Self-Organizing Maps Applied to Engine Health Diagnostics

Thomas Bryant, Michael Hodges
Department of Computer Science and Engineering
Oakland University
Rochester, MI USA
Email: tcbryant {at} oakland.edu

Mohamed Zohdy
Department of Electrical and Computer Engineering
Oakland University
Rochester, MI USA

Abstract— ***This research uses an algorithm implemented by the use of a self-organizing feature map (SOFM) to analyze the similarities and differences between engine sounds to indicate their health status. A neural array feature map was subjected to variation of the parameters of the self-organizing map algorithm, improving map recognition quality. In this specific unsupervised computer learning study, we examined automobile engine sounds in varying degrees of health. With the results in the paper, we have shown how self-organizing feature maps can distinguish between engines of differing ages and health. With the results that were gathered from two tests with three sets each of engine sounds across three different domains, the clusters have shown that distinct differences between the engine sets contribute to the study of sound discrimination in a substantial way.*** **(Abstract)**

***Keywords-** sound, self-organizing map, clusters, features, time, frequency, engine health (key words)*

I. Introduction

Cluster analysis is the assignment of observations into subsets (clusters) so that observations within the same cluster are similar according to self-organizing feature map (SOFM) criteria. Different clustering techniques make different assumptions on the structure of the data, often defined by SOFM similarity metrics and evaluated by criteria such as internal compactness and separation between different clusters. Data clustering aims at identifying clusters as more densely populated regions in the space R^m. This is a traditional problem of unsupervised pattern recognition. The general strategy is as follows: at first, find the optimal partition of the points into K classes, and then change the value of the parameter K from N to 1 [1]. Cluster analysis is an iterative process of knowledge discovery as well as interactive optimization that involves trial and error.

Unsupervised learning neural networks are characterized as competitive networks that compete on an input vector. Self-organizing maps (SOFM) are an example of unsupervised neural networks. The learning process of the self-organizing maps is competitive, meaning no teaching is needed to define the correct output for a given input [2]. The purpose is not to find an optimal clustering of the data, but to get good insight into the cluster structure of the data for data mining purposes. Therefore, the clustering method should be fast, robust, and visually efficient. The clustering is carried out using a two-level approach, where the data set is first clustered using the SOFM, and then the SOFM is clustered [3].

Sounds are multidimensional signals and can have practical applications in device health monitoring, containing complex features such as volume, pitch, frequency, and timbre. Humans can easily discern differences and similarities between sounds, but this paper explores how a computer program would handle this problem. By extracting suitable features from sounds, a self-organizing feature map is able to display (in a visual environment) the discriminant clustering that occurs.

Sound can be represented in three domains, the time domain, the frequency domain, and the time-frequency domain, otherwise known as the joint domain. A time-domain graph shows how a signal changes with time. A graph from the frequency domain shows how much of the signal lies within each given frequency band over a range of frequencies [9]. The joint domain, represented by a spectrogram, studies both the time and frequency domains simultaneously. Unlike individual frequency and time domains, the joint graph also displays color to represent magnitudes. A red color on the graph indicates a high magnitude, while a blue color is a low magnitude. The color helps to add another visual dimension to the data. Key features were extracted from each domain. The key features extracted from the time domain are obtained by using principal component analysis, which is discussed in more detail later. The key features extracted from the frequency domain are resonances, which are obtained by analyzing the peaks and frequencies of the respective graph. Lastly, the features extracted from the joint domain come from the quadtree function and are obtained by splitting the matrix into four quadrants and the most dense quadrant into another four quadrants, and measuring mean and range of spectrogram magnitudes in each quadrant.

II. ENGINE SOUND APPLICATION

The number of SOFM features varies in our tests. The optimal is between 4 and 14; for the time domain, they were between 4 and 8, and for the frequency domain, it varied depending on the number of peaks.

Once the features were obtained, the map was initialized with random weights. Nodes were created to form map anchors with these pseudo-random entities. The algorithm reads the data features from the input file and compares it to the random numbers, determining which pseudo-random number it is closest to using the Euclidean distance formula.

