*, vol. 20, no. 7, pp. 1483–1510, 2006. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0888327005001512

[5] H.-E. Kim, A. C. Tan, J. Mathew, and B.-K. Choi, "Bearing fault prognosis based on health state probability estimation," *Expert Systems with Applications*, vol. 39, no. 5, pp. 5200–5213, 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0957417411015491