# A Study of Machine Learning Techniques for Detecting and Classifying Structural Damage

William Nick, Kassahun Asamene, Gina Bullock, Albert Esterline, and Mannur Sundaresan

***Abstract*—We report on work that is part of the development of an agent-based structural health monitoring system. The data used are acoustic emission signals, and we classify these signals according to source mechanisms, those associated with crack growth being particularly significant. The agents are proxies for communication- and computation-intensive techniques and respond to the situation at hand by determining an appropriate constellation of techniques. It is critical that the system have a repertoire of classifiers with different characteristics so that a combination appropriate for the situation at hand can generally be found. We use unsupervised learning for identifying the existence and location of damage but supervised learning for identifying the type and severity of damage. The supervised learning techniques investigated are support vector machines (SVM), naive Bayes classifiers, and feed-forward neural networks (FNN). The unsupervised learning techniques investigated are k-means (with *k* equal to 3, 4, 5, and 6) and self-organizing maps (SOM, with 3, 4, 5, and 6 output neurons). For each technique except SOM, we tested versions with and without principal component analysis (PCA) to reduce the dimensionality of the data. We found significant differences in the characteristics of these machine learning techniques, with trade-offs between accuracy and fast classification runtime that can be exploited by the agents in finding appropriate combinations of classification techniques. The approach followed here can be generalized for exploring the characteristics of machine-learning techniques for monitoring various kinds of structures.**

***Index Terms*—Machine learning, multiagent systems, structural health monitoring.**

## I. Introduction

Threats to the integrity of a structure, such as corrosion and cracking, produce challenges for the safety and operational capability of the structure as well as costs involved in monitoring and maintaining it. Structural health monitoring (SHM) provides real-time data and consequently information on the condition of the monitored structure. The research reported here has been carried out as part of the NASA Center for Aviation Safety (CAS) at North Carolina A&T State University. The structures of interest are aircraft although experiments at the stage reported here are performed on laboratory specimens. SHM is particularly important for aircraft as structural failure can result in massive loss of life. In our approach, agents typically serve as proxies for techniques with intensive communication or computation requirements. Agents negotiate to determine a team of techniques for solving the task at hand, and they communicate a workflow to a workflow engine, which actually carries out the tasks on the data streams provided. The agents provide flexibility and intelligence so that combinations of techniques suitable for the situation at hand may be determined. It is thus important that a variety of classifiers with different characteristics be available. For example, some will be fast but not very accurate while others will be slow but very accurate. The data we use are acoustic signals, and the condition we address is crack growth. As the source of the signals is unobservable, classifying acoustic signals by their source must be based on machine learning. Note that sensing here is passive: there is no energy input required to generate or sense the signals (although energy is required to store and communicate the data).

In SHM, data is interpreted using parameters that are trained with machine-learning techniques. For our experiments, a correlation coefficient is computed between an observed waveform and six reference waveforms that are generated from numerical simulations of acoustic emission events. The vector of all six correlation coefficients characterizes the waveform. Our dataset consists of a different set of 60 samples from the work reported by Esterline and his colleagues [1].

Worden and his colleagues [2] have formulated seven axioms for SHM that capture general aspects that have emerged in several decades of experience. Of particular interest is their Axiom III, which states that unsupervised learning can be used for identifying the existence and location of damage but identifying the type and severity of damage can only be done with supervised learning. Supervised learning tries to generalize responses based on a training set with the correct responses indicated. Unsupervised learning tries to categorize the inputs based on their similarities. Note that unsupervised learning does not assume that we have already identified categories and, in fact, comes up with categories for classifying data points.

Following Axiom III, our research uses two unsupervised and three supervised learning techniques for different aspects of the SHM problem. The results of machine learning provide a more sophisticated level that will allow us to look at the problem of damage identification. We may then address a multitude of issues and provide diagnoses of the problems. The unsupervised learning techniques are k-means and self-organizing maps (SOM). Supervised learning techniques are support vector machines (SVM), naive Bayes classifiers, and feed-forward neural networks (FNN). For each technique except SOM, we tested a version with principal component analysis (PCA) as a frontend to reduce the dimensionality of the data (usually to three principal components), and we

Manuscript received November 20, 2014; revised February 26, 2015. This work was supported in part by the NASA grant NNX09AV08A and Army Research Office grant 60562-RT-REP.

