To appear *IEEE Trans. on Aerospace and Electronic Systems*, October 2007.

Hybrid Discriminative/Class-Specific Classifiers for Narrow-Band Signals

Brian F. Harrison and Paul M. Baggenstoss
Naval Undersea Warfare Center
Sensors and Sonar Systems Department
Code 1521, Building 1320
Newport, RI 02841-0001
Phone: 401-832-8239
Fax: 401-832-7453
Email: harrison_bf@ieee.org

Abstract

The class-specific (CS) method of signal classification operates by computing low-dimensional feature sets defined for each signal class of interest. By computing separate feature sets tailored to each class, i.e., class-specific features, the CS method avoids estimating probability distributions in a high-dimension feature space common to all classes. Building a CS classifier amounts to designing feature extraction modules for each class of interest. In this paper we present the design of three CS modules used to form a CS classifier for narrow-band signals of finite duration. A general module for narrow-band signals based on a narrow-band tracker is described. The only assumptions this module makes regarding the time evolution of the signal spectrum are: (1) one or more narrow-band lines are present, (2) the lines wandered either not at all, e.g., CW signal, or with a purpose, e.g., swept FM signal. The other two modules are suited for specific classes of waveforms and assume some *a priori* knowledge of the signal is available from training data. For *in situ* training, the tracker-based module can be used to detect as yet unobserved waveforms and classify them into general categories, for example short CW, long CW, fast FM, slow FM, etc. Waveform-specific class-models can then be designed using these waveforms for training. Classification results are presented comparing the performance of a probabilistic conventional classifier with that of a CS classifier built from general modules and a CS classifier built from waveform specific modules. Results are also presented for hybrid discriminative/generative versions of the classifiers to illustrate the performance gains attainable in using a hybrid over a generative classifier alone.

1 Introduction

The classical Bayesian classifier computes a common feature space for all classes of interest from which the probability density function (PDF) of the features is estimated. If the dimension of the feature space is too high, severe errors in estimating its PDF will occur, resulting in classification errors. Additionally, the amount of training data required to accurately estimate the PDF increases exponentially with feature dimension [1]. If the feature dimension is too low, the signal classes will overlap in feature space, again causing classification errors. This tradeoff is what is known as the "curse of dimensionality." Although to a lesser degree, this also affects discriminative classifiers which must construct decision boundaries in a high dimensional space.

The class-specific (CS) method of signal classification [2] avoids the curse of dimensionality by computing individual low-dimensional feature sets defined for each class, i.e., class-specific features. A PDF projection operator, which amounts to a data-dependent correction term depending on the feature transformation, is applied to the feature PDFs to convert them back to likelihood functions defined on the raw data domain where classification decisions are made. The fundamental building block of a CS classifier is the feature extraction module. Each module computes features specific to a given class as well as its corresponding PDF projection operator. Multiple modules can also be linked in series to form more sophisticated processing chains.

Classifiers which seek to discriminate between classes without attempting to estimate the PDFs are called *discriminative* classifiers, while those that seek to fully describe a class by modeling the PDF are called descriptive classifiers. They are also called *generative* classifiers because, being statistical models, they may be used to generate synthetic data. In the limit, both types of classifiers are equivalent because if each class is fully described, all the information exists to discriminate. Classical theory can be regarded as both generative and discriminative since the likelihood ratio forms a decision boundary which is discriminative. There has been some attention paid recently to the comparison of generative and discriminative approaches to classification [3],[4]. The widely-held belief is that discriminative classifiers outperform generative approaches - simply because generative approaches attempt to solve a more difficult problem than necessary. In other words, while the end goal is to draw decision boundaries in the feature space, the generative approach does so as a by-product of attempting to solve the more difficult PDF estimation problem - while the discriminative approach estimates the decision boundaries directly. There is ample evidence that discriminative approaches outperform generative approaches in one-to-one comparisons [4],[5]. This has led to the large amount of attention paid to discriminative approaches, most notably, support vector machines (SVM)s [6],[7]. Why then use the CS method, which is a generative method?

There are two ways to answer this question. The first answer, the least important but necessary to say nevertheless, is that just as there are cases when a discriminative classifier can outperform a generative classifier, there are cases when a generative classifier performs better. There has been increasing recognition of the strengths of the generative approach under some conditions [8]. But the fact that errors of discriminative and generative approaches are largely uncorrelated [9] has opened the door to a number of hybrid approaches that have made performance advances over SVMs by combining the two approaches into a single classifier [9],[10],[11],[12].

The second and more pertinent answer is that there is something misleading about the question itself. Discriminative classifiers have only been shown to outperform generative classifiers operating in the same feature space. It does not at all recognize that each class may be best characterized using a different set of parameters and therefore different feature space. While the discriminative classifier is forced to operate in a common high-dimensional feature space, who is to say that a generative classifier does not exist that operates in a totally unexpected way, using a different low-dimensional feature space to describe each class, that outperforms the discriminative classifier operating in a high-dimensional space? Most so-called generative classifiers in the literature are not truly generative. This is because they operate in a feature space made up of features selected for their discriminative power. Their ability to generate raw data is questionable. Instead, they generate a set of synthetic features from which we would have a difficult time generating synthetic samples of raw data. For this reason, the full potential of the hybrid idea is not fulfilled. The only way to fully exploit the hybrid idea is to develop the very best possible generative classifier that is not restricted to use a common set of features and the features do not need to be selected for discrimination power. The CS method provides a formal approach to finding the best features and models for a particular problem.

The two primary objectives of this paper are: (1) to present the development of new modules for a CS classifier and demonstrate their utility by using them to build CS classifiers and (2) to demonstrate the performance gains possible from using a hybrid classifier by developing hybrid discriminative/generative classifiers from the CS classifiers.

Each new CS module that is developed extends the applicability of the CS method. In this paper, we present three modules designed to form CS classifiers for finite-duration narrow-band signals. One is a general module which is based on a narrow-band tracker and makes few assumptions about the spectral content of the signal. It assumes only that the signal contains one or more narrow-band spectral lines and that these lines either remain at a constant frequency or change frequency in a characteristic way over time. This module can be used as the building block for a classifier which assigns signals to broadly defined categories. The versatility of such a classifier is that it allows for the classification of previously unforeseen waveforms. The other two modules are appropriate for specific types of waveforms and assume some *a priori* knowledge of the signal is available from training data. Of these, one is applicable to frequency modulated (FM) signals and computes as features autoregressive coefficients from matched-filtered, segmented data. Knowledge of the time-frequency distribution of the FM waveform is required in designing the matched filter. The other module is trained as a joint class-model for narrow-band signals plus autoregressive noise. It uses training data to compute a signal subspace from the spectrum of the waveform. A noise subspace is generated from the set of cosine basis functions corresponding to the computation of the autocorrelation function. Features are then computed as the projection of the segmented data's spectrum onto the signal and noise subspaces.

Following a brief overview of the CS classification method in the next section, we will describe the three modules in Section 3. Data results using CS classifiers built from the modules will be presented in Section 4. The results of the CS classifier will be compared to that of a conventional classifier. Results using hybrid classifiers developed from the generative classifiers will be presented to illustrate the performance improvement gains in using a hybrid classifier over that of using the generative alone.

2 Summary of Class-Specific Method

We present here a brief overview of the CS method and some of the fundamental concepts in designing a CS feature module. For a detailed description of the CS method, the reader is referred to [2], [13].

A CS feature module computes features $\mathbf{z}_i$ from the raw data $\mathbf{x}$, where the features computed by module i are specific to the i^{th} data class of M class hypotheses H_i. Using CS features enables the feature spaces to be of low dimension. Therefore, the PDF of the features $p(\mathbf{z}_i|H_i)$ for each data class $i = 1, ..., M$ can be accurately estimated using training data.

