# Neural-network-based Classification of Acoustic Transients

**David Montana and Kenneth Theriault**
**Bolt Beranek and Newman, Inc.**
**10 Moulton Street, Cambridge, MA 02138**

## Abstract

For the last year and a half, we have developed systems for detection and classification of acoustic transients. In this paper, we describe the insights and interim results we have so far obtained. We focus in particular on (1) the difficulties of this problem and (2) the design and development guidelines we have evolved for overcoming these difficulties. We illustrate these with examples from the project, and we describe a set of experiments which support many of our development guidelines

## 1. Introduction

Ocean transients form patterns in time-frequency space which human analysts can usually identify aurally and/or visually (via a spectrogram). Recently, there has been a lot of interest in building systems to automatically identify these patterns. Because of the large amount of variability in most classes of transients (see Section 4), matched filters are usually not a viable classification method, and hence recent work has focused on more general methods of pattern classification. Neural networks are a class of pattern classification algorithms which have two important advantages: (1) they have been shown empirically to give good results on a wide range of problems and (2) there exists hardware which implements them efficiently. This is the main reason why much recent work, including our own, has focused on neural-network-based solutions to the problem of detection and classification of acoustic transients.

In this paper, we discuss our research on this problem. The discussion is organized as follows. Section 2 gives an overview of the Neural Classification of Acoustic Transients (NCAT) project. Section 3 presents the general processing architecture we have used. Section 4 examines the major difficulties of this problem as compared with simpler pattern classification problems. Section 5 discusses a set of experiments which support many of our development and design guidelines. Section 6 describes what these guidelines are and provides further justification for their importance.

## 2. Project Description

The goal of the NCAT project has been to build neural-network-based systems for detection and classification of acoustic transients. We have used two fundamentally different datasets, work on which has divided the project into two phases. The first dataset is an unclassified, synthetic dataset. It contains six different signals labeled A-F and described in Table 1. These signals have been coherently replicated in colored Gaussian noise at various signal-to-noise ratios (SNR's). (This coherent replication is one reason why the first dataset is a poor approximation to real ocean data, which has great amounts of

| Signal | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| FreqRange (Hz) | 3K–3.4K | 4K-5K | 3K tone | 3K tone | 150 tone | 250 tone |
| Duration | 10msec | 35msec** | 10msec | 100msec | 1sec | 8sec |

** signal B consists of two pulses of 5msec duration separated by a gap of 25 msec

Table 1. Description of the signals from the first dataset.

variability within most classes of signals.) The second dataset is recorded ocean data. The examples from this dataset cover most of the catalogued classes of ocean transients in the audible region. These examples have been grouped into eleven basic classes, five target and six non-target, which our system is responsible for identifying. We refrain from discussing this dataset in detail due to its classified nature.

We have completed a system to classify signals from the first dataset, and we are presently working on a system to classify signals from the second dataset. Due to the lack of realism of the first dataset, we consider the major output of that phase to be the insights, procedures and tools developed (and used for work on the second dataset). Since we do not yet have results from the second dataset, these insights and procedures are so far the primary results of the project and hence the focus of this paper. (Note that throughout the paper we will illustrate general insights with specific examples from both datasets.)

## 3. General System Architecture

Figure 1 shows a block diagram of the general architecture we have used for all the systems with which we have experimented for both the first and second datasets. This architecture has five basic components, each of which we now briefly discuss:

(1) **Computation of atomic features:** This module consists of transforming a segment of the raw time series into another set of features which is more descriptive of the signal. It often includes some type of filtering (e.g. a prewhitening filter or low-pass filter) as well as one or more Fourier transforms and/or matched filters. (Note that matched filters can be effective for classes of signals which meet the following two condition: (1) small variability at the source and (2) small time-bandwidth product and hence small relative phase shifts at different frequencies.)

(2) **Detection / Normalization:** The detection module decides when there is some signal present as opposed to just noise and hence whether or not to continue to the classification stage. It usually does this by compiling some running estimate of the characteristics of the atomic features in the presence of only noise and looking for sufficiently-sized deviations from the norm to appear signal-like. We have avoided sophisticated detection algorithms because the performance requirements on this module are not very stringent: we need not make the probability of false alarm too small because we have a second chance to reject noise in the classification module, and we need not make the probability of detection too large by detecting very low SNR signals because the classification module cannot classify these signals anyway. Having estimated the noise spectrum, we can use this estimate to normalize each atomic feature by its noise estimate and thus work with SNR's rather than raw amplitudes. This normalization step is optional.