The authors are with the Department of Computer Science, North Carolina A&T State University, Greensboro, NC 27411, USA (e-mail: wmnick@aggies.ncat.edu, kass842@yahoo.com, ginabull@hotmail.com, esterlin@ncat.edu, mannur@ncat.edu).

DOI: 10.7763/IJMLC.2015.V5.526

tested another version without PCA. The objective is to explore these techniques and note their characteristics so that various combinations of them may be used appropriately in various circumstances.

The approach followed here can be generalized for exploring the characteristics of machine-learning techniques for monitoring various kinds of structures. One must first determine what signals are appropriate for monitoring the structures, (For example, acoustic signals are appropriate for monitoring metallic structures while signals propagated through optical fiber are appropriate for bridge type structures) One then determines the sensor and communication infrastructure. Finally, as per this paper, one determines the characteristics of various supervised and unsupervised learning techniques for monitoring the structures in question (given the signals and infrastructure chosen). Admittedly, the repertoire of techniques explored here is far from complete, but we have included the ones most often encountered in structural health monitoring.

The remainder of this paper is organized as follows. The next section looks into previous work in machine learning for SHM, and Section III provides an introduction to SHM. Section IV presents our approach, Section 5 presents our results, and the last section concludes.

## II. Previous Work in Machine Learning for SHM

Most previous work on machine learning for SHM has targeted bridges; we consider mature, representative work in this area and then turn to research that has targeted aircraft, which is our domain. Figueiredo and his colleagues performed an experiment on a three-story frame aluminum structure that used a load cell and four accelerometers [3]. For each test of state conditions, the features were estimated by using a least squares technique applied to time-series from all four accelerometers and stored into feature vectors. They used four machine learning techniques in an unsupervised learning mode: 1) auto-associative neural network (AANN), 2) factor analysis (FA), 3) singular value decomposition (SVD), and 4) Mahalanobis squared distance (MSD). First the features from all undamaged states were taken into account. Then those feature vectors were split into training and testing sets. In this case, a feed-forward neural network was used to build-up the AANN-based algorithm to perform mapping and de-mapping. The network had ten nodes in each of the mapping and de-mapping layers and two nodes in the bottleneck layer. The network was trained using back-propagation. The AANN- and MSD- based algorithms performed better at detecting damage. The SVD- and FA-based algorithms performed better at avoiding false indications of damage.

Tibaduiza and his colleagues [4], in investigating SHM for an aircraft fuselage and a carbon fiber reinforced plastic (CFRP) composite plate, made use of multiway principal component analysis (MPCA), discrete wavelet transform (DWT), squared prediction error (SPE) measures and a self-organizing map (SOM) for the classification and detection of damage. Each PCA was created using 66 percent of the whole data set from the undamaged structure. Signals from the remaining 34 percent of this data set plus 80 percent of the data set of the damaged structure were used in classifying with the SOM. This approach had an area under the ROC curve of 0.9988. A ROC chart is a display of the performance of a binary classifier, with true positive rate vs. false positive rate.

Esterline and his colleagues [1] (also targeting aircraft) ran an experiment with two approaches. Their first approach used as training instances experimental data with eighteen traditional acoustic emission features to train a SVM, while their second approach used six correlation coefficients between basic modes and waveforms from simulation data also to train a SVM. The SVM with the second approach performed as well or better than the SVM using the first approach, suggesting the superiority of a set of correlation coefficients over a substantial set of traditional acoustic emission features for learning to identify the source of acoustic emissions.

## III. Structural Health Monitoring

In general, damage is defined as change introduced into a system that will adversely affect its current or future performance [5]. This idea of damage is meaningless without a comparison between two states of the system, one assumed to be the unloaded and undamaged state. For mechanical structures, damage can be defined more narrowly as change to the material and/or geometric properties. SHM provides real-time information on the integrity of the structure. It allows better use of resources than scheduled maintenance, which may take place when there is no need.