To classify a raw data event $\mathbf{x}$, the CS method computes features corresponding to each class $\mathbf{z}_i = T_i(\mathbf{x})$, evaluates the likelihoods $\widehat{p}(\mathbf{z}_i|H_i)$, and then converts the likelihoods back to the raw data domain, where classification decisions are made, using the PDF projection operator as

$$p_p(\mathbf{x}|H_i) = \frac{p(\mathbf{x}|H_{0,i})}{p(\mathbf{z}_i|H_{0,i})}\widehat{p}(\mathbf{z}_i|H_i), \tag{1}$$

which is an approximation to $p(\mathbf{x}|H_i)$. The ratio

$$\mathbf{J}(\mathbf{x}, T_i, H_{0,i}) = \frac{p(\mathbf{x}|H_{0,i})}{p(\mathbf{z}_i|H_{0,i})} \tag{2}$$

in (1) is the PDF projection operator which is called the "J-function." This operator converts the feature PDFs to raw data PDFs. It is very important that the J-function be accurate such that $p_p(\mathbf{x}|H_i)$ results in a valid PDF. The J-function is based on a CS reference hypothesis $H_{0,i}$ and allows for a fair comparison of likelihoods computed from different feature sets. The primary consideration in selecting an $H_{0,i}$ is that both the numerator and denominator of the resulting J-function can be written in closed form, or else to a good approximation, out to the (far) tails of the distribution. Another important consideration is that in-class variations are compactly described with respect to $H_{0,i}$. This results in a low-dimensional feature set. For example, if H_i is a sinusoid in white noise, then $H_{0,i}$ should not be selected to be a colored noise assumption. Choosing a white noise assumption for $H_{0,i}$ would only require computing features of the sinusoid to distinguish H_i from $H_{0,i}$. Alternatively, a colored noise assumption for $H_{0,i}$ would require computing additional features of the background noise to distinguish H_i from $H_{0,i}$. One mitigating factor is that no matter which $H_{0,i}$ is chosen, as long as the primary consideration is met, the PDF projection theorem guarantees that the result is a PDF. Given that, likelihood comparisons can be used to empirically evaluate the choices made for $H_{0,i}$ and feature selection. Further guidance in selecting an appropriate $H_{0,i}$ and a procedure for validating the accuracy of the J-function are given in [2]. Substituting (1) into the expression for the optimal Bayesian classifier given by

$$i^* = \arg\max p(\mathbf{x}|H_i)p(H_i), \quad i = 1, ..., M, \tag{3}$$

where $p(H_i)$ is the prior probability of class H_i, results in the CS classifier,

$$i^* = \arg\max \frac{p(\mathbf{x}|H_{0,i})}{p(\mathbf{z}_i|H_{0,i})}\widehat{p}(\mathbf{z}_i|H_i)p(H_i), \quad i = 1, ..., M. \tag{4}$$

The general form of a CS feature module computes features $\mathbf{z}_i = T_i(\mathbf{x})$ and the associated J-function $\mathbf{J}(\mathbf{x}, T_i, H_{0,i})$.

Feature modules can also be linked in a serial processing chain to compute additional transformations and conditioning of features. The overall J-function for this configuration is equal to the product of the J-functions of each of the modules in the chain. In the case of computing log-likelihoods, the overall J-function would be the sum of the log J-function values of each module. Assuming a processing chain consisting of 3 modules, (1) would become

$$\log p_p(\mathbf{x}|H_i) = j_1 + j_2 + j_3 + \log \widehat{p}(\mathbf{z}|H_i), \tag{5}$$

where j_l is the log of the J-function of module l, and $\mathbf{z}$ is the feature vector computed by module 3.

3 Class-Specific Feature Modules

In this section, we present the three CS feature modules designed for narrow-band signals. These modules compute features based on the frequency spectrum of the data. The first of these is called the class-specific tracker and is a general module applicable for a wide range of waveforms. The other two modules are waveform specific and require prior knowledge of the signal of interest's spectrum for training. Used together, the general and specific modules can be combined to form a versatile classifier for *in situ* training. As new unforeseen waveforms are detected by the general class-models, a small number of these events can be used to train a specific class-model which can then be added to the classifier. Additionally, as new types of false alarms are detected, they also can be trained away using specific class-models.

3.1 Class-Specific Tracker

The class-specific tracker (CST) was developed out of the need to have a general module applicable to a variety of narrow-band signal types. A good set of features to use to observe the duration and variation of the spectral lines would be the spectrogram bins of the signal. For example, using the sequence of bin observations over time to train a hidden Markov PDF model for each general class. However, using the spectrogram bins would result in a high-dimensional feature set and as such would be impractical for PDF estimation. Instead, we propose the CST as a novel approach to this problem. The CST uses a narrow-band tracker combined with Bayes' theorem to efficiently compute log-likelihoods using spectrogram bins as features.

The CST begins by computing the spectrogram of the data using non-overlapping, rectangular windowed segments. We assume only that the spectrogram contains one or more spectral lines and that these lines either remain at a constant frequency or change frequency in a characteristic manner, e.g., swept FM waveform. The spectrogram is thresholded to detect narrow-band lines and then input to a narrow-band tracker which tracks the spectral lines computing a number of statistical parameters describing each line tracked, e.g., track velocity, bandwidth, and age. These parameters are utilized in computing the PDF of the spectrogram bins. An example tracker output surface is shown in Fig. 1 for a hyperbolic FM signal. The white dots connected by line segments is the track of the signal. The individual dots are random spectral lines that were detected, but resulted in no tracks.

To efficiently compute the PDF of the spectrogram, the CST uses Bayes' theorem to decompose the PDF of the spectrogram into a product of conditional PDFs. Assume a N-point fast Fourier transform (FFT) is used to compute the spectrogram of a raw data event consisting of a narrow-band signal in Gaussian noise. Define $\mathbf{y}(k)$ as the length $N/2+1$ vector of spectrogram bins (non-negative frequencies) of the real-valued raw data time-series vector $\mathbf{x}(k)$ for time-block segment k. We can write the PDF of the spectrogram as

$$p(\mathbf{y}(1), \mathbf{y}(2), ..., \mathbf{y}(K)) = p(\mathbf{y}(1))\; p(\mathbf{y}(2)|\mathbf{y}(1))\; p(\mathbf{y}(3)|\mathbf{y}(1), \mathbf{y}(2))\; ...\; p(\mathbf{y}(K)|\mathbf{y}(1), ..., \mathbf{y}(K-1)), \tag{6}$$

where K is the total number of time-block segments. While (6) is an exact equation, we will approximate each term only "in spirit." Each term is of the form $p(\mathbf{y}(k)|\mathbf{y}(1), ..., \mathbf{y}(k-1))$ which we approximate by developing a PDF estimate for segment k based only on information from prior segments up to and including segment $k-1$. We propose to approximate the expected power spectrum of the current segment based on prior segments. From prior segments, we can track the frequency, frequency rate, bandwidth, and amplitude of any existing narrow-band signals, plus the smooth background noise spectrum. These parameters can then be extrapolated to the current segment using tracker parameters to derive an estimate of the current segment's spectrum. The power spectrum estimate that we derive from prior segments, evaluated at each FFT bin frequency, will be used as a mean estimate of each magnitude-squared FFT bin in the current segment. The expected value of the n-th magnitude-squared FFT bin for segment k, derived from previous segments, is denoted by $\mu_{n,k}$.