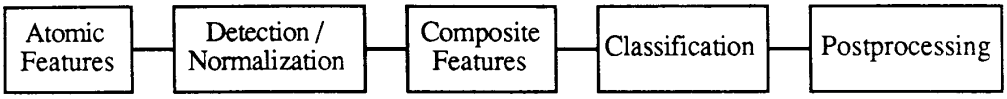


Figure 1. Block diagram of the system architecture.

(3) **Computation of composite features:** Composite features are features computed from the atomic features. They are the inputs to the classification module. For example, for the first dataset for the "high band", the system had ten composite features, which were the peak frequency and SNR for each of five consecutive time frames (see Figure 2). It is this module which is responsible for reducing the size of the features vectors fed to the classification module with minimal loss of relevant information. Hence, the selection of the composite features is a key determinant of system performance, and Section 6 discusses guidelines on how to select them.

(4) **Classification:** This module consists of pure pattern classification, i.e. given a feature vector of predetermined dimension, decide to which class of a predetermined set of classes (including one class for noise/other) it belongs. Any such classifier we use has been trained off-line on a training set of feature vectors of known class. Section 6 discusses the various classification algorithms with which we have experimented.

(5) **Postprocessing:** The postprocessing module performs high-level reasoning about the outputs of the detection and classification modules to make the classification decisions of the system unified and consistent. For example, for a long event consisting of a sequence of short subevents, the postprocessing module can identify the long event based on the classification module's identification of some of the subevents. As another example, if one signal is classified differently with different alignments of the sliding time window, the postprocessing should resolve this inconsistency.

An example instantiation of this architecture is shown in Figure 2, which illustrates the processing used in the final system for the first dataset. We discuss some of the details of this figure in Section 6.

## 4. Complicating Factors

We have so far identified five factors which keep this problem from being a straightforward pattern classification exercise and thus one which requires insight and careful design. We now discuss each of these.

(1) **Small number of examples:** Target transients are inherently difficult to collect. For example, in the second dataset certain classes have less than fifteen examples. Lack of data leads to problems with statistical significance manifest in a variety of forms. Training a classifier with too few examples often leads to overfitting of the data and thus poor generalization (see Section 5). Furthermore, when one class has far fewer examples than some of the others, a training algorithm (such as backpropagation [Rumelhart 86]) which relies on global optimization will often ignore examples from this small class. Evaluation of the classification performance also suffers when there are too few examples. For example, consider a test set that has only five examples for a particular class, one system which classifies four of these correctly, and another system which classifies three correctly. Then, the confidence that the first system is better than the second system is very small despite the fact that there is a 20% difference in percentage correctly classified. A final

casualty of the lack of data is robust preprocessing (i.e., all the processing before the classification module). Without knowledge of the range of variability of each class, it is

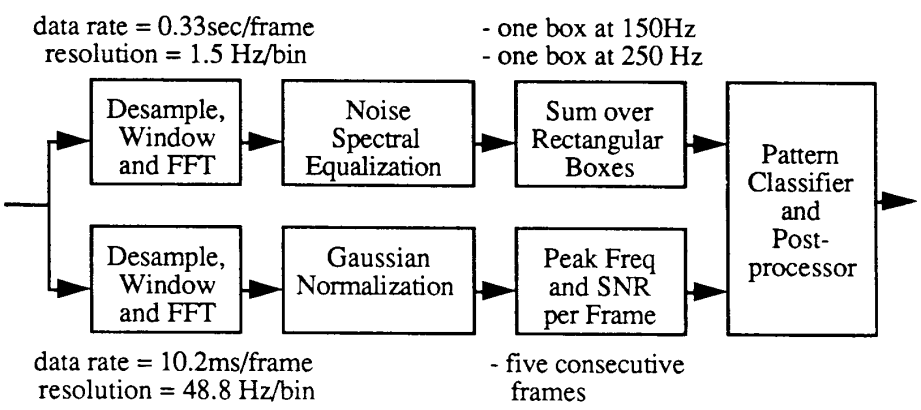


Figure 2. System functionality for the first dataset.

difficult to design composite features which fully capture the information relevant for classification.