In characterizing the state of damage in a system, we can ask whether there is damage, where in the system it is, what kind of damage it is, and how severe it is. Damage prognosis is the estimation of the remaining useful life of a mechanical structure [6]. Such an estimation may be the output from models that predict behavior.

The field of SHM has matured to the point where several fundamental axioms or general principles have emerged. Worden and his colleagues [2] suggest seven axioms for SHM. The following are those most relevant to this paper.

**Axiom III**: Identifying the existence and location of damage can be done in an unsupervised learning mode, but identifying the type of damage present and the damage severity can generally only be done in a supervised learning mode.

**Axiom IVa**: Sensors cannot measure damage. Feature extraction through signal processing and statistical classification is necessary to convert sensor data into damage information.

**Axiom IVb**: Without intelligent feature extraction, the more sensitive a measurement is to damage, the more sensitive it is to changing operational and environmental conditions.

**Axiom V**: The length- and time-scales associated with damage initiation and evolution dictate the required properties of the SHM sensing system.

Axiom III is particularly relevant here. Supervised learning together with either analytic models or data from the structure can be used to determine damage type and extent. Statistical methods may also be used.

## IV. Approach

Our overall architecture involves a multiagent system where the agents are typically proxies for computation- or communication-intensive techniques. These techniques are executed on one or more high-performance platforms structured as a workflow engine. Wooldridge defined an agent as an autonomous, problem-solving, computational entity that is capable of effectively processing data and functioning singularly or in a community within dynamic and open environments [7]. The agents in our system negotiate to determine what techniques in what combinations will be used in a monitoring task, thus determining a workflow that is executed on the workflow engine. The multiagent system is thus the "brains" and the workflow engine the "brawn" of our SHM system. Much of the intelligence here is finding the appropriate techniques for the situation at hand. In one situation, we might want a given task done quickly with only rough accuracy, while in another situation accuracy may be paramount and speed of only secondary importance. Regarding the results of machine learning for SHM, we would like an assortment of classifiers to provide a range of possibilities for the diversity of situations that arises in SHM.

Machine learning is generally facilitated by reducing the dimensionality of the data. For this, we use PCA [8]. PCA is an algorithm that centers the data by subtracting off the mean then choosing the eigenvector of the data covariance matrix with the largest eigenvalue [8]. It places an axis in that direction, and then incrementally and similarly places the other axes orthogonally to the first in a way maximizing the possible variation. The number of axes is chosen to be fewer than the number of axes (dimensionality) of the original data set. The data is thus reduced in dimensionality while most of the variation is retained.

Recall that the unsupervised learning techniques we investigated are k-means and SOM. The k-means algorithm [8] classifies *n* observations into *k* clusters. The value of *k* is set by the user. The cluster centers are distributed randomly at first, and a data point is assigned to the cluster nearest it in terms of Euclidean distance. Each cluster center is then updated to be the average of the points assigned to it. The data points are reassigned to clusters and the cluster means are recomputed until the distance of the data points from their cluster centers are within some threshold or some maximum number of iterations is reached.

A SOM [8] is a type of neural network used to produce a low-dimensional, discretized representation of the space of the training data. A SOM identifies features across the range of input patterns. Output neurons compete to be activated, and only one is activated at any one time. A SOM needs very little to no preliminary data cleansing [9].

Recall that the supervised learning techniques we investigated are FNN, SVM, and naïve Bayes classifiers. An artificial neural network (ANN) is a computational model based on the structure and functions of a biological neural network [8]. In a FNN, or multilayer perceptron, input vectors are put into input nodes and fed forward in the network. The inputs and first-layer weights will determine whether the hidden nodes will fire. The output of the neurons in the hidden layer and the second-layer weights are used to determine which of the output layer neurons fire. The error between the network output and targets is computed using the sum-of-squares difference. This error is fed backward through the network to update the edge weights in a process known as back propagation.

SVMs rely on preprocessing to represent patterns in the data in a high dimension, usually higher than the original feature space, so that classes that are entangled in the original space are separated by hyper-planes at higher dimension. Training a SVM [9] involves choosing a (usually nonlinear) function that maps the data to a higher-dimensional space. Choices are generally decided by the user's knowledge of the problem domain. SVMs can reduce the need for labeled training instances.