It is well known that for any zero-mean Gaussian input, the magnitude-squared FFT bins are central chi-square random variables. The first bin (zero frequency) and $N/2+1$ bin (half sample rate) are real and therefore chi-square with 1 degree of freedom. The remaining (complex) bins are exponentially distributed (central chi-square with 2 degrees of freedom). Since we have not used a data window function in computing the FFT, i.e., we used a rectangular window, we have some justification to use the large-N approximation that the bins are statistically independent [14], although the underlying assumption for this approximation is that the spectrum is smooth, which is contradictory to the existence of narrow-band signals. Nevertheless, we find the approximation to be useful. We also assume conditional independence of the segments. This essentially ignores any sample-to-sample correlation that occurs between the abutting samples in adjacent segments. Based on these assumptions, we can write the conditional PDF of $\mathbf{y}(k)$ as

$$\begin{aligned} p(\mathbf{y}(k)|\mathbf{y}(1),...,\mathbf{y}(k-1)) \;\simeq\; & \frac{e^{-y_1(k)/(2\mu_{1,k})}}{\sqrt{2\pi y_1(k)\mu_{1,k}}} \\ & \cdot\frac{e^{-y_{N/2+1}(k)/(2\mu_{N/2+1,k})}}{\sqrt{2\pi y_{N/2+1}(k)\mu_{N/2+1,k}}} \\ & \cdot\prod_{n=2}^{N/2}\left[\frac{1}{\mu_{n,k}}\exp\left\{\frac{-y_n(k)}{\mu_{n,k}}\right\}\right], \end{aligned} \tag{7}$$

which is a product of chi-squared univariate PDFs with 1 degree of freedom for bins 1 and N/2+1, and 2 degrees of freedom (exponential) for bins 2 through N/2. In all cases, $\mu_{n,k}$ is the mean of the distribution for bin n. In the log domain,

$$\begin{aligned} \log p(\mathbf{y}(k)|\mathbf{y}(1),...,\mathbf{y}(k-1)) \;\simeq\; & -\frac{1}{2}\,\log\left\{2\pi y_1(k)\mu_{1,k}\right\} - \frac{y_1(k)}{2\mu_{1,k}} \\ & -\frac{1}{2}\,\log\left\{2\pi y_{N/2+1}(k)\mu_{N/2+1,k}\right\} - \frac{y_{N/2+1}(k)}{2\mu_{N/2+1,k}} \\ & +\sum_{n=2}^{N/2}\left[-\log\mu_{n,k} - \frac{y_n(k)}{\mu_{n,k}}\right]. \end{aligned} \tag{8}$$

Note that while this is an approximation (we use $\simeq$), it is an exact form for a PDF, so it integrates identically to 1. The importance of this is that after we combine $p(\mathbf{y}(1),...,\mathbf{y}(K))$

with the J-function to obtain the raw data PDF, we have an exact form of a raw-data PDF. To abide by the spirit of (6), the mean value of each bin $\mu_{n,k}$ for segment k can only be determined from information obtained from segments up to and including segment $k-1$. Therefore, we need a procedure of predicting the values of $\mu_{n,k}$ using prior information.

An iterative procedure to predict the values of $\mu_{n,k}$ at segment k from prior information was devised using Gaussian mixture (GM) modeling and the tracker velocity estimates. Note that we are using the GM not as a statistical model of the $N/2+1$ dimensional space, but instead as a smooth function composed of a sum of Gaussian-shaped functions to approximate the spectrogram at each time step. First, a GM model is fit to the spectrogram segment, $\mathbf{y}(k-1)$. Gaussian modes are placed centered on the bin of each significant peak, as determined by the tracker, in $\mathbf{y}(k-1)$. Two additional single-sided Gaussian modes (i.e., extending from their mean value in a single direction only), whose standard deviation is fixed such that each encompasses one-half of the frequency spectrum, have their means fixed at bins $n = 1,\ N/2+1$ to model the power spectrum of the noise. If no significant peaks were indicated from the tracker information for segment k, the GM model would consist of only a model of the noise. The GM model of $\mathbf{y}(k-1)$ is then input to a expectation-maximization (EM) algorithm which finds the set of GM parameters producing the best model. These parameters consist of the peak bin locations, the standard deviations of the GM modes and their associated mixing weights. Let the GM model of $\mathbf{y}(k-1)$ produced by the EM algorithm be denoted as $\tilde{\mathbf{y}}(k-1)$. From the tracker velocity estimates at $k-1$, we can predict the locations of the peaks at k. Using the GM parameters for $\tilde{\mathbf{y}}(k-1)$ except replacing the given peak bin locations at $k-1$ with the predicted peak locations results in a GM model producing the predicted mean values $\mu_{n,k}$. These values are then used in computing the conditional log-PDF in (8) for segment k. In this manner, we can compute the log-PDF of the spectrogram $\log p(\mathbf{y}(1), ..., \mathbf{y}(K))$ as the sum of the conditional log-PDF terms over all of the segments $k = 1, ..., K$.

Suppose the first appearance of a spectral line in a given bin, i.e., the first sample in a track, begins at time k_1. At time $k_1 - 1$, the predicted GM model for $\mathbf{y}(k_1)$ computed from

$\mathbf{y}(k_1-1)$ would not be able to anticipate the advent of this new track. Therefore, this abrupt onset will result in (7) computing a poor PDF estimate since $\mu_{n,k}$ for the bin will have been estimated assuming noise only. To compensate for abrupt bin amplitude changes an alternative method to (7) is to use a exponential mixture PDF to compute the conditional PDFs. This PDF combines (7) with an additional exponential PDF of fixed mean μ_0 to model the onset of spectral lines as

$$p_m(\mathbf{y}(k)|\mathbf{y}(1),...,\mathbf{y}(k-1)) = (1-\alpha)p(\mathbf{y}(k)|\mathbf{y}(1),...,\mathbf{y}(k-1)) + \alpha p_0(\mathbf{y}(k)) \tag{9}$$

with

$$p_0(\mathbf{y}(k)) = \prod_{n=1}^{N/2+1} \frac{1}{\mu_0} \exp\left\{\frac{y_n(k)}{\mu_0}\right\} \tag{10}$$

and mixture weight $\alpha \ll 1$. Training data is used in an EM algorithm to optimize the values of μ_0 and α for a given signal class. The value of μ_0 is typically determined to be on the order of the magnitude of a spectral peak for the given signal class.

As described in Section 2, the CS method converts likelihoods from feature space back to the raw data domain where classification decisions are made. The log J-function for the magnitude-squared DFT (i.e., for the feature transformation that produces $\mathbf{y}(k)$ from $\mathbf{x}(k)$), assuming an H_0 of independent zero-mean Gaussian noise of variance 1, is found in [2] and can be simplified to

$$\begin{aligned} J(\mathbf{x}(k)) &= \log\left\{\frac{p(\mathbf{x}(k)|H_0)}{p(\mathbf{y}(k)|H_0)}\right\} \\ &= -(N/2-1)\log 2\pi + N/2 \log N + \frac{1}{2}\left(\log y_1(k) + \log y_{N/2+1}(k)\right). \end{aligned} \tag{11}$$

Because H_0 assumes independent Gaussian noise at the input of the module, the segments are independent. Thus, the complete PDF of the raw data is

$$\log p(\mathbf{x}(1),\mathbf{x}(2),...,\mathbf{x}(K)) = \log p_m(\mathbf{y}(1),\mathbf{y}(2),...,\mathbf{y}(K)) + \sum_{k=1}^{K} J(\mathbf{x}(k)). \tag{12}$$

To use the CST as a class-model for a given general signal class, training data representative of the signal class is used to optimize some of the tracker parameters. For our purpose, we have defined five general signal classes as short continuous wave (CW), medium