(2) **Large amount of data per example:** To illustrate this problem, consider the first dataset. It has a sampling rate of 25kHz and has signals which are at least tens of msec's long. Therefore, there are hundreds, and in some cases thousands, of time samples associated with each example. To avoid gross overfitting of the data when training the classifier, the preprocessing must reduce the data by orders of magnitude.

(3) **Large variability within each class:** Variability in ocean transients comes from a number of sources. For one, there is often great variability at the origin of the sound. For example, "porpoise whistles" may vary significantly not only from animal to animal, but also from sound to sound for the same animal. Second, there are variable distortions due to propagation arising from effects such as non-uniform attenuation and multipath interference. A third source of variability is different background noise characteristics, and a fourth source of variability is different SNR's (the only type of variability captured in the first dataset). Making our system robust with respect to the variability requires careful design and a training set which captures its range.

(4) **Superclasses with very different properties:** We illustrate this complication by considering the problem of choosing an FFT size to classify signals from the first dataset. Signals E and F are narrowband signals separated by only 100 Hz; we therefore would like at least 25Hz frequency resolution and thus an integration time of at least 40msec. Signal B is distinguished by its two-pulse structure, but this structure becomes blurred with integration times much longer than 10msec. So, a single FFT cannot capture all the information necessary for classification. (The solution in this case is shown in Figure 2 and discussed further in Section 6 is the use of multiple FFT's.)

(5) **Ambiguous data:** In ocean data, signals of different classes often occur simultaneously. For instance, in one example from the second dataset labeled as whale cries, there are simultaneous occurrences of whale cries, porpoise whistles, and snapping

shrimp. Most pattern classification algorithms have the implicit assumption of the occurrence of at most one pattern, and it is as yet an unsolved problem how to handle such multiple occurrences. (For now, we are leaving ambiguous examples out of the training set and asking for such examples to be left out of the test set, but this only postpones facing a fundamental issue.)

In Section 6, we discuss a set of development and design guidelines to handle these complications. However, we first examine a set of experiments which further illuminate some of these problems as well as pointing the way to possible solutions.

## 5. Experiments on Feature Vector Size

One of the key issues raised in Section 4 was the need for data reduction during preprocessing to increase the statistical significance of classifier training. We now describe experiments we have performed which attempt to quantify the effect of feature vector size on classifier performance. In particular, these experiments investigated how classifier performance degrades with the addition of irrelevant information.

For these experiments, we constructed a sequence of four training sets. Training Set 1 was generated during work on the first dataset. It has ten features, five classes, and 516 examples. Training Set 2 is the same as Training Set 1 except that we supplemented the ten features of Training Set 1 with five additional features, which were random numbers uniformly distributed between zero and one (and hence contained no information relevant to classification), thus giving a total of 15 features. Training Set 3 is the same as Training Set 2 except with ten irrelevant features added and hence a total of 20 features. Like Training Set 3, Training Set 4 has 20 features. It is obtained from Training Set 3 as follows. Pair each of the true features with one of the irrelevant features. Call the feature values of the $i^{th}$ pair $f_i$ and $g_i$. Then, replace these features values with the values $0.5(f_i+g_i)$ and $0.5(f_i-g_i+1)$, thus mixing up the relevant features with the irrelevant features via linear combinations.

On each of these four training sets we evaluated four different pattern classification algorithms. The algorithms used were: (1) backpropagation [Rumelhart 86] with four hidden nodes, (2) CART [Breiman 84], a decision tree classifier, (3) a probabilistic neural network (PNN) [Specht 88] with sigma = 0.05, which is a close relative of k-nearest neighbors, and (4) weighted PNN (WPNN) [Montana 90b], a variation of PNN which uses a weighted distance function with the weights optimized by a genetic algorithm. The evaluation was performed using the 10-fold cross-validation scheme described in Section 6. The results are shown in Table 2.

| Algorithm \ Training Set | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Backprop | 11(69) | 16(51) | 20(27) | 13(64) |
| PNN | 9 | 94 | 109 | 29 |
| WPNN | 10 | 11 | 11 | 25 |
| CART | 14 | 17 | 18 | 53 |

Table 2. Errors committed by each algorithm on each training set (parenthesized quantities are training errors).