Naïve Bayes classifiers (NBs) form a supervised learning technique that belongs to a family of classifiers based on Bayes' theorem with a strong assumption about the independence of features [9]. Assumptions and the underlying probabilistic model allow us to capture any uncertainty about the model. This is generally done in a principled way by determining the probabilities of the outcomes. NBs were introduced to solve diagnostic and predictive problems. Bayesian classification provides practical learning through the use of algorithms, prior knowledge, and observation of the data in combination. A Gaussian NB assumes that the conditional probabilities follow a Gaussian or normal distribution.

## V. Results

The learning techniques were run on a machine running a Windows 7 64-bit operating system with a 2.4 GHz quad core processor and 16 GB of RAM. Software from scikit-learn [10] was used for PCA, k-means, SVM, and Gaussian NBs. Software from PyBrain [11] was used for the FNN, and software from Weka [12] was used for the SOM. Weka is written in Java while scikit-learn and PyBrain are written in Python. We recorded the time taken by the classifiers produced by each technique to classify the data points in our test set. For the SOM, this involved executing Java code, while for the others Python code was run. We first present the results for the supervised learning techniques, and then we present the results for the unsupervised learning techniques. Our dataset is split into 30 samples in our training set and 30 samples in our test set.

### A. Supervised-Learning Results

To compare supervised learning techniques, we used classification accuracy, the number of samples classified correctly over the number of samples in the dataset. We also compared techniques on how long the classifiers they trained took to classify the 30 data points in our test set.

TABLE I: Accuracy of the SVM without PCA (30 Points, 26 Runs)

| Kernel | RBF | Polynomial | Linear | Sigmoid |
|---|---|---|---|---|
| mean | 0.90 | 0.13 | 0.87 | 0.83 |

TABLE II: Accuracy of the SVM with PCA (30 Points, 26 Runs)

| Kernel | RBF | Polynomial | Linear | Sigmoid |
|---|---|---|---|---|
| mean | 0.77 | 0.13 | 0.80 | 0.77 |

We ran a SVM with four types of kernel functions: linear, radial basis (RBF, with $\gamma = 0.03125$), polynomial and sigmoid. Table I displays the accuracy with which our SVMs classified the 30 data points in our test set. The SVMs were also trained

with a PCA frontend and run on the same data. Table II displays the resulting classification accuracy. The mean for all our results is for 26 runs. The standard deviation in all cases for SVMs was essentially zero.

A Gaussian naïve Bayes classifier was trained and run with and without PCA. Table III shows the resulting classification accuracy. Again, standard deviations are essentially zero.

TABLE III: ACCURACY OF THE GAUSSIAN NAIVE BAYES CLASSIFIER (30 POINTS, 26 RUNS)

| Technique | Gaussian NB without PCA | Gaussian NB with PCA |
|---|---|---|
| mean | 0.73 | 0.60 |

A FNN classifier was trained and run with and without PCA. Table IV shows the resulting classification accuracy, giving means and (relatively small) standard deviations.

TABLE IV: ACCURACY OF THE FNN (30 POINTS, 26 RUNS)

| Technique | FNN without PCS | FNN with PCA |
|---|---|---|
| mean | 0.62 | 0.64 |
| st. dev. | 0.075 | 0.021 |

Of the four types of kernel functions, the SVM with a radial basis kernel function performed the best without PCA but the SVM with a linear kernel function performed the best with PCA. The SVM with a polynomial kernel function performed much worse than any other technique. The Gaussian NB and the FNN did not perform well. All techniques except FNN performed at least as well without PCA as with PCA.

Table V and Table VI show the time in milliseconds for each of the kernel functions of our SVM with and without PCA to classify the 30 data points in our test set. (As before, the mean is over 26 runs.) All techniques classified the 30 points in 0.09 to 0.17 msec. Processing with PCA made classifying 20-40% faster. The only exception was due to the speed-up PCA gave to SVM with a polynomial kernel function, cutting its time in half.