Before the SOFM algorithm begins to cluster the nodes, it also takes in ten parameters: the neighborhood radius, learning rate, rate decay, neighborhood decay, input history consideration limit, and input history factor. The default values are best for different types of data, but the user can specify any or all of the values to obtain more effective clustering. The history function in particular is an important contribution, as it clusters the map based on a factor of recent inputs. The colors of the previous several iterations are displayed underneath the map in our program.

In the last step of the feature extraction process, features are entered. Fourteen text boxes manipulate the data. Once the "input" is given, the top node of a context tree changes to the color of the noise it most closely resembles. This comparison is based solely on the information obtained from the input file, and does not affect the SOFM.

Preprocessed data is used to obtain the salient features. Once the features are obtained, the program learns the patterns hidden in the data and maps the nodes accordingly. To classify the sounds, the winning node is calculated using the Euclidean distance formula to determine which node is closest. The output is visualized and displayed on a two-dimensional map, and the vector is colored according to the sound that has the closest node. It visualizes sounds as similar colors that act as agents that compete against each other. The vectors that weigh the least in difference "win".

III. FEATURE EXTRACTION ALGORITHM

A. *Features from the Time Domain*

To extract the features from the time domain, principal component analysis (pca) was used. In this procedure, the eigenvalues of the covariance matrix are calculated. The values are clustered together in groups of different standard deviations, and a score is calculated as the value of each of these clusters. The largest clusters are used.

B. *Features from the Frequency Domain*

To obtain features from the frequency domain, the data was plotted. Then, the peaks and frequencies are analyzed and resonances and their magnitudes are determined from this. W_i, or the frequency of the signal, is obtained from the x-axis and M_i, or magnitude (or peak), is obtained from the y-axis. Specifically, the peaks are determined and one resonance is obtained from each one.

C. *Features from the Time-Frequency Domain*

In order to obtain the features from the time-frequency domain, quad division was used to split the data. The densities of each quadrant were calculated to determine the most active quadrant, based on the proportions of values which are greater than the average of all the values in the matrix. The most active quadrant (the one with the highest density) was then further divided into four sub-quadrants using the same equations shown above, and again was ordered. Using the seven quadrants, the arithmetic means and the ranges were calculated.

IV. LEARNING ALGORITHM

A. *Training*

The primary goal of self-organizing feature maps is to transform patterns of arbitrary dimensionality into the responses of one or two-dimensional arrays of neurons. These three steps are required [10]:

1) an array of neurons that compute low dimension output functions of incoming inputs of arbitrary dimensionality;

2) a mechanism for selecting the neuron with the closest matching output;

3) an adaptive mechanism that updates the weights of the selected neuron and its neighbors.

The algorithm used to train self-organizing maps is based on the training algorithm by Kohonen and summarized as follows:

Initially, the weight vectors of the map's nodes are randomized. It takes an input vector and traverses each node in the map calculating the Euclidean distance:

$$d(p,q)=d(q,p) \tag{1}$$

$$= \sqrt{(q_1-p_1)^2+(q_2-p_2)^2+\cdots+(q_n-p_n)^2} \tag{2}$$

The similarities between the input vector and node weight vector correspond to distances between them. The algorithm then tracks the node that produces the smallest distance (this node is called the best matching unit, or BMU). The neighborhood function has the BMU as center node along with its neighborhood making them even more similar to the input vector by using the equation below:

$$W_v(t+1)= W_v(t)+ \alpha(t)*\beta(t)*[dist(t)-W_v(t)] \tag{3}$$

Increase *t*, which is a positive constant and repeat from 2 while $t< \lambda$:

t denotes current iteration

λ is the limit on time iteration

W_v is the current weight vector of node v

dist(t) is the target input data vector

α(t) is learning restraint due to time *β(t)* is restraint due to distance from BMU, usually called the neighborhood function

The above parameters determine the efficiency of the SOFM algorithm.

