FEATURE EXTRACTION NETWORKS FOR DULL TOOL MONITORING

Lane Owsley*, Les Atlas*, and Gary Bernard**

*Interactive Systems Design Laboratory, Department of Electrical Engineering, FT-10
University of Washington, Seattle, WA 98195, U.S.A.

**Boeing Commercial Airplane Group, P.O. Box 3707, M/S 5K-14,
Seattle, WA 98124-2207, U.S.A.

ABSTRACT

Automatic feature extraction is a need in many current applications, including the monitoring of industrial tools. Currently available approaches suffer from a number of shortcomings. The Kohonen self-organizing neural network (SONN) has the potential to act as a feature extractor, but we find it benefits from several modifications. The purpose of these modifications is to cause feature variations to be aligned with the SONN indices so that the indices themselves can be used as measures of the features. The modified SONN is applied to the dull tool monitoring problem, and it is shown that the new algorithm extracts and characterizes useful features of the data.

1. MOTIVATION FOR THIS WORK

Monitoring the evolution of machine tool performance is an area in which manufacturing industries are very interested. Great expense is potentially involved either in damaging a part by using a dull tool or in attempting to avoid this damage by replacing tools prematurely. Automatic on-line evaluation is thus a strong desire, but current systems to this end are based on features chosen by human observation of the data. While such systems have had success on some applications [1,2], they are inherently biased toward the use of features which humans can readily perceive and model.

Figure 1 shows short-time Fourier transforms (spectrograms) of vibration patterns from holes drilled using both sharp and dull drills. It is easy to see that, for the dull tool, the main carrier frequency has decreased and that a 9.7 kHz resonance has become significant in amplitude. However, these features do not capture the full complexity of the evolution of the dulling process. The desired classification system would provide more information than is available using these hand-selected features.

2. TRADITIONAL FEATURE EXTRACTION

Ideally, we would like to analyze the drilling vibration patterns by applying an automatic feature extractor to them, *i.e.* finding new variations which are present in the data set instead of being forced to pre-determine which variations we are interested in. However, the bulk of current automatic feature extraction is based on clustering approaches developed not for vector time-sequences but instead for single vectors (e.g. [4]). The ability of clusters to express variation over a continuum is limited, because they introduce artificial boundaries and because they contain no sense of ordering. For example, a clustering method applied to the spectrograms of Figure 1 might map all main resonances below

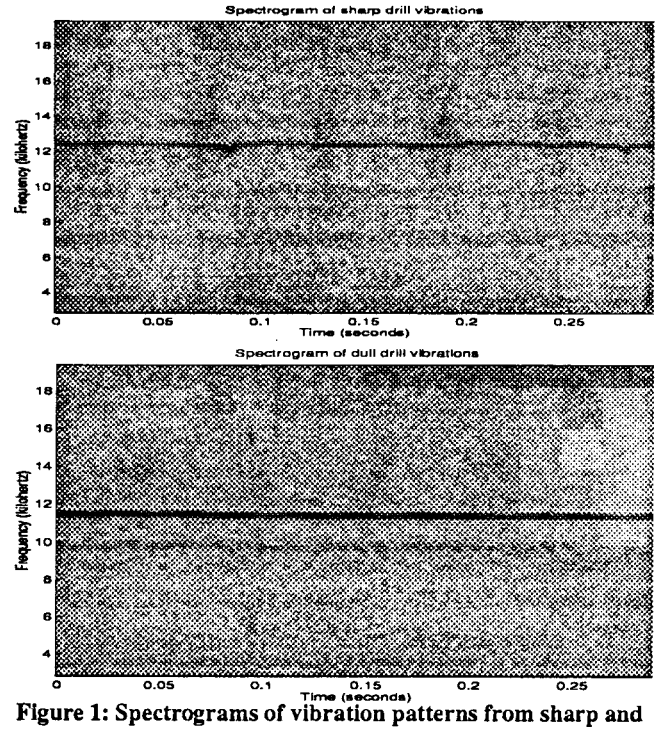

Figure 1: Spectrograms of vibration patterns from sharp and dull tools.

11.5 kHz to cluster A, resonances above 12 kHz to cluster B, and resonances from 11.5-12 kHz to cluster C. Thus a tone which moved smoothly through the frequency spectrum might be represented unnaturally by a series of abrupt jumps between disjoint and unordered clusters.