TABLE V: TIME (MSEC.) FOR SVM KERNEL FCTNS W/O PCA TO CLASSIFY 30 POINTS

| Kernel | RBF | Polynomial | Linear | Sigmoid |
|---|---|---|---|---|
| mean | 0.14 | 0.15 | 0.12 | 0.17 |
| St. dev. | 0.019 | 0.020 | 0.014 | 0.025 |

TABLE VI: TIME (MSEC.) FOR SVM KERNEL FCTNS W/ PCA TO CLASSIFY 30 POINTS

| Kernel | RBF | Polynomial | Linear | Sigmoid |
|---|---|---|---|---|
| mean | 0.10 | 0.09 | 0.09 | 0.14 |
| st. dev. | 0.023 | 0.024 | 0.015 | 0.034 |

Table VII shows the timing in milliseconds for the rest of our supervised learning techniques with and without PCA to classify the 30 data points. PCA again speeds up classifying but not as much as with SVMs.

TABLE VII: TIME (MSEC.) FOR THE REST OF OUR SUPERVISED LEARNING TECHNIQUES (WITH AND WITHOUT PCA) TO CLASSIFY 30 POINTS

| Technique | Gaussian NB w/o PCA | Gaussian NB w/ PCA | FNN w/o PCA | FNN w/ PCA |
|---|---|---|---|---|
| mean | 0.24 | 0.20 | 2.75 | 2.69 |
| st. deev, | 0.029 | 0.018 | 0.224 | 0.202 |

### B. Unsupervised-Learning Results

Regarding unsupervised learning techniques, we first ran k-means clustering without PCA with *k* equal to 3, 4, 5 and 6. Then we ran k-means clustering with PCA with *k* again equal to 3, 4, 5 and 6. Table VIII shows the times taken to classify 30 data points without PCA, and Table IX shows the same times with PCA. All times are in the narrow range 0.16-0.18 msec.

TABLE VIII: TIME (MSEC.) FOR KMEANS W/O PCA TO CLASSIFY 30 POINTS (26 RUNS)

| Technique | Kmeans 3 | Kmeans4 | Kmeans5 | Kmeans6 |
|---|---|---|---|---|
| mean | 0.16 | 0.18 | 0.17 | 0.16 |
| st. dev. | 0.03 | 0.07 | 0.03 | 0.02 |

TABLE IX: TIME (MSEC.) FOR KMEANS W/ PCA TO CLASSIFY 30 POINTS (26 RUNS)

| Technique | Kmeans 3 | Kmeans4 | Kmeans5 | Kmeans6 |
|---|---|---|---|---|
| mean | 0.17 | 0.17 | 0.18 | 0.17 |
| st. dev. | 0.07 | 0.03 | 0.03 | 0.02 |

Fig. 1- Fig. 4 show the clusters for *k* = 3-6, respectively, when PCA with three components was used so that points may be plotted in three dimensions. Note that *k* = 4 identifies four visually convincing clusters.

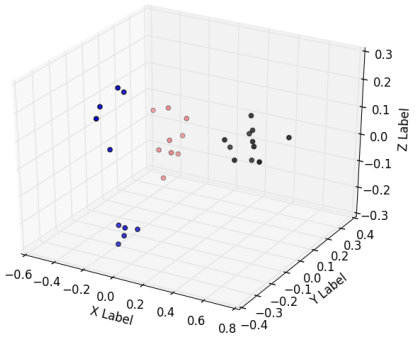


Fig. 1. k-means with 3 clusters with PCA (30 data points).

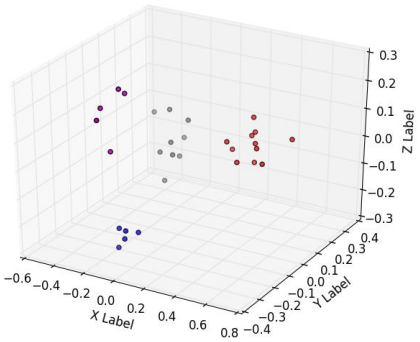


Fig. 2. k-means with 4 clusters with PCA (30 data points).