B. Clusteringof the Self-Organizing Map

The Euclidean distance method described earlier is used to find the distance of between each set of two-dimensional vectors in such a manner that it arranges the N^2 selected cluster centers into an *NxN* matrix to form a self-organizing feature map of various characteristics.

$$S_{nn} = \frac{\sum_i \min_j \{\|x_i - x_j\|\}}{N_k} \quad (4)$$

This cluster equation is known as the nearest-neighbor equation and is used in conjunction with the Euclidean method. The specific type of clustering is known as agglomerative clustering and requires the following steps:

1) Initialize: Assign each vector to its own cluster.
2) Compute distances between all clusters.
3) Merge the two clusters that are closest to each other.
4) Return to step 2 until there is only one cluster left.

Each loop iteration performs a single search for the nearest neighbor of a cluster, and either adds one cluster to a stack or removes two clusters from it. Every cluster is only ever added once to the stack, because when it is removed again it is immediately made inactive and merged. There are a total of 2n − 2 clusters that ever get added to the stack: n single-point clusters in the initial set, and n − 2 internal nodes other than the root in the binary tree representing the clustering. Therefore, the algorithm performs 2n − 2 pushing iterations and n − 1 popping iterations, each time scanning as many as n − 1 inter-cluster distances to find the nearest neighbor. By using the above steps, it ensures that the clusters of data are compact and well-separated [5].

V. SIMULATION RESULTS

A. Sound and Feature Selection

Several types of sounds were analyzed to test the SOFM. The focus of this project went into engine sounds, distinguishing engines in ideal health from those in poor health. Also considered were engines in good health but old in age. For a large portion of the project, several older engine samples and several newer ones were used. The sound file is saved to a variable called wave, and the sound format of choice was the wav format. The file is then displayed using the fast-Fourier transform function and then using the spectrogram function. The level of visualization of the SOFM was determined using the following parameters: sequential and random input, a radius, a learning rate, a rate decay, a neighborhood decay, a history limit, and a history factor. The neighborhood value has a two-fold character - a size and a function of distance to influence [12].

First, different engine sounds were gathered [17-18]. The sounds were first preprocessed in MATLAB by plotting the sound data into a joint time-frequency domain. The magnitude of the matrix was calculated using the abs() function, which created a matrix. This matrix could be converted into a text file, which was then read by the Java application. The extracted matrix is analyzed by the feature extraction algorithm. The output is a feature vector whose content can range from 4 to 14 features, which is the optimal range. Several features were selected from time and frequency domains as vectors and joint time-frequency domain is selected as a matrix [4].

Figures 1 and 2 show graphs of the time domain for one of the oldest engine sounds and one of the newest ones used in this experiment, respectively. In the graph, length serves as the x-axis and amplitude as the y-axis. The cycling nature of the waveform indicates the repetitiveness of the engine sounds.

Figures 3 and 4 display the frequency domains for a good engine and a bad engine that have had key features extracted from them. Frequency represents the x-axis and magnitude represents the y-axis. The fast Fourier transform (fft) starts and ends at exactly the same spot, indicating the repetitive nature of the sound.

Figure 5 and Figure 6 represent spectrograms as the joint time-frequency domain. It represents quad energy, and the energy level is higher where the time variable is lower and the frequency variable is higher. All of the above graphs were first produced in YMEC [16], and later in MATLAB.

The nodes in the SOFM are initialized with small random weights. This is an important step, since the initial weight determines the learning speed of the map. The larger the learning rate, more iterations it will take. However, if the numbers are too small, then the user will be able to see on the SOFM that the sounds did not cluster very well, since the colors will not group together seamlessly on the visual display. The next step involved clustering and classifying the training data using the learning algorithm. The 17x17 neural network was trained with data that came from sound variations of two engines in good health, two older engines that were in moderately good health, and two older engines in poor health. These engines were chosen because they represented data that clustered ideally. Engine training data was fed to the neural network repeatedly so that enough iterations would cause the feature map to cluster and separate the feature vectors in 2D space.