CW, long CW, slow FM, and fast FM. The first step in training the module for each of the signal classes is to determine the optimum FFT size N for each. Using training data and a nominal set of tracker parameters, the log-likelihood $\log p(\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(K)|N)$ is computed for a given value of N over each of the training events and summed to form a total log-likelihood. For this stage, the mixture weight, α, in (9) is set to 0. The value of N which achieves the highest log-likelihood is used for that class. Note that in practice FFTs computed from overlapped data segments can be used to produce more stable tracks provided only the spectrogram segments appropriate to maintain statistical independence of the bins are used to compute (12). For example, if 50% overlapped data segments are used, then (12) is computed using only the odd indexed spectrogram segments $\mathbf{y}(1)$, $\mathbf{y}(3)$, $\mathbf{y}(5)$, etc. Next, the parameters for the mixture PDF μ_0 and α are optimized for each class using the previously determined value of N. This is done using an EM algorithm to perform an iterative search over the mixture parameters until the total log-likelihood converges. The EM algorithm is initialized using a small value for $\alpha \ll 1$ and a value for μ_0 on the order of the average amplitude of the spectral peaks for the given training data. Finally, using the previously determined values of N, μ_0, and α for each class, the tracker parameters λ and $\mathbf{w}$ are optimized for each class. The parameter λ is the target maneuvering index. The vector $\mathbf{w}$ is a two-element vector of association-window width parameters. The first element defines how far away, in FFT bins, a threshold crossing can be from an existing track. The second element defines how close two tracks can be before one of them is eliminated. An iterative approach is used to search over λ and $\mathbf{w}$ which moves along a path of increasing total log-likelihood until convergence. This results in a set of CST parameters for each class which define the class-model for the given class. For each class $i = 1, ..., 5$, we have the parameters

$$\theta_i = \begin{bmatrix} N_i \\ \mu_{0i} \\ \alpha_i \\ \lambda_i \\ \mathbf{w}_i \end{bmatrix}. \tag{13}$$

The CST can be used as a class-model for each of the five general signal classes by applying the appropriate parameter set. A 5-class classifier can be built by processing data through a parallel set of 5 CSTs, each parametrized by one of the θ_i.

3.2 Class-Specific Module for Frequency Modulated Signals

When the signal of interest is a known FM signal, the module described next can be trained as a class-model for it. The first stage of the module applies an all-pass matched filter (MF) to pulse compress the FM signal of interest. Prior knowledge of the time-frequency trajectory of the FM signal is therefore required to design the replica for the MF. The design of a unique MF replica is required in order to work within the CS paradigm. A MF replica which results in an all-pass transformation was designed as a one-to-one mapping that has a Jacobian with a unit determinant. This eliminates the requirement to derive a J-function (see [13]). The MF replica has the magnitude spectrum of an all-pass filter while still producing the desired pulse compression. This replica uses a combination of linear FM (LFM) signals with a replica of the FM signal of interest. The LFM signals are used to fill out the spectrum in the bands outside of the FM signal's band. As a synthetic example, Fig. 2 shows the power spectral density and the time-frequency distribution of such a replica for a hyperbolic FM (HFM) signal. The HFM signal has a bandwidth from 20 to 30 kHz. One LFM signal is inserted prior to the HFM signal which sweeps from 0 to 20 kHz and another is appended to the HFM signal which sweeps from 30 to 50 kHz (one-half of the sampling frequency for this example). The LFM signals were designed to ensure that phase continuity was maintained at the LFM-HFM transitions.

Implementation of the MF in the module is done in the frequency domain. An FFT of the replica is computed and it's spectrum is conditioned such that the magnitude of each of the bins is normalized to one. Because the magnitude of each bin is constrained to equal one, this results in an all-pass filter. This normalized spectrum is multiplied with the input signal's FFT and the resulting spectrum is transformed back to a time series using an inverse FFT.

The overall processing performed by the module is relatively straightforward. A detailed description of the steps taken in the progression of the development of this module is given in [15]. The final design will be presented here. The first step in the module is to apply the MF described above to the input signal which is then input to a CS module that computes autoregressive (AR) coefficients as features. A detailed description of the AR module as well as the derivation of it's J-function are given in [2]. The basic operation of the AR module is to segment the input time-series into N-sample segments and compute a P^{th} order AR model for each segment, where the values of N and P are specific to a given signal class. This is followed by some conditioning of the AR coefficients to better facilitate estimating the PDF of the features using a GM model. A hidden Markov model (HMM) is then used to statistically model the sequence of AR features. Training the module for a given FM signal class consists of using training data to perform a joint optimization over N and P to determine the values that maximize the total log-likelihood. Note that an ideal input signal will appear as a short pulse at the MF output having the same spectral character as the input. In this case, a small value of N will be appropriate. However, with distortion, imperfect replicas, etc. the energy will be spread in time requiring a larger value of N. The value of P should be sufficient to represent the signal spectrum.

3.3 Spectral Projection Class-Specific Module

The spectral projection (SP) module for CS feature extraction is applicable to CW signals consisting of one or more tones. It is also applicable to narrow-band pulses and FM signals after processing with a matched filter. The SP module is trained as joint class-model for CW signals plus autoregressive noise. It uses training data to generate a signal subspace from the spectrum of the signal class of interest. Stationarity of the spectral peaks is necessary so that the signal subspace be low rank, thereby producing a low-dimensional feature space. Features are computed by projecting the spectrogram of the data onto the signal and noise subspaces. The noise subspace is generated from a subset of the cosine basis functions used to compute the autocorrelation function (ACF).

The first stage in training the SP module for a given signal class is the determination of the signal subspace. Using training data, spectrograms for all of the training events are computed by segmenting each event into L-sample segments and computing an N-point FFT over each segment. Note that if the signal of interest is an FM signal it must be matched filtered, using the matched filter described in Section 3.2, prior to computing the spectrograms. It is assumed that $L \geq N$ and that L is evenly divisible by N. With $L > N$, $\frac{L}{N}$ N-point spectrograms are computed and averaged together for each segment. The spectrograms over all of the training events are collected together as the columns of the matrix $\mathbf{Y} = [\mathbf{y}(1), ..., \mathbf{y}(J)]$. An eigen-decomposition of $\mathbf{YY}^H$ is then computed as $\mathbf{YY}^H = \mathbf{U\Sigma U}^H$, where the columns of $\mathbf{U}$ are the eigenvectors and the diagonal elements of $\mathbf{\Sigma}$ are the corresponding eigenvalues of $\mathbf{YY}^H$. The eigenvectors corresponding to non-zero eigenvalues are the basis vectors for the column space of $\mathbf{Y}$.

Generation of the noise subspace is based on computing the ACF of the segmented data. The assumption here is that the noise background will be broadband in nature and can be well represented by a small number of ACF lags. The first $P+1$ ACF lags can be computed from the spectrogram as

$$r_k = \frac{1}{N^2} \sum_{i=1}^{N/2+1} y_i \epsilon_i \cos\left\{\frac{2\pi(i-1)k}{N}\right\}, \quad k = 0, 1, ..., P, \tag{14}$$

where $\epsilon_i = 1$ for $i = 1, N/2+1$, and $\epsilon_i = 2$ for i otherwise. Equation (14) can be written equivalently in matrix form as

$$\mathbf{r} = \mathbf{C}^T \mathbf{y}, \tag{15}$$

where the columns of the matrix $\mathbf{C} = [\mathbf{c}_0, ..., \mathbf{c}_P]$ are the cosine basis functions

$$\mathbf{c}_k^T = \epsilon_i \cos\left\{\frac{2\pi(i-1)k}{N}\right\}, \quad i = 1, ..., N/2+1 \tag{16}$$

used to define the noise subspace.