One commonly used approach to feature extraction which can encode features over a continuum is principal component analysis (PCA)[4]. This approach involves taking a high-dimensional data set and extracting from it the directions of maximum variance. The "features" which are then extracted are the projections of the input data along these principal directions.

A restriction of the PCA approach,though, is the fact that it is looking for features which are linear combinations of the input data vector elements. Figure 2 shows a region of some of the principal components trained on the drill spectrograms of Figure 1. The feature which the components are attempting to express is the frequency of the main carrier resonance, which varies from 11.2 kHz to around 12.5 kHz. However, these spectral vectors are related in a nonlinear fashion to the locations of the main resonant frequency, an as a result PCA is not able to express this feature

0-7803-2431-5/95 $4.00

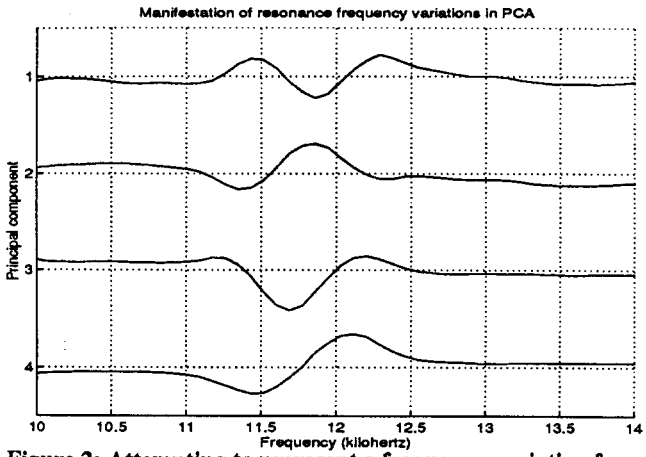

Figure 2: Attempting to represent a frequency-variation feature using linear principal components. Different components are sensitive to different regions of the spectrum, which results in an awkward and inefficient encoding.

succinctly but instead clumps the spectrum up into correlated bins. The ideal feature extractor would be able to express the frequency feature as a single quantity.

3. KOHONEN SELF-ORGANIZING NEURAL NETWORKS

Kohonen has presented a type of feature extractor with the potential to describe feature variation along a nonlinear continuum [5]. His self-organizing neural networks (SONNs) take the form of grids or other low-dimensional structures. The grid vertices are data vectors, and when the SONN is being used in feature extraction an input data vector maps to the most similar vector in the grid (i.e. the SONN grid is thought of as a codebook and the encoding process is the feature extraction process). In training, these grids are elastic in that they respond to the training vectors by deformation. The training vectors successively attract the nearby sections of the map, and as a result the map becomes stretched out across the training data set. SONNs have been shown useful in data analysis [9], because they can provide low-dimensional representations of high-dimensional data, and this makes it easy for a human to visualize data clusters and relations. However, use in automatic feature extraction is not as straightforward. These approaches usually require *a priori* knowledge of which features are of interest [6], so that the data vectors can be pre-labeled with their feature values. The reason for this is that the actual map locations (e.g. the row and column indices in a two-dimensional grid map) do not have a consistent meaning across the map. The training algorithm does not make use of the index structure of the map and as such may not cause variation to be aligned with the grid [7]. As a result, there is no consistent way to extract feature information from the SONN output. Our objective is a feature map in which the index values themselves can be used as measures of the data features [8].

4. MODIFICATIONS TO KOHONEN'S ALGORITHM

In the Kohonen SONN algorithm, the decision on where to position a training vector on the map is based purely on which grid vector is most similar to the training vector. We are instead interested in basing this decision on which row and column are most appropriate for the training vector. That is, we want to think of grid locations as intersections of rows and columns; mapping a data vector to a particular location should mean that it is similar to the other vectors within the same column or row, since it will be addressed with the same column or row index value. We achieve this through three modifications:

1. In selecting the point to place a training vector for map deformation, we base the choice not just on the map vector at that location but also on all the vectors in the corresponding rows and columns. This is achieved by the formation of a characteristic vector for each row or column (*i.e.* the mean vector) which is compared to the incoming training vector, or by comparing the training vector to each member of the row or column.