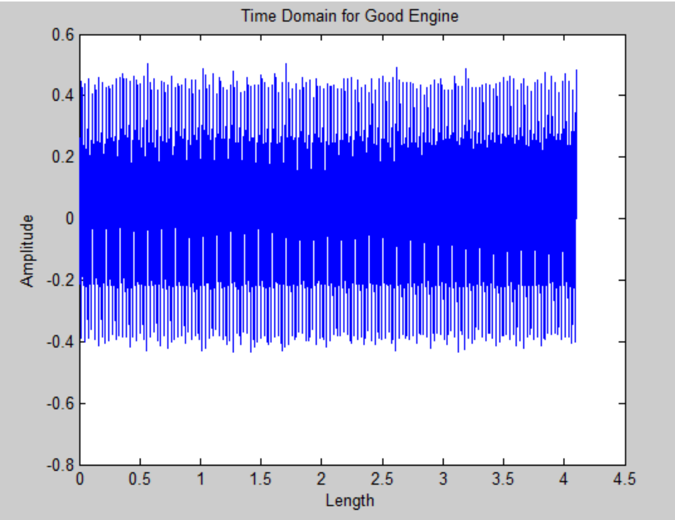

Figure 1. Good engine in the time domain.

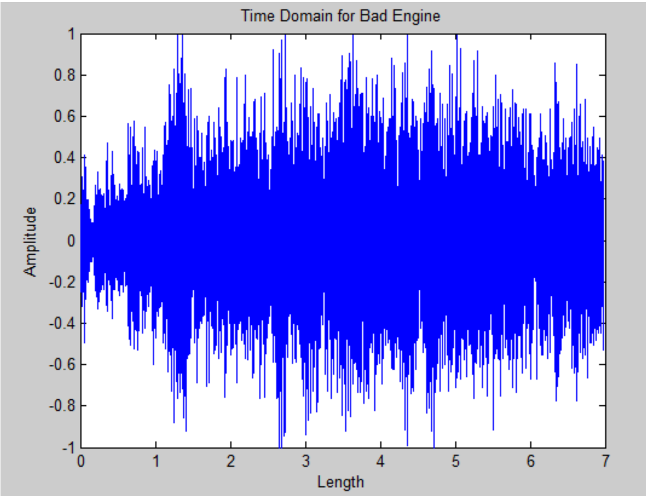

Figure 2. Bad engine in the time domain

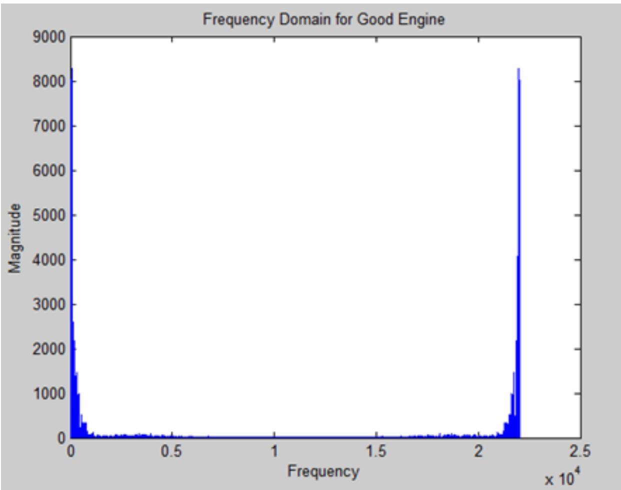

Figure 3. Good engine in the frequency domain

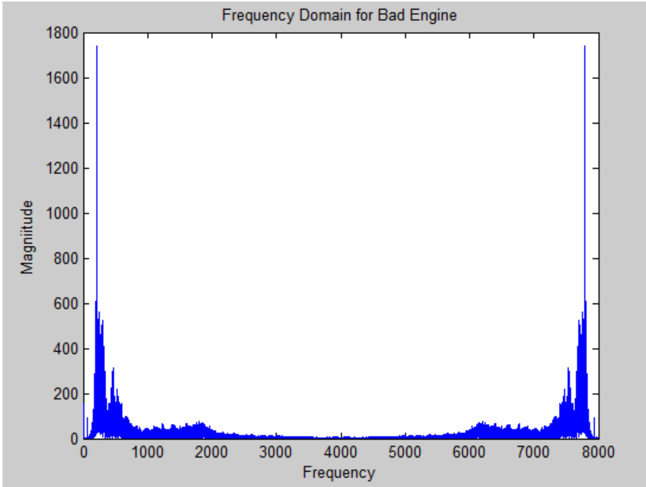

Figure 4. Bad engine in the frequency domain.