Utilizing the eigen-decomposition of $\mathbf{YY}^H$, we create orthogonal spectrogram $\mathbf{G}$ and noise subspaces using the projection

$$\mathbf{G} = \mathbf{P}_\perp \mathbf{U}\mathbf{\Sigma}\mathbf{U}^H \mathbf{P}_\perp, \tag{17}$$

where

$$\mathbf{P}_\perp = \mathbf{I} - \mathbf{C}(\mathbf{C}^T\mathbf{C})^{-1}\mathbf{C}^T \tag{18}$$

is the projection matrix onto the subspace orthogonal to the noise subspace. An orthonormal basis set for the column space of $\mathbf{G}$ is then computed using the eigen-decomposition $\mathbf{G} = \mathbf{U}_G\mathbf{\Sigma}_G\mathbf{U}_G^H$. The eigenvectors in $\mathbf{U}_G$ corresponding to non-zero eigenvalues are the basis vectors for the spectrogram subspace. We can visually compare the eigenvectors in $\mathbf{U}_G$ with the known spectra of the signal class of interest to select which eigenvectors to use to define the signal subspace. This approach is useful when the training data is corrupted by interferers or events from other signal classes. The spectra of these signals will also appear as basis vectors in $\mathbf{U}_G$ and need to be identified and eliminated. Let those eigenvectors selected to define the signal subspace be collected in order of significance as the columns of the matrix $\mathbf{S} = [\mathbf{s}_1, ..., \mathbf{s}_\rho]$ where ρ is the rank of the signal subspace. From this we can define a matrix of basis vectors for both the signal and noise subspaces as the concatenation $\mathbf{A} = [\mathbf{C}; \mathbf{S}]$. This matrix containing the trained signal subspace will be used by the SP module for feature extraction.

To compute features for a given signal class, the SP module begins by computing the spectrogram of the data in same manner that was used in training the signal subspace as discussed above. That is, the data is segmented into L-sample segments and the spectrogram of each segment is computed using an N-point FFT. The values of L and N used in computing features are the same as those used in training the signal subspace. As before if the SP module is being used as a class-model for an FM signal class, the data must be matched filtered using a replica of the FM signal prior to computing the spectrogram. The spectrogram segments $\mathbf{y}(k)$ are then projected onto the signal and noise subspaces as

$$\mathbf{z}(k) = \mathbf{A}^T \mathbf{y}(k), \tag{19}$$

resulting in a sequence of feature vectors $\mathbf{z}(1), ..., \mathbf{z}(K)$. The J-function for (19) assuming a reference hypothesis of zero-mean Gaussian RVs of unit variance is derived in [16]. Conditioning of the features is then performed to aid in estimating their PDF using GM modeling as follows. The noise subspace projection (ACF) features are first converted to reflection coefficients and then a log-bilinear transformation is applied. The signal subspace projection features are normalized by the magnitude of the projection onto $\mathbf{s}_1$. The J-functions for these conditioning operations are derived in [2]. A HMM is used to statistically model the feature sequence produced by (19).

4 Classification Results using Synthetic Data

To test the performance of the modules, classification experiments were conducted using synthetic data. Synthetic training and testing data for six different classes of waveforms were generated. Of these, three were a type of CW signal and three were a type of FM signal. Using these data sets, experiments were conducted on a general category hybrid discriminative/CS classifier built from trained CST modules and a 6-class CS classifier built from trained SP modules. For comparison, experiments were also conducted on a 6-class hybrid discriminative/probabilistic conventional classifier using a 10-dimensional feature set. The results demonstrate the classification performance improvement for the CS classifiers over the conventional when only a small number of training events are available.

4.1 Synthetic Data Sets

The synthetic signals generated for training and testing the classifiers consisted of six different waveform types which we will describe here. We will designate the signal classes as Ping

1 through Ping 6. Training and testing events generated for all of the waveform types consisted of 110,000 time samples for each event at a sampling frequency of 100 kHz.

Ping 1 is comprised of two simultaneous CW pulses at 31.0 kHz and 34.25 kHz each of 0.013 s duration. A smaller amplitude reverberation component of random duration for each event of between 0.01 s and 0.04 s is appended to both pulses. An example of Ping 1 along with its spectrogram is shown in the left half of Fig. 3. Ping 1 would be considered a short CW signal under the general signal classes.

Ping 2 is a single CW pulse of random frequency 31.25 ± 0.20 kHz for each event of duration 1.0 s. An example of Ping 2 and its spectrogram are shown in the right half of Fig. 3. Ping 2 would be considered a long CW under the general signal classes.

Ping 3 is a HFM signal which sweeps from 35.5 kHz to 34.0 kHz in 0.03 s. The left half of Fig. 4 shows an example of Ping 3 and its spectrogram. Under the general signal classes, Ping 3 would be considered a slow FM signal.

Ping 4 is a HFM signal which sweeps from 38.75 kHz to 33.5 kHz in 0.064 s. An example of Ping 4 along with its spectrogram is shown in the right half of Fig. 4. Ping 4 would be considered a fast FM under the general signal classes.

Ping 5 is a HFM signal which sweeps from 30 kHz to 40 kHz in 0.1 s. The left half of Fig. 5 shows an example of Ping 5 and its spectrogram. Under the general signal classes, Ping 5 would be considered a fast FM signal.

Ping 6 consists of three consecutive CW pulses each of 0.09 s duration and frequencies 30 kHz, 31.75 kHz, and 31 kHz, respectively. A smaller amplitude reverberation component of duration 0.04 s is appended to the last pulse. An example of Ping 6 along with its spectrogram is shown in the right half of Fig. 5. Ping 6 would be considered a medium CW under the general signal classes.

Two independent data sets, one for training and one for testing, using these synthetic signals were generated for the classification experiments. For training, a set of 100 events was generated for each signal class, i.e., Ping number. Each event was generated at a randomly selected signal-to-noise ratio (SNR) uniformly distributed over the range given in Table I.

The ranges of SNR were selected to be in the region where the conventional classifier's performance degraded in classifying the given Ping number. Testing data consisted of 200 events of each Ping number, resulting in a total of 1200 events, all generated at the lowest SNR used in the training data for the given Ping number as given in Table I.

4.2 Conventional Classifier

A 6-class hybrid discriminative/probabilistic conventional classifier was designed using a 10-dimensional feature set. The feature set was based on signal parameters which intuitively should provide good separability of the signal classes. These features are based on the CW characteristics of the signal. The features are called: CW_len_64, CW_f_64, CW_amx_64, CW_amxn_64, CW_len_128, CW_f_128, CW_amx_128, CW_amxn_128, CW_f_256, CW_a_256, and we will explain them presently. They are all computed from the magnitude-squared spectrogram of the signal using an N-point FFT. The features CW_amx_N and CW_amxn_N assume there exists a signal in one FFT bin at one time slice of the spectrogram. The computation of CW_amx_N proceeds by normalizing the spectrogram for each time slice by the total power in that time slice. Next, the largest normalized bin for each time slice is identified with CW_amx_N being the maximum of these values. Computation of CW_amxn_N is done in the same fashion with the exception that the spectrogram at each time slice is normalized by the power in a range of bins surrounding the given bin instead of the total power in the time slice. The features CW_a_N, CW_f_N, and CW_len_N assume that the signal is spread over multiple time slices, but remains in a single FFT bin. Computation of these features proceeds by passing the time sequence of each FFT bin into a bank of integrators of various integration window lengths. Next, the bin with the largest integrator peak is identified with CW_f_N being the frequency of the given bin, CW_a_N is the integrator peak value, and CW_len_N is the integrator's window length.