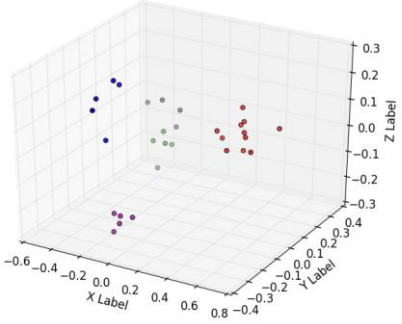


Fig. 3. k-means with 5 clusters with PCA (30 data points).

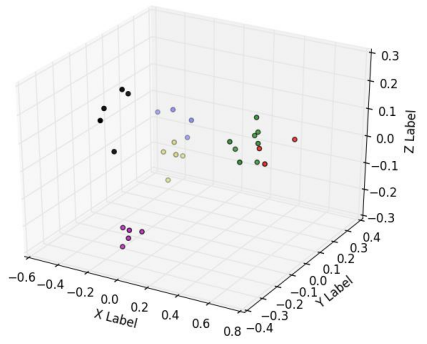


Fig. 4. k-means with 6 clusters with PCA (30 data points).

We ran a SOM with 3, 4, 5 and 6 output neurons. Table X shows the time (in milliseconds) it took each of the SOM instances to classify all 30 data points in our test set. Clearly, the SOMs took much longer than did the k-means classifiers.

TABLE X: TIMING OF THE SOM

| Technique | SOM 3 | SOM4 | SOM5 | SOM6 |
|---|---|---|---|---|
| mean | 409.62 | 445.38 | 617.31 | 600.00 |
| st. dev, | 13.11 | 17.49 | 8.74 | 11.66 |

Table XI shows the sizes of the clusters for the SOMs with 3, 4, 5, and 6 output neurons. Twenty-six runs produced no variation in cluster sizes for any of the SOMs. In the table, the right-side column gives the sizes of the clusters in the order cluster 0, cluster 1, and so on. Clusters of exactly the same size are found with three and four output neurons, but the four case adds an empty "cluster". Indeed, the partition of the set of data points into three clusters gives the least variation in cluster size.

TABLE XI: MEAN CLUSTER SIZES FOR VARIOUS SOMS (26 RUNS), ORDERED AS CLUSTER 0, CLUSTER 1, ETC

| | |
|---|---|
| SOM3: | 14, 7, 9 |
| SOM4: | 14, 7, 0, 9 |
| SOM5: | 7, 2, 5, 5, 11 |
| SOM6: | 5, 11, 5, 0, 2, 7 |

The four sharp clusters (as per k-means with $k = 4$) can be anticipated given our experimental set-up, as we expect signals from four different sources. First of all, there are signals from the crack growth itself, but incremental crack growth deeper in the specimen produces waveforms with rather different characteristics from those produced on the surface, and this difference is enough to pull the crack data into two distinct clusters. Next, there is friction where the specimen is attached, accounting for a third cluster. Finally, the electrical environment produced a consistent kind of noise, giving rise to a fourth cluster. We conjecture that the SOMs tend to find only three clusters because there is a sort of continuum of data points between those related to deep crack growth and those related to crack growth at the surface; a SOM preserves the topological properties of the input space, and these intermediate points apparently pull together otherwise disparate clusters.

## VI. CONCLUSION

This paper reports on work that is part of our development of an agent-based structural health monitoring (SHM) system. The data used are acoustic signals, and one attempts to classify these signals according to source, those associated with crack growth being particularly significant. The agents are for the most part proxies for communication- and computation-intensive techniques. They negotiate to determine a pattern of techniques for understanding the situation at hand. Such a pattern determines a workflow. The agents respond in an intelligent way by determining a constellation of techniques appropriate for the situation at hand. It is critical that the system have a repertoire of classifiers with different characteristics so that a combination appropriate for the situation at hand can generally be found.