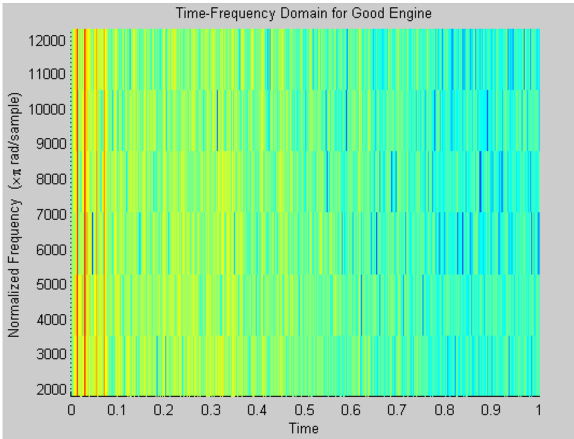

Figure 5. Good engine in the joint domain.

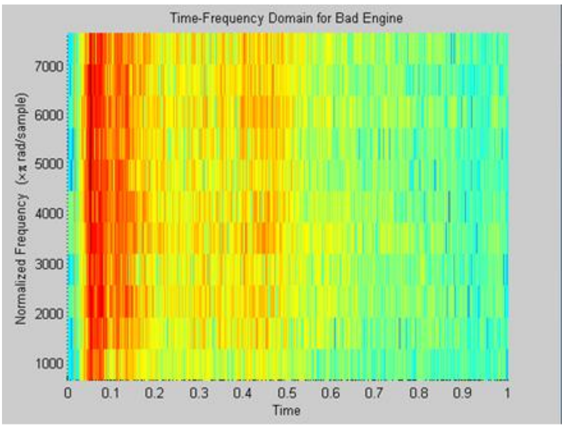

Figure 6: Bad engine in the joint domain

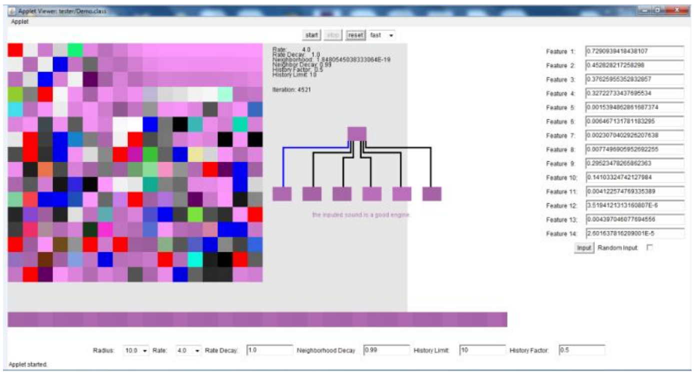

Figure 7. The self-organizing map being trained to recognize clusters.

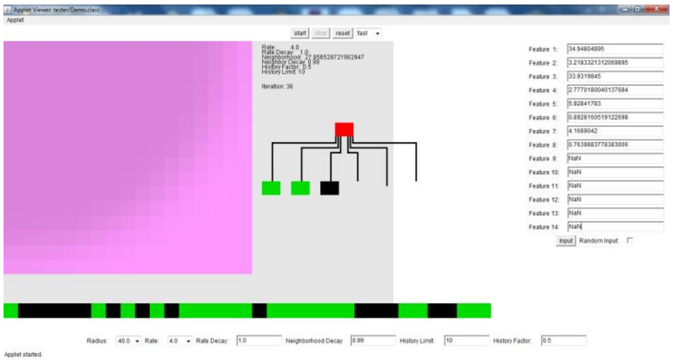

Figure 8. The self-organizing map in time domain.