2. Instead of effecting the deformation in a circular region, we deform the map along the row and column of the center point. This allows all vectors mapped to a particular row or column to affect the characteristics of the entire row or column.

3. We use an expanding network similar to that of Rodrigues and Almeida [10]. This facilitates the use of the first two modifications by giving their effects a larger range in the early training and then narrowing their focus as training progresses.

4.1 The Choice of an Appropriate Location for the Training Vector

As mentioned previously, Kohonen's SONN algorithm selects an appropriate updating region for each training vector by centering the region around the codevector most similar to the training vector, that is the codevector which minimizes the function

$$dist\{s_k, neu(m,n)\} \tag{1}$$

where $dist\{\}$ is the distortion function being used to calculate the relationship between vectors, s_k is the training vector and $neu(m,n)$ is the codebook vector from row m and column n. The modified algorithm chooses the location (m,n) which minimizes the function

$$\alpha\Big(dist\{s_k, neu(m,n)\}\Big) + (1-\alpha)\,(d_{row}(n) + d_{col}(m)) \tag{2}$$

where

$$d_{row}(n) = dist\left\{s_k, \frac{1}{M}\sum_{m=1}^{M}(neu(m,n))\right\} \tag{3}$$

and

$$d_{column}(m) = dist\left\{s_k, \frac{1}{N}\sum_{n=1}^{N}(neu(m,n))\right\} \tag{4}$$

This expression (Equation 2) is a weighted sum of the original distortion measure from Equation 1 with two other measures (Equations 3 and 4), a row distortion measure and a column measure. The row measure is a comparison of the training vector to the mean value of all the codevectors in the row being considered, and the column measure is a comparison of the training vector to the mean value of all the codevectors in the column being considered. The result is that instead of making the choice for the best location of the training vector based on a measure of the single best element of the codebook, we are including a measure of the best row and column as well. The parameter α controls the extent

to which this new distortion measure is incorporated, with an α of 1 resulting in the use of the original distortion measure alone and an α of 0 causing the decision to be made based purely on the new information. As a result of this modification, the algorithm now has an explicit knowledge of the index (row-and-column, in this case) structure, which gives the algorithm an incentive to assign similar vectors to a particular value of a given index, regardless of the value of the other index.

4.2 Expanding Feature Variations Across the Codebook

The second element of the modifications we have made to the original Kohonen algorithm is to introduce a codebook which expands during training. This is similar to a variation proposed by Rodrigues and Almeida [10], who were seeking a way to reduce the amount of computation in the self-organizing process. We begin with a small number of codewords to represent the range of features in the training set, and then gradually increase the codebook size at various points in training. The codebooks resolution is increased through a linear interpolation to produce new neurons. For example, a 2x2 codebook would be expanded to produce a 3x3 codebook by interpolating between each of the four neighbor pairs, and the center neuron would be produced by finding the mean point of all four vectors. At a later point in the training, the 3x3 network could be expanded to a 5x5 network by repeating the process on each 2x2 sub-block. These periodic expansions continue until the desired network size is reached.

4.3 The Modification Region

One important issue which arises as a result of the expanding network is the schedule over which the modification neighborhood (the region of the codebook deformed by each training vector) decreases in size. A neighborhood radius which covered a given fraction of the network at the training step before an expansion would cover a proportionally smaller fraction of the network after the expansion. Rodrigues and Almeida [10] simply maintained a constant radius of the modification neighborhood throughout the training process (resulting in the fraction of the network covered by this radius decreasing in distinct increments at the points where the network expands). This is consistent with the spirit of the original Kohonen algorithm [5] which decreased the radius of its modification region at distinct increments. However, other implementations of the Kohonen SONN [3] produced better results by using a smoothly decreasing function. Therefore, we implemented the modified SONN with a neighborhood which increased at the moment of network expansion by an amount proportional to the network expansions. The end result is that the modification region decreases smoothly with respect to the size of the overall map.