We draw a few conclusions from these results. First, the inclusion of even relatively small amounts of irrelevant information does indeed hurt classifier performance. Second, certain classification algorithms are more robust than others to the inclusion of irrelevant information. Third, different classification algorithms perform best on different data, and (except during experiments where we generate the data and hence know its underlying distribution) it is extremely difficult to predict a prior which algorithm will perform best on given data. (In fact, each classification algorithm has an "inductive bias", and the best algorithm is usually the one whose inductive bias best matches the underlying distribution of the data.)

## 6. Development and Design Guidelines

We have determined a set of development and design guidelines to compensate for the difficulties described in Sections 4 and 5. We now discuss each of these principles and how we have applied them in practice.

(1) **Use the classification algorithm which performs best on the given training set:** As illustrated in Section 5, which classification algorithm is best depends heavily on the data. For this reason we maintain a library of different algorithms, which now includes neural network approaches (backpropagation [Rumelhart 86], PNN [Specht 88], Cascade-Correlation [Fahlman 90] and NINGA [Montana 89,91a]) and non-neural network approaches (k-nearest neighbors and CART [Breiman 84]). We evaluate the performance of each algorithm on the training set and use whichever performs best.

To evaluate the algorithm's performance, we use 10-fold cross-validation [Stone 74] (sometimes called "jackknifing"), which is a way to make statistically significant evaluations of a classifier's performance with relatively small amounts of data. It works by dividing the training set into ten subsets such that the exemplars from each class are equally distributed among the subsets. It rotates through the ten subsets, withholding the chosen one as the test set while training on the other nine. It then compiles the test results from the ten runs into a single confusion matrix. The jackknife procedure increases statistical significance because it effectively uses 90% of the data for training and 100% for testing.

(2) **Match the preprocessing to the data:** This is a qualitative principle which is best illustrated by example. Consider the system shown in Figure 2, which is matched to the signals of interest in a variety of ways. In this system there are two separate processing chains, called the "high band" and "low band", matched to two different superclasses of signals. The low-band superclass consists of signals E and F, which have longer duration and lower frequency and which are narrowband. The high-band superclass consists of signals A-D, which have shorter duration and higher frequency and some of which are broadband.

The low-band processing uses much longer integration times for the FFT's. This not only provides the necessary frequency resolution to distinguish the signals but also provides more apparent SNR for the signals. (By decreasing the frequency resolution, we proportionally decrease the noise power in each frequency bin. This proportionally increase the SNR of the signal in a bin.) The high-band processing uses shorter FFT's to provide the necessary time resolution (see Section 4). The low-band processing uses a detection/normalization technique of noise spectral equalization, which assumes that a signal is narrowband and uses the power at the same time and nearby frequencies to compute a noise estimate. The high-band processing uses Gaussian normalization, which assumes that the signals are of short duration and uses the power at the same frequency and

previous times to compute its noise estimate. Finally, the features of the high band are designed to let the classifier learn the time-frequency patterns while the features of the low-band take advantage of the fact that these patterns are known a priori.

(3) **Limit the number of composite features:** There are certain rules of thumb for statistical pattern classification that the ratio of the number of exemplars per class to the number of features needs to be sufficiently large (e.g., greater than 3) to ensure statistical significance. While such rules are useful, they ignore an important factor, which is how much information these features contain. Even when the number of features is small, we can always do better by eliminating irrelevant features. Conversely, even when the number of features is large, we can do worse by eliminating features with relevant information.

One method of reducing the number of features without losing the important ones is automatic feature selection. This is an automated method of deciding which features to keep and which to discard to achieve optimal classifier performance. Some experiments have indicated that a genetic algorithm is the best method of automatic feature selection [Simmons 90]. If after automatic feature selection the classifier still shows signs of overfitting to the data, it is then time for human intelligence to work on consolidating the information form the selected feature set into a smaller number of features.