The data was extracted into a feature vector of input length fourteen and shown on a context tree. After testing of the parameters, they were fine-tuned using the following parameters: sequential input, a radius of 10.0, a learning rate of 4.0, a rate decay of 1.0, a neighborhood decay of 0.99, a history limit of 10.0, and a history factor of 0.5. In this step, the clustering of the sound could be affected by the input sequence, since this SOFM uses a history function. This history function updates previous winning nodes in order to decrease the time necessary to complete the training phase; the history limit shows the previous winning nodes below the SOFM, and the history factor determines the accuracy. Using a random input sequence makes it harder to track the pattern, in comparison to a sequential input sequence. After training, different random engines were used for testing the map. Random input means that the files were selected randomly to be put into the SOFM. The sequential input used the data in the files in the order that they were written.

B. Cauchy Distribution

In order to obtain similar but slightly different engine sounds from all three categories, they were altered slightly using the process of perturbation. To form many patterns, each matrix was perturbed many times. More specifically, the Cauchy distribution function, an offset of the Gaussian, was used to determine the patterns. The fact that the Cauchy distribution is not merely one distribution, but many random distributions rolled into one, made it a good model for the perturbation purposes [15]. The probability density, or fat-tailed density, of the Cauchy distribution deviates from the mean by more than five standard deviations, but it allows the data to arise naturally. The mode is a, and the parameter restriction is that b must be greater than 0:

$$f(x) \qquad f(x)=\left\{\pi b\left[1+\left(\frac{x-a}{b}\right)^{2}\right]\right\}^{-1} \tag{5}$$

Even if the feature map does not separate and integrate the node colors perfectly, however, it is still useful to test new sounds using a measure Euclidean distance.

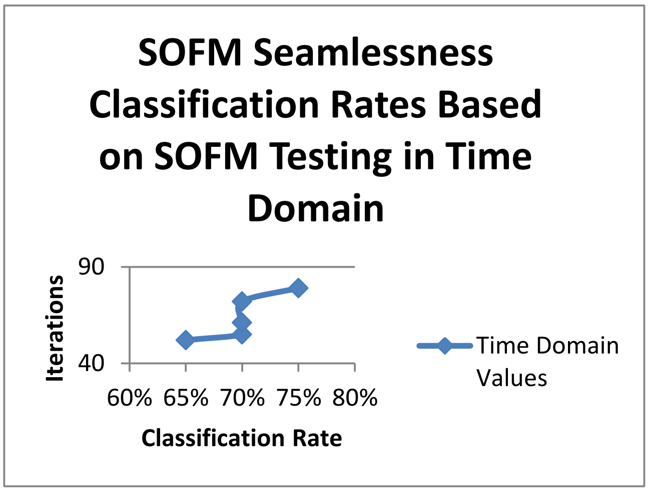

Figure 10. Time domain results.

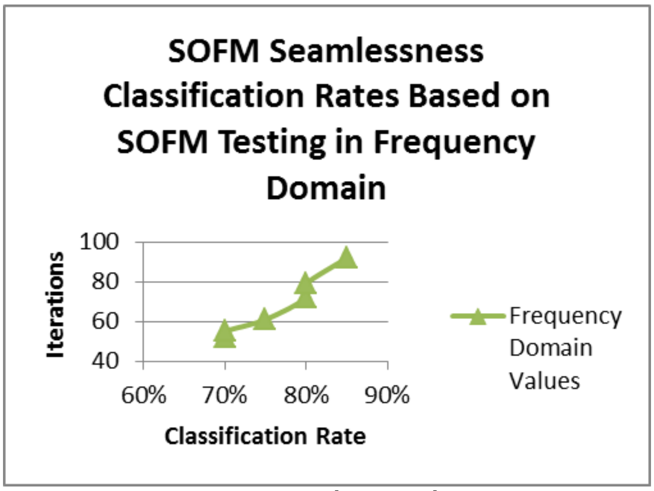

Figure 11. Frequency domain results.