4.4 The Effects of the Modified Algorithm

The new method of training is an important variation on the original algorithm. To begin with, we avoid producing a twisted network because the network starts out untwisted and hence is resistant to becoming twisted [10]. (A twisted network results when nearby regions of space are represented by distant codebook members and distant regions of space are represented by nearby codebook members.) This is a notable step toward the achieving of global structure for the network, because twisted networks are an automatic contradiction to the idea that codewords which are distant in the book should be different. (Fixed size networks can of course avoid twisting with a large initial neighborhood and a slowly shrinking modification neighborhood, but this greatly increases the training time and results in unwanted sensitivity of the algorithm to the neighborhood size parameters.) However, the major influence of the expanding network is the way in which it interacts with the row- and column-information described in Section 4.1. As mentioned, the column-information feature incorporates the information about the appropriateness of particular rows and columns into its decision on where to place the training vector for updating the codebook. During the early parts of training using the modified algorithm, each row and column encodes what will eventually become a large strip of the final codebook. As a result, the original row- and column-information decisions are in effect decisions as to which global region of the codebook the training vector will be placed in. The formation of new neurons by interpolation causes the newly emerging regions which will be present in the final codebook to be ordered, and the row and column information causes this ordering to occur in a manner which is aligned with the axes or indices of the codebook.

5. RESULTS

Industrial machine tools such as drills emit sounds while they are being used, and the operators have noted that these sounds change as the tool dulls. We are interested in applying an automatic classification system to these sounds. The attachment of an accelerometer to the machine enabled us to capture the vibration patterns of the tool in process. Human analysis of the signals resulted in the extraction of several features which were correlated with dullness. However, this analysis also noted the presence of more structure in the signal but it was not characterized nor was it determined whether the information was related to the dullness of the tool.

We calculated the spectrograms of the vibration patterns, and used the spectral vectors as input data to train our modified SONN feature extractor. The training vectors had a length of 192 (the middle section of 256-point STFTs). The final codebook was of size 5x5. There were 5 drilling sequences available, and training was done on three sets at a time to enable the other two to serve as test series.

Figure 3 shows a typical feature map, trained on holes from three drills. The codebook is a two-dimensional 5x5 codebook, shown row by row. It can be seen that the row index maps the main-carrier frequency feature which had previously been detected by human observation (this is the feature which PCA was unable to map in Section 2). The column index, though, maps something which had not been characterized by the human analysis: the increasing broadband energy in the 13-14 kHz region.

To investigate the usefulness of the codebook as a measure of this new feature, we can map a spectral vector sequence (the sharp drill hole presented in Figure 1) using the codebook and plot the resulting index sequence (shown in Figure 4). The result is a sequence which is cyclical with the revolution of the drill. This cycle is visible in the data, but it could not be easily described or quantified. The SONN now gives us a simple numerical measure of

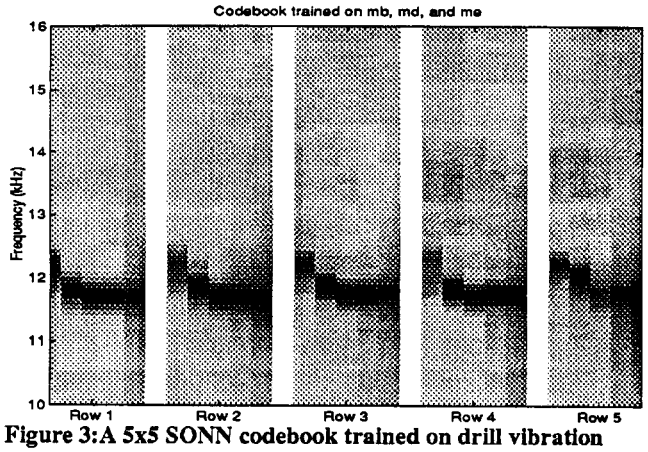

Figure 3:A 5x5 SONN codebook trained on drill vibration spectrograms. The 25 vectors are shown one row at a time, with all 5 columns of a particular row drawn adjacently.

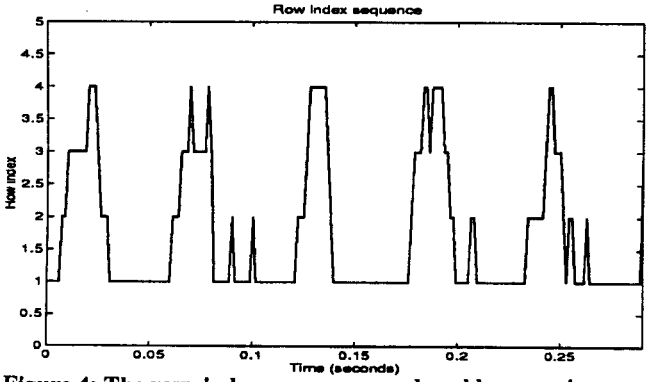

Figure 4: The row index sequence produced by mapping a spectrogram using the codebook of Figure 3 to map the spectrogram of the sharp drill hole in Figure 1.