Following Worden and his colleagues [2], we use unsupervised learning for identifying the existence and location of damage but supervised learning for identifying the type and severity of damage. Our objective at this stage is to explore various machine-learning techniques and note their characteristics so that various combinations of them may be used appropriately in various circumstances The supervised learning techniques investigated are support vector machines (SVMs), naive Bayes classifiers (NBs), and feed-forward neural networks (FNNs). The unsupervised learning techniques investigated are k-means (with k equal to 3, 4, 5, and 6) and self-organizing maps (SOMs, with 3, 4, 5, and 6 output neurons). For each technique except SOM, we tested a version with principal component analysis (PCA) as a frontend to reduce data dimensionality to three principal components, and we tested another version without PCA.

Turning to the results, first of all for the supervised learning techniques, the most accurate are the SVMs, in the range 77-90% except for the one with a polynomial kernel function (13% with or without PCA), which we henceforth ignore. Using PCA decreases the accuracy of the SVMs by about 10%. The SVM that performed best without PCA was the one with the RBF kernel function (90%), and the one that performed best with PCA was the one with the linear kernel function (80%). The Gaussian NB classifiers did noticeably worse than SVMs, doing worse (60%) with PCA than without (73%). The FNN performed at a level similar to or worse than that of the NB classifiers, with little difference between the case with (60%) and without (62%) PCA. Regarding the time to classify the 30 samples in the test set, the SVMs were the fastest, 20-40% faster with PCA. PCA also speeds up NB and FNN, but not as much as it speeds up the SVMs. NB with PCA takes about twice as long as the SVMs with PCA, and FNN in general requires about six times more time than the SVMs to classify the 30 data points.

Regarding unsupervised learning techniques, the time required for k-means to classify our 30 data points, for all values of $k$ (= 3-6) and with or without PCA, was in the narrow range 0.16-0.18 msec. The SOMs took much longer, around 500 msec. (about 400 msec. with three or four output neurons, and about 600 msec. with five or six neurons).

Note that PCA degrades the accuracy of the supervised techniques except in the case of FNN. This suggests that reducing data dimensions to three as we did with PCA can often obscure information needed for classification. Still, PCA results in faster classification with all the supervised techniques (up to 40% faster) and gives only marginally slower performance for k-means. This is because generally the time required to form the linear combinations of input feature values required by PCA is more than offset by the

speed up in the basic classifier due to reduction in dimensionality.

We can consider combinations of classifiers trained in unsupervised and supervised learning mode, the first to find existence and location of damage and then the second to determine the extent and type of damage. In a practical situation, we look at a large number of events and watch for cases when hundreds are classified as originating from crack growth. So we can tolerate a certain amount of inaccuracy. Cracks, however, grow over months, yet relevant events may be only milliseconds apart, and monitoring a large structure may put a premium on speed. So the extent to which classification time is critical is an involved issue.

These results are generally encouraging for our multiagent system as they reveal significant differences in the machine learning techniques investigated. For both supervised and unsupervised techniques, there are trade-offs between accuracy and fast runtime that can be exploited by agents in finding a combination of techniques appropriate for a given situation. We stated how our approach can be generalized for exploring the characteristics of machine-learning techniques for monitoring various kinds of structures.

Future work will include investigation of the physical reality behind the clusters found with unsupervised learning. We are tagging the waveforms as they are classified so that we may get full information on the waveforms that end up in each cluster. Future work will also consider which classifiers work best in combination with which other classifiers.

Finally, we will investigate approaches to dimensionality reduction. We will investigate what dimensionalities provided by PCA are most efficient in various situations. And we will consider alternatives to PCA. We are looking at coalition game theory [13] as a way to select combinations of feature that do the best job distinguishing the data by class. We are also considering GEFeWS (Genetic and Evolutionary Feature Selection and Weighting), which evolves linear combinations of feature values (over a certain threshold) that optimize classification accuracy [14].

## ACKNOWLEDGMENT

The Authors thank the Army Research Office for funding proposal number 60562-RT-REP and NASA for Grant # NNX09AV08A. Thanks are also due to ISM lab members at North Carolina A&T State University for their assistance.

## REFERENCES

[1] A. Esterline, K. Krishnamurthy, M. Sundaresan, T. Alam, D. Rajendra, and W. Wright, "Classifying acoustic emission data in structural health monitoring using support vector machines," in *Proc. the AIAA Infotech@Aerospace 2010 Conference*, 2010.