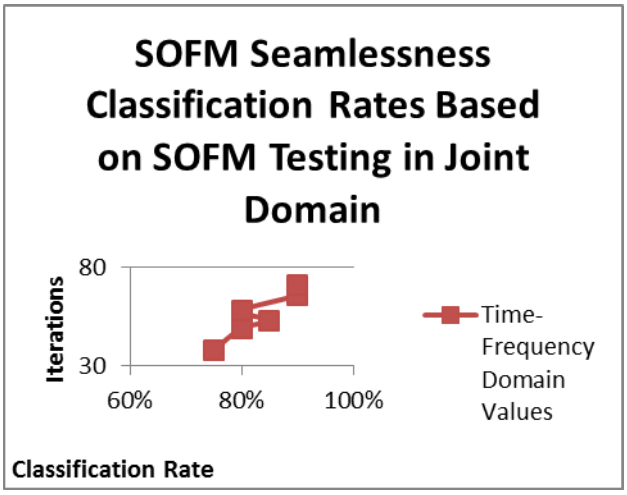

Figure 12. Joint domain results

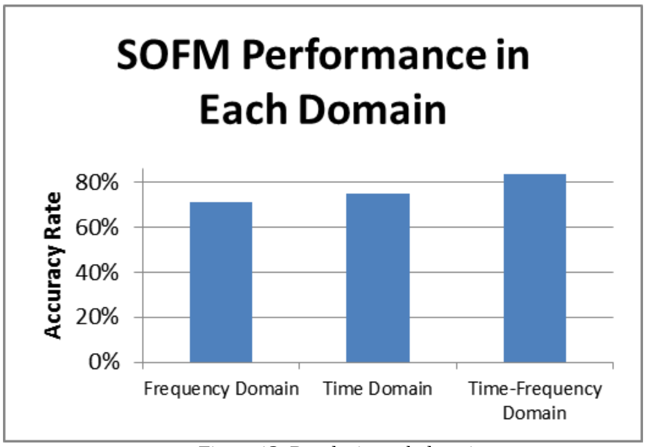

Figure 13. Results in each domain

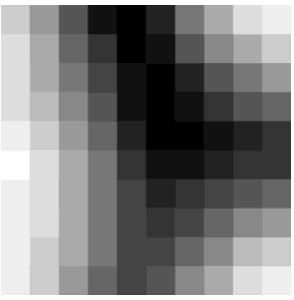

Figure 14. With history

Figure 15. Without history

C. *Other Perturbed Engine Sounds*

Regardless of the convergence of nodes, our next step was to force the feature map to categorize a sound as one of the three types of engines. This map could then be clustered with a higher degree of seamlessness.

We perturbed the engine sounds with a .4 Hz two-stage phaser, a four octave pitch increase, and a 50% tempo increase in three trials for each of three engine types. The results were 67% correct classification in the time domain, 78% correct in the frequency domain, and 100% in the joint domain. In this test we forced classification to one of three outputs represented in the SOFM based on Euclidean distance. This shows the range of feature qualities among the different domains.

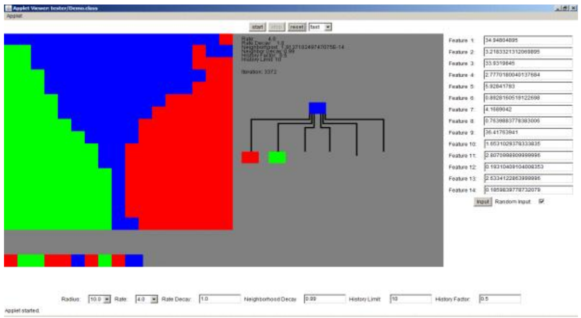

Figure 16. Classification forced to one of three outputs

VI. CONCLUSION

The SOFM (self-organizing feature map) classified sequential engine sound inputs according to selected features and accurately clustered the data. Moreover, it provided a context for input features that served as training data with a context tree diagram and displayed the color that most resembled the sound. After testing, the SOFM performed best with sequential inputs, a neighborhood radius of 10.0, a learning rate of 4.0, a rate decay of 1.0, a neighborhood decay of 0.99, a history limit of 10.0, and a history factor of 0.5. This was determined by how well the colors clustered together and how many randomly colored squares were left on the SOFM. Clustering based on feature map Euclidean distance was still quite successful regardless of the clustering, and produced up to 100% accuracy in the joint domain.