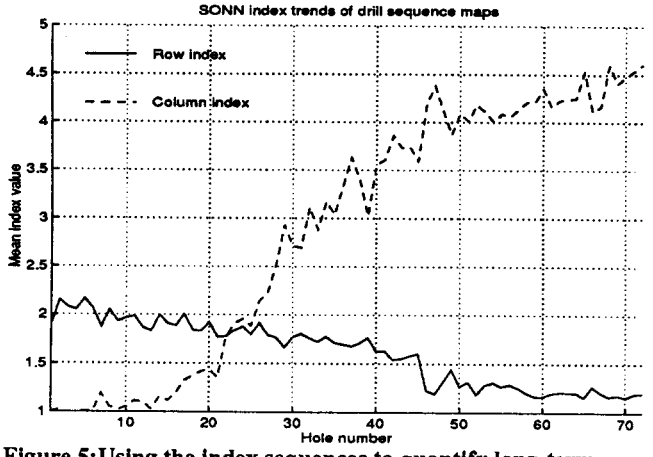

Figure 5:Using the index sequences to quantify long-term changes by plotting the average index value for each hole in the drilling sequence.

the feature.

To determine whether the feature is related to the dulling process, we can plot the average index values over the drilling sequences. Figure 5 shows plots of this over one complete hole set from sharp to dull (the codebook was trained on three other sequences). It can be seen that the new feature (as well as the previously identified feature) is indeed related to the sequence progression and our feature extractor has made a contribution to our ability to characterize the dulling process.

6. SUMMARY

Dull tool monitoring and many other time-series classification applications currently require human selection of the appropriate features for classification, which can miss subtle or complicated features. Standard automatic feature extraction approaches which do exist are single-vector clustering techniques poorly suited to describing continuous variations which characterize the information in time-series patterns. The Kohonen SONN provides a method for describing these variations, but the output of this mapping is not readily interpretable for further processing. We have presented a modification of the Kohonen mapping which is structured so that the map indices are measures of the features found by the algorithm. This feature extractor has been applied to dull tool monitoring problems and has been shown to be a useful approach to extracting continuous feature variations and characterizing them for further processing.

References

[1] S. Narayanan, J. Fang, G. Bernard, and L.Atlas, "Feature Representations for Monitoring of Tool Wear," *Proc of ICASSP*, vol. 6, pp. 137-140, 1994.

[2] D. Dornfeld, W. Koeing, and G. Ketteler, "Present state of tool and process monitoring in cutting," *Proceedings of the International CIRP/VDI Conference*, September, 1993.

[3] E. Erwin, K. Obermayer, and K. Schulten, "Self-Organizing Maps: Ordering, Convergence Properties, and Energy Functions," *Biological Cybernetics*, vol. 67, pp. 47-55, 1992.

[4] K. Mardia, J. Kent, and J. Bibby, *Multivariate Analysis*. New York: Academic Press, 1979.

[5] T. Kohonen, *Self-Organization and Associated Memory*, Third Edition. New York: Springer-Verlag, 1989.

[6] T. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, pp. 1464, 1990.

[7] L. Owsley, *Ordered Vector Quantizers for Feature Extraction*, Master's Thesis, University of Washington, 1994.

[8] L. Owsley and L. Atlas, "Ordered Vector Quantizers for Neural Network Feature Extraction," *Proceedings of the 1993 IEEE Workshop on Neural Networks for Signal processing*, pp 141-150.

[9] H. Ritter and T. Kohonen, "Self-organizing semantic maps," *Biological Cybernetics*, vol. 61, pp. 241-254, 1989.

[10] J. Rodrigues and L. Almeida, "Improving the Learning Speed in Topological Maps of Patterns," *1990 International Neural Network Conference*, vol. 2, pp. 813-816.