[2] K. Worden, C. Farrar, G. Manson, and G. Park, "The fundamental axioms of structural health monitoring," in *Proc. the Royal Society A: Mathematical, Physical and Engineering Science*, vol. 463, no. 2082, pp.1639-1664, 2007.

[3] E. Figueiredo, G. Park, C. Farrar, K. Worden, and J. Figueiras, "Machine learning algorithms for damage detection under operational and environmental variability," *Structural Health Monitoring*, vol. 10, no. 6, pp. 559-572, 2011.

[4] D. Tibaduiza, M. T. Arredondo, L. Mujica, J. Rodellar, and C. Fritzen, "A study of two unsupervised data driven statistical methodologies for detecting and classifying damages in structural health monitoring," *Mechanical Systems and Signal Processing*, vol. 41 no. 1, pp. 467-484, 2013.

[5] C. Farrar and K. Worden, "An introduction to structural health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 303-315, 2007.

[6] C. Farrar and N. Lieven, "Damage prognosis: the future of structural health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sc*iences, vol. 365, no.1851, pp. 623-632, 2007.

[7] M. Wooldridge, *An Introduction to Multiagent Systems*, Wiley, 2009.

[8] C. Bishop, *Pattern Recognition and Machine Learning*, New York: Springer, vol. 1, 2006.

[9] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, Wiley, 1999.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, and É. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

[11] T. Schaul, J. Bayer, D. Wierstra, Y. Sun, M. Felder, F. Sehnke, and J. Schmidhuber, "PyBrain," *The Journal of Machine Learning Research*, vo l. 11, pp. 743-746, 2010.

[12] R. Dimov, "Weka: Practical machine learning tools and techniques with Java implementations," presented at AI tools seminar, University of Saarland, WS 6.07, 2007.

[13] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*, Cambridge, MA: MIT Press, 1994.

[14] A. Alford, K. Popplewell, G. Dozier, K. Bryant, J. Kelly, J. Adams, T. Abegaz, and J. Shelton, "GEFeWS: A hybrid genetic-based feature weighting and selection algorithm for multi-biometric recognition," in *Proc. the 22nd Midwest Artificial Intelligence and Cognitive Science Conference* (*MAICS 2011*), 2011.

**William Nick** received his BS degree in computer science from North Carolina A&T State University in December 2012. From the summer of 2013 until his graduation date, William is funded under the NASA Center for aviation safety and has conducted research related to this field of study. He is a candidate for the MS in computer science.

**Kassahun Mekonnen Asamene** has been a postdoctoral scholar in the Mechanical Engineering Department at North Carolina A&T State University since February 2014. He has BS and MS degrees in mechanical engineering from Addis Ababa University, and he completed his PhD in mechanical engineering in May of 2013 at North Carolina A&T State University.

**Gina La Brooke Bullock** received her BS degree in computer science from Shaw University in May 2002, and MS degree in computer science from North Carolina A&T State University in December of 2003. From the spring semester of 2010 until her graduation date, Gina is funded under the NASA Center for aviation safety and has conducted research related to this field of study. She is a candidate for the PhD in computational science and engineering at North Carolina A&T. She is also on the faculty of the Computer Systems Technology Department at A&T.

**Albert Esterline** is an associate professor of computer science at NC A&T State University. He has graduated over 70 MS students and led the department's effort in establishing a PhD program. His research interests include formal methods, multiagent systems, the semantic web, situation awareness, and network science. Dr. Esterline is the author or co-author of over 100 articles in journals and conference proceedings and four book chapters. Dr. Esterline received a PhD in philosophy from the University of St. Andrews; he also received an MS in mathematics and a PhD in computer science from the University of Minnesota. He frequently leverages his multidisciplinary background in his research.

**Mannur Sundaresan** is currently a professor of mechanical engineering at North Carolina A&T State University and the director of Structural Health Monitoring Laboratory at this institution. His research interests include structural health monitoring, smart structures, damage mechanics in composite materials, nondestructive evaluation, and experimental mechanics. He has been the principal investigator of a number of federal grants and contracts.