In conclusion, SOFMs were successful in categorizing and identifying engines in good health from those in medium and bad health. The history function that serves as one of the algorithm parameters improves the performance during the training phase substantially. Correlation increases the success rate of identifying unknown sounds by 15-20%. Blended distance not only has a significant effect in improvement of computational time in our self-organizing feature maps, but it can also be used in a number of applications such as recognizing speech patterns, genetics, and chemical sensors.

ACKNOWLEDGMENT

The authors would like to recognize George Corser for his preparation advice and his efforts in helping with the necessary Java code. Also, thank you to Matthew Bradley, Kay Jantharasorn, and Keith Jones, who created this program that we modified for the course of this project. Moreover, we appreciate the funding provided by the NSF that allowed us to complete the research. This research work was conducted at Oakland University and funded by the NSF under grant number 22029.

REFERENCES

[1] Y. Stekh, FME Sardieh, and M. Lobur. “Neural network based clustering algorithm.” MEMSTECH’2009. 2009:1-2.

[2] J. Mohammadpour, M. Franchek, and K. Grigoriadis. “A survey on diagnostic methods for automotive engines.” International Journal of Engine Research, 2011, vol. 13, no. 1, 41-64.

[3] J. Vesanto, and Alhoniemi, E. “Clustering of the Self-Organizing Map.” IEEE Transactions on Neural Networks, vol. 11, No. 3, May 2000. Neural Networks Res. Centre, Helsinki Univ. of Technol., Espoo, Finland.

[4] Mgaya, Richard Hans, Charles D. McCurry, and Saleh Zein-Sabatto. “Intelligent Methodology for Turbine Engine Diagnosis and Health Management.” Southeastcon, 2009. SOUTHEASTCON '09. IEEE. doi: 10.1109/SECON.2009.5174115.

[5] Bradley, Matthew, Kay Jantharasorn, Keith Jones, and Dr. M. Zohdy. “Self Organized Neural Networks Applied to Animal Communications.” REU Report, 2008.

[6] Vesanto, Juha and Esa Alhoniemi. "Clustering of the Self-Organizing Map.” IEEE Transactions on Neural Networks, vol. 11, no. 3, May 2000, pp. 586-600.

[7] Wernick, Yang, Brankov, Yourganov, and Strother, “Machine Learning in Medical Imaging”, IEEE Signal Processing Magazine, vol. 27, no. 4, July 2010, pp. 25-38.

[8] Nsour, Ahmad R., and Mohamed A. Zohdy. "Self Organized Learning Applied to Global Positioning Systems (GPS Data)." Print.

[9] Su, Mu-Chun and Hsiao-Te Chung. “Fast Self-Organizing Feature Map Algorithm.” IEEE Transactions on Neural Networks. Vol. 11, no. 3, May 2000, pp. 721-733.

[10] Erwin, E., Obermyer, K., and Schulten, K. “Self-Organizing Maps: Ordering, Convergence, and Energy Functions.” University of Illinois at Urbana-Champagn.

[11] Pang, Kevin. “Self-organizing Maps.”, unpublished.

[12] Hank-Cluster Visualization. Sinica University. http://gap.stat.sinica.edu.tw/Talks/Hank-ClusterVisualization.pdf

[13] Blei, Dave. Principal Component Analysis (PCA). http://www.cs.princeton.edu/courses/archive/spr08/cos424/scribe_notes/0424.pdf

[14] Vlachos, Michalis and Spiros Papadimitriou, IBM T.J. Watson Research Center.

[15] Diagnostic System for Sound Fields. 2013. Yoshimasa Electronic Inc.

[16] FindSounds. 2013. Comparisonics. 12 June, 2013. http://www.findsounds.com/

[17] AutoSpeak. 2013. Library of Sounds. http://www.autospeak.com/library.htm