A conventional Bayesian classifier was used for the generative stage of the hybrid classifier and assumed that all of the classes were equally likely. The PDFs of the features for each of the six signal classes were estimated from the training data sets using GM models. An EM algorithm was used to estimate the GM parameters combined with a heuristic approach to determine the appropriate number of GM modes based on the given training samples. For the discriminative stage, classification rules were constructed for each of the signal classes using the features extracted from the training data. These rules were manually designed by comparing pairs of features in feature space for a given signal class versus the remaining signal classes. When a pair of features is observed which provides reasonable separability between the given signal class and the remaining classes, a rejection boundary can be drawn between the classes. This rejection boundary serves as a classification rule designed to reject events whose computed features lie outside of the boundary for the given signal class. The rules are applied to the data after the evaluation of the feature log-likelihoods across the class models. If an event's log-likelihood is maximum for a given signal class, but it violates the signal class' classification rule, it is assigned to the signal class with the highest log-likelihood for which the class' rule is not violated. If an event violates the rules for all of the signal classes it is assigned to "None" meaning it does not belong to any of the signal classes.

The results of processing the testing data with the 6-class hybrid conventional classifier are shown in Fig. 6 with respect to percentage of correct classification. To observe the impact the classification rules have on performance, results were generated with and without applying them to the data. The bar to the left of the Ping number is the classification result obtained without applying the classification rules (generative classifier only) and the bar to the right is that with applying the classification rules (hybrid classifier). The corresponding confusion matrices for the classification experiment are given in Tables II and III, where the columns are the Ping numbers and the rows are the signal classes. In four out of the six signal classes, the classification performance improved when the rules were applied. The classification performance in processing Ping 1 and Ping 5 decreased with the application of the rules. This is likely due to the small number of training events. Since the training data was generated over a range of SNR and the testing data was generated at the lowest SNR only, it is likely that an insufficient number of lowest SNR events were observed to properly draw the rejection boundary for the classes. This is also the reason why some of the Ping 4 and Ping 5 events were assigned to None with the application of the rules.

4.3 CST Classifier

A general category 5-class hybrid discriminative/CS classifier, for the general signal classes defined previously in Section 3.1, was built from trained CST modules. Using the training data, the CST modules were trained using the procedure discussed in the previous section. The fast FM class-model was trained using the training data for Ping 4 only despite the fact that Ping 5 is also categorized as a fast FM signal. This was done to evaluate the robustness of the CST classifier in properly classifying a waveform type it had not trained on. Classification rules were also constructed for the discriminative stage using statistics derived from the narrow-band tracker as rule-based features. The rule-based features computed for each event were the length of the longest track and the mean velocity, velocity variance, and frequency variance along it. Additionally, log-likelihood ratio thresholds determined from the training data were applied by each class model. The log-likelihood ratio is computed as the difference between the log-likelihood of the event given the reference hypothesis, $\log p(\mathbf{x}|H_{0,i})$ where $H_{0,i}$ is white, zero-mean Gaussian noise of variance 1, and the log-likelihood of each event given the i^{th} class-model $\log p(\mathbf{x}|H_i)$. A relative log-likelihood ratio is also computed as the log-likelihood ratio divided by the event length. For a given class model, if an event does not exceed the threshold for either the log-likelihood ratio or the relative log-likelihood ratio that class model is eliminated from contention. If an event does not exceed the thresholds of any of the class models, it is assigned to None.

The results of processing the testing data with the 5-class hybrid CST classifier are shown in Fig. 7 with respect to percentage of correct classification. As before, the bar to the left of the Ping number is the classification result obtained without applying the classification rules (generative classifier only) and the bar to the right is that with applying the classification rules (hybrid classifier). Tables IV and V show the corresponding results for the classification experiment, where the columns are the Ping numbers (the general class the Ping belongs to is abbreviated in brackets next to the number) and the rows are the signal classes. In contrast to the hybrid conventional classifier, in all cases the classification performance improved, or for Ping 4 remained unchanged, with the application of the rules. Notice that although the fast FM class-model was trained using only Ping 4 training data, it still achieved excellent performance in correctly classifying Ping 5 as a fast FM signal. Overall, the hybrid CST classifier demonstrated a significant performance improvement over the hybrid conventional classifier.

4.4 SP Classifier

A 6-class CS classifier was built using trained SP modules as class-models for each of the Ping types. Training of the SP modules was done following the training procedure given in the previous section. The first step in training the SP module is determining the optimum FFT size N from the training data for a given signal class. Recall that each event is segmented into L-sample segments and an N-point FFT is computed for each segment. For the synthetic data experiments we will assume the segment size and FFT size to be equal $L = N$. The optimization process begins by selecting the signal subspace basis vectors for a given value of N. We will assume $P = 4$ for the noise subspace. Using these parameters in the SP module, features are computed by the module for all of the events in the training data. An estimate of the PDF of the features is computed using a GM model. A total log-likelihood for the given value of N is computed by evaluating the log-likelihoods of the features on the log-PDF across all training events and summing them. This process is repeated over a range of values for N. The value of N producing the maximum log-likelihood is used in the given class model along with the associated signal subspace basis vectors. Features computed from the training data using this value of N are used to train an HMM PDF model for the feature sequence.

The 6-class SP classifier was comprised of trained SP modules for each of the six Ping types. No rules or thresholds were applied by any of the class models. In processing the testing data, the SP classifier achieved perfect classification performance as shown in the resulting confusion matrix given in Table VI. This is due to the fact that each class-model was tailored for each Ping type by including more *a priori* information regarding the spectrum of each Ping type. Given this, the SP classifier should also be robust as a function of SNR.

5 Conclusion

The development of new feature extraction modules for the classification of narrow-band signals using the class-specific (CS) method were presented. One was a general module based on a narrow-band tracker which was designed to classify waveforms into general categories. The others were designed for classifying specific classes of waveforms, where some *a priori* knowledge of the signal is necessary. Results from classification experiments using synthetic data were presented comparing the performance of classifiers built from these modules with that of a conventional classifier using a 10-dimensional feature set. These results demonstrate the increasing performance gains in using the hybrid discriminative/CS general classifier over the hybrid discriminative/probabilistic conventional and that of using the specific CS classifier over the general.

The CS classifiers out-performed the conventional classifier for the given number of training samples. This provides another example that given a small training data set the CS method can provide a vast performance improvement over that of a conventional classifier as was also demonstrated in [2]. The incorporation of classification rules into the generative classifiers to form hybrid classifiers resulted in improved performance for most of the classes with the conventional classifier. However, the CST classifier showed performance improvement for all classes with the incorporation of classification rules in going from a generative to a hybrid classifier. As expected, as more *a priori* information was mapped into the class models, e.g., SP models, the performance improved further. Since it is assumed that not all signals will be known *a priori* in most realistic applications, this suggests that a combination of the general CST classifier and the SP classifier is most appropriate.

6 Acknowledgment

This work was supported by the Office of Naval Research document numbers N0001406WX20206 and N0001406WR20058.

References

[1] C.J. Stone, "Optimal Rates of Convergence for Nonparametric Estimators," *Annals of Statistics*, vol. 8, pp. 1348-1360, (1980).

[2] P.M. Baggenstoss, "Class-Specific Classifier: Avoiding the Curse of Dimensionality," *IEEE A&E Systems Magazine*, vol. 19, no. 1, pp. 37-52, (2004).

[3] T. Jebara, *Machine Learning: Discriminative and Generative.* Kluwer, 2003.

[4] Y. D. Rubinstein and T. Hastie, "Discriminative vs Informative Learning," in *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining*, Newport Beach, CA, pp. 49–53, 1997.

[5] C. Goutte, E. Gaussier, N. Cancedda, and H. Dejean, "Generative vs Discriminative Approaches to Entity Recognition from Label-Deficient Data," *Proc. of JADT 2004 (Journees internationales d'Analyse statistique des Donnees Textuelles)*, Louvain La Neuve, Belgium, 2004.