(4) **Employ iterative redesign:** Iterative redesign is the process of cycling through the following two steps: (i) evaluate the system and (ii) modify the system based on this evaluation. We employ two distinct types of iterative redesign. The first is error-driven feature selection, whereby new features are designed to distinguish between those classes most often confused. For example, consider the confusion matrix shown in Figure 3. According to it, signals C and D are being confused three times. Since the difference between these signals is their large difference in duration, we can probably eliminate this confusion by explicitly computing signal duration as a composite feature.

| True Class \ Chosen Class | A | B | C | D | N / O |
|---|---|---|---|---|---|
| A | 56 | 0 | 0 | 0 | 2 |
| B | 0 | 50 | 0 | 0 | 2 |
| C | 2 | 0 | 62 | 2 | 0 |
| D | 0 | 0 | 1 | 25 | 0 |
| Noise/Other | 1 | 1 | 0 | 0 | 312 |

Figure 3. Sample confusion matrix for the "high band" of the first dataset.

The second type of iterative redesign is remedial training [Montana 90,91a] (called "windowing" in the machine learning literature). Its operation is pictured in Figure 4. For those classes (generally non-target and noise classes) with too many examples to all be included in the training set, remedial training iteratively adds to the training set only those examples that actually help to redefine the decision boundaries. This leads to continual improvement of the classifier

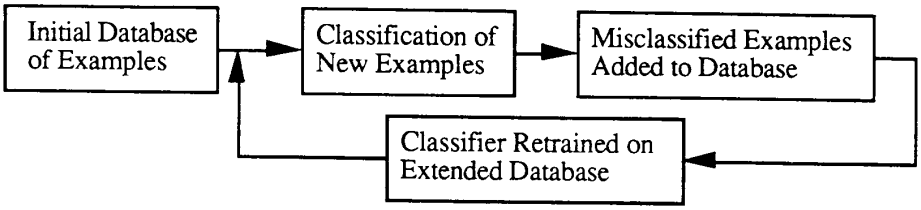


Figure 4. Cycle of operation for remedial training.

## 7. Conclusion

In this paper, we have described the problem of detection and classification of acoustic transients and discussed the NCAT project, whose goal is to build systems for solving this problem. We have provided an overview of the various functional components of our systems. We have discussed the major difficulties which make this problem harder than straightforward pattern classification problems. We have examined the results of experiments which further illustrate these complications as well as providing insights into their solutions. Finally, we have described a set of design and development guidelines which allow us to overcome these difficulties.

## Acknowledgements

We would like to thank our contract monitors, Barbara Yoon, Tom McKenna, and Ray Glass, and Morgan Grover of DGI, for obtaining the datasets and guiding the project to its present point. We would also like to thank the other major contributors to the NCAT project at BBN: Ivan Vazquez, Karl Haberl, and Mike Sullivan. This work has been supported by DARPA through ONR under Contract N00014-89-C-0264 as part of the Artificial Neural Networks initiative.

## References

**[Breiman 84]** L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Monterey, CA: Wadsworth, 1984.

**[Fahlman 90]** S.E. Fahlman and C. Lebiere, "The Cascade-Correlation learning architecture," Tech Report CMU-CS-90-100, 1990.

**[Montana 89]** D.J. Montana and L. Davis, "Training Feedforward Neural Networks using Genetic Algorithms," *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1989, pp. 762-767.

**[Montana 90]** D.J. Montana, "Empirical Learning using Rule Threshold Optimization for Detection of Events in Synthetic Images," *Machine Learning*, vol. 5, no. 4, pp. 427-450, 1990.

**[Montana 91a]** D.J. Montana, "Automated Parameter Tuning for Interpretation of Synthetic Images," in *The Genetic Algorithms Handbook*, edited by L. Davis, Van Nostrum Reinhold, pp. 282-311, 1990.

**[Montana 91b]** D.J. Montana, "A Weighted Probabilistic Neural Network," summary submitted to NIPS-91.

**[Rumelhart 86]** D. Rumelhart, G. Hinton, and R. Williams, "Learning Representations by Backpropagating Errors," *Nature*, vol. 323, pp. 533-536, 1986.

**[Simmons 90]** A. Simmons and D. Hamilton, "Sonar Signal Processing with Neural Networks," presented at the Government Neural Network Applications Workshop, 1990.

**[Specht 88]** D.F. Specht, "Probabilistic Neural Networks for Classification, Mapping or Associative Memory," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 1, pp. 525-532, 1988.

**[Stone 74]** M. Stone, "Cross-validatory Choice and Assessment of Statistical Predictions," *Journal of the Royal Statistical Society*, vol. 36, pp. 111-147, 1974.