[6] B. Schoelkopf, C. Burges, and V. N. Vapnik, "Extracting Support Data for a Given Task," in *Proc. 1st Int. Conf. Knowledge Discovery Data Mining* (U.M.Fayyad and R. Uthurusamy, eds.), (Menlo Park, CA), AAAI Press, 1995.

[7] K. Mueller, S. Mika, G. Raetsch, K. Tsuda, and B. Schoelkopf, "An Introduction to Kernel-Based Learning Algorithms," *IEEE Trans. Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001.

[8] A. Ng and M. Jordan, "On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes," *Advances in Neural Information Processing Systems 14.* MIT Press, 2002.

[9] S. Fine, J. Navratil, and R. Gopinath, "Enhancing GMM Scores Using SVM Hints," in *Proceedings of the 7th European Conference on Speech Communication and Technology (EuroSpeech)*, 2001.

[10] T. Jaakkola and D. Haussler, "Exploiting Generative Models in Discriminative Classifiers," Tech. report, Dept. of Computer Science, Univ. of California, 1998.

[11] L. Quan and S. Bengio, "Hybrid Generative-Discriminative Models for Speech and Speaker Recognition," Tech. report IDIAP-RR 02-06, Institute for Perceptual Artificial Intelligence, Martigny, Switzerland 2002.

[12] R. Raina, Y. Shen, A. Ng, and A. McCallum, "Classification with Hybrid Generative /Discriminative Models," *Proc. of NIPS (Neural Information Processing Systems)*, 2004.

[13] P.M. Baggenstoss, "The PDF Projection Theorem and the Class-Specific Method," *IEEE Trans. on Signal Processing*, vol. 51, no. 3, pp. 672-685, (2003).

[14] T.W. Anderson, *The Statistical Analysis of Time Series.* Wiley, 1971.

[15] B.F. Harrison, "Development of a Class-Specific Module for Hyperbolic, Frequency-Modulated Signals," Tech. report TR 11,681, Naval Undersea Warfare Center, Newport, RI, June 2005.

[16] S.M. Kay, A.E. Nuttall, and P.M. Baggenstoss, "Multidimensional Probability Density Function Approximations for Detection, Classification, and Model Order Selection," *IEEE Trans. Signal Process.*, vol. 49, no. 10, pp. 2240-2252, (2001).

Ping	Train SNR	Test SNR
1	[0, 6] dB	0 dB
2	[-10, -6] dB	-10 dB
3	[-10, -8] dB	-10 dB
4	[-10, -6] dB	-10 dB
5	[-10, -6] dB	-10 dB
6	[-6, 0] dB	-6 dB

Table I: Range of SNR used for generating training and testing data.

		Ping					
		1	2	3	4	5	6
Class	1	52	0	4	13	8	49
	2	0	0	0	0	0	0
	3	0	0	144	27	9	0
	4	87	0	6	58	28	48
	5	60	0	46	102	155	3
	6	1	200	0	0	0	100
	None	0	0	0	0	0	0

Table II: Confusion matrix for conventional classifier without applying classification rules.

		Ping					
		1	2	3	4	5	6
	1	34	0	2	2	3	3
	2	7	200	0	0	0	0
	3	0	0	146	21	4	0
Class	4	50	0	47	152	29	0
	5	108	0	5	4	130	24
	6	1	0	0	0	0	173
	None	0	0	0	21	34	0

Table III: Confusion matrix for conventional classifier with classification rules applied.

		Ping					
		1(sCW)	2(lCW)	3(sFM)	4(fFM)	5(fFM)	6(mCW)
	short CW	94	0	15	0	0	4
	long CW	0	164	0	0	0	3
Class	slow FM	101	0	166	9	11	2
	fast FM	2	0	18	191	189	0
	medium CW	0	36	0	0	0	191
	None	3	0	1	0	0	0

Table IV: Classification results for CST classifier without applying classification rules.

		Ping					
		$1_{\text{(sCW)}}$	$2_{\text{(lCW)}}$	$3_{\text{(sFM)}}$	$4_{\text{(fFM)}}$	$5_{\text{(fFM)}}$	$6_{\text{(mCW)}}$
Class	short CW	192	0	16	3	1	6
	long CW	0	200	0	0	0	0
	slow FM	3	0	183	6	2	0
	fast FM	0	0	0	191	197	0
	medium CW	0	0	0	0	0	194
	None	5	0	1	0	0	0

Table V: Classification results for CST classifier with classification rules applied.

		Ping					
		1	2	3	4	5	6
Class	1	200	0	0	0	0	0
	2	0	200	0	0	0	0
	3	0	0	200	0	0	0
	4	0	0	0	200	0	0
	5	0	0	0	0	200	0
	6	0	0	0	0	0	200

Table VI: Confusion matrix for SP classifier.

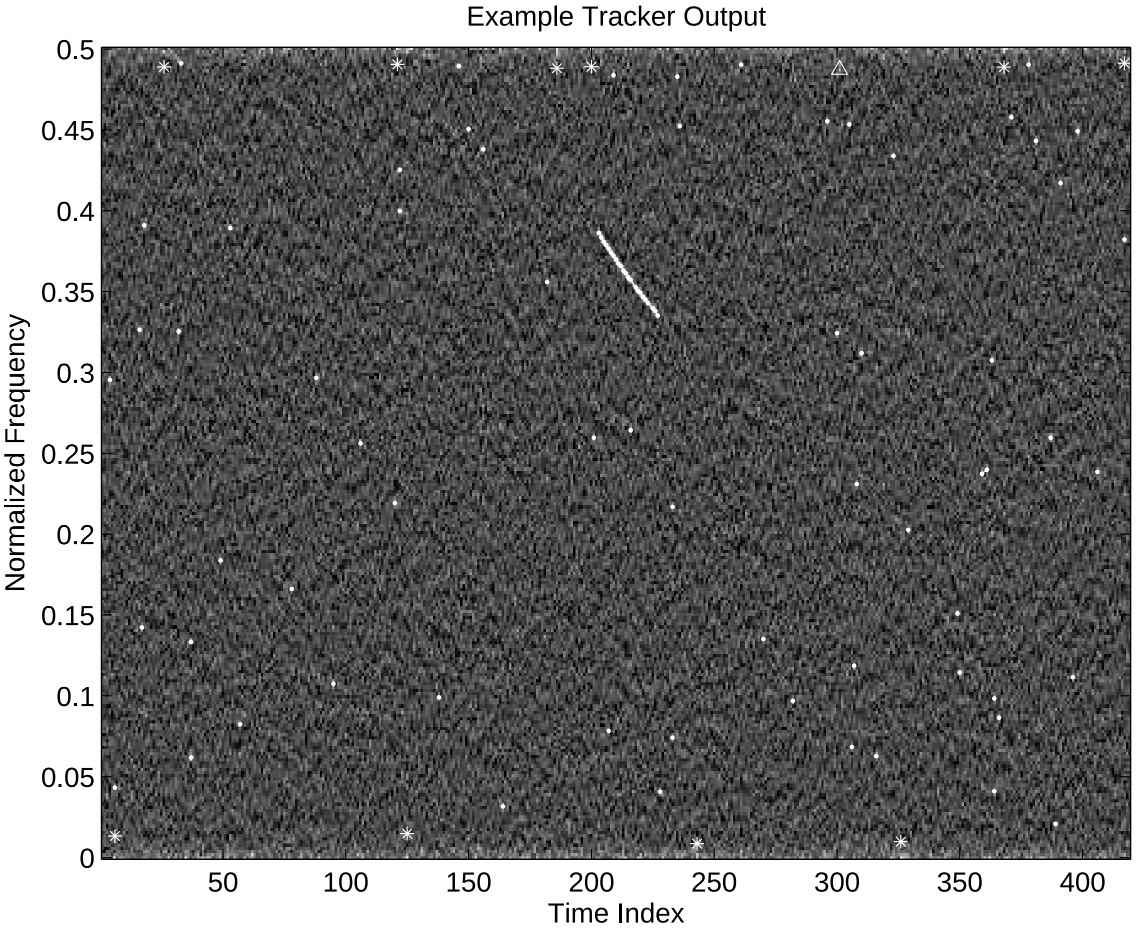

Figure 1: Example output surface from narrow-band tracker for a hyperbolic FM signal.

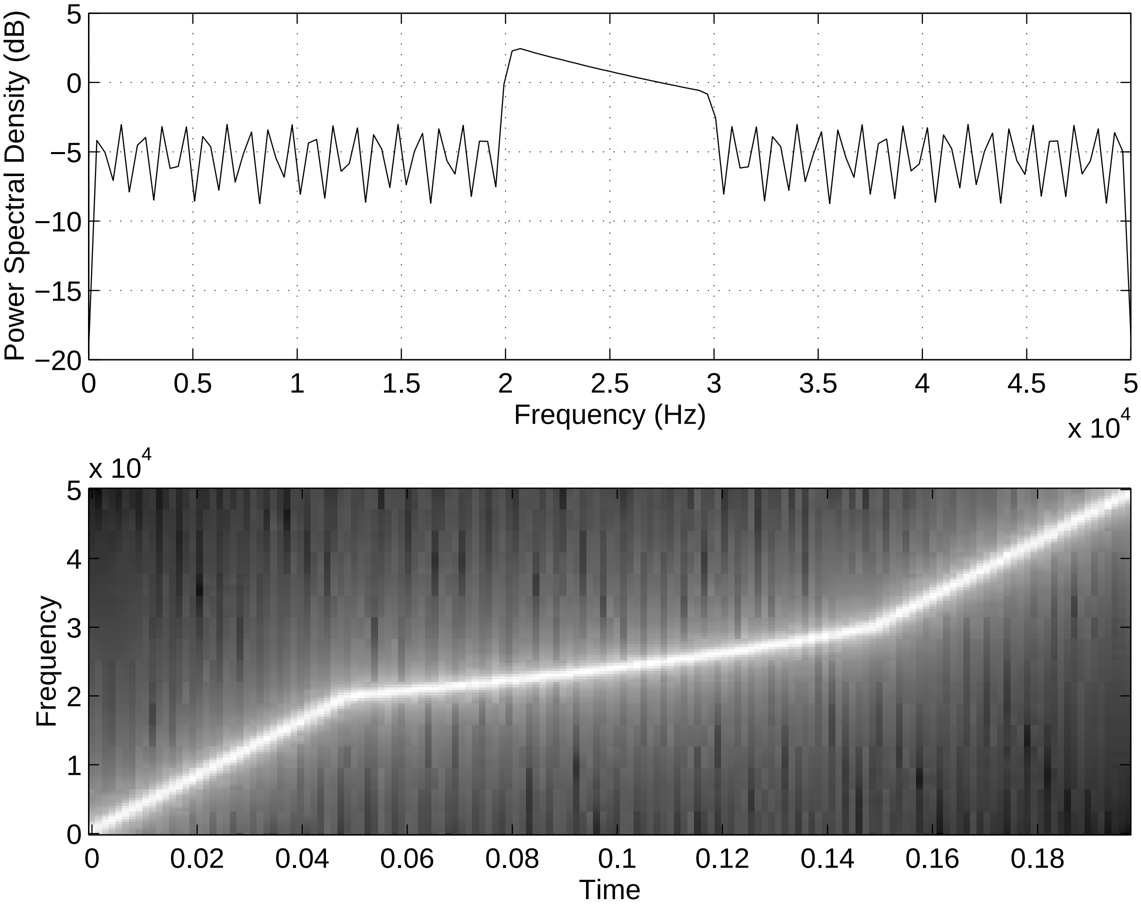

Figure 2: Power spectral density (top) and spectrogram (bottom) of example HFM replica used for matched filtering.

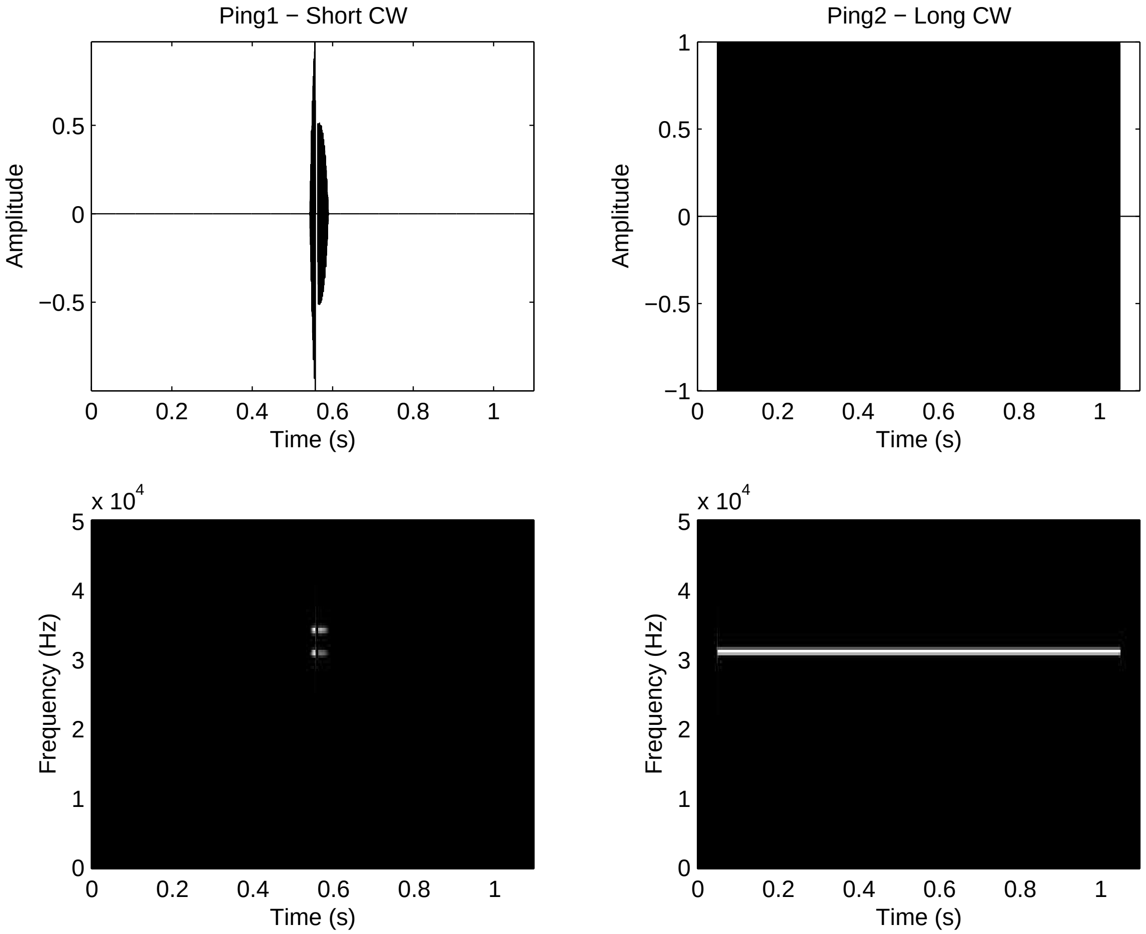

Figure 3: Time series and spectrograms of signal classes Ping 1 (left) and Ping 2 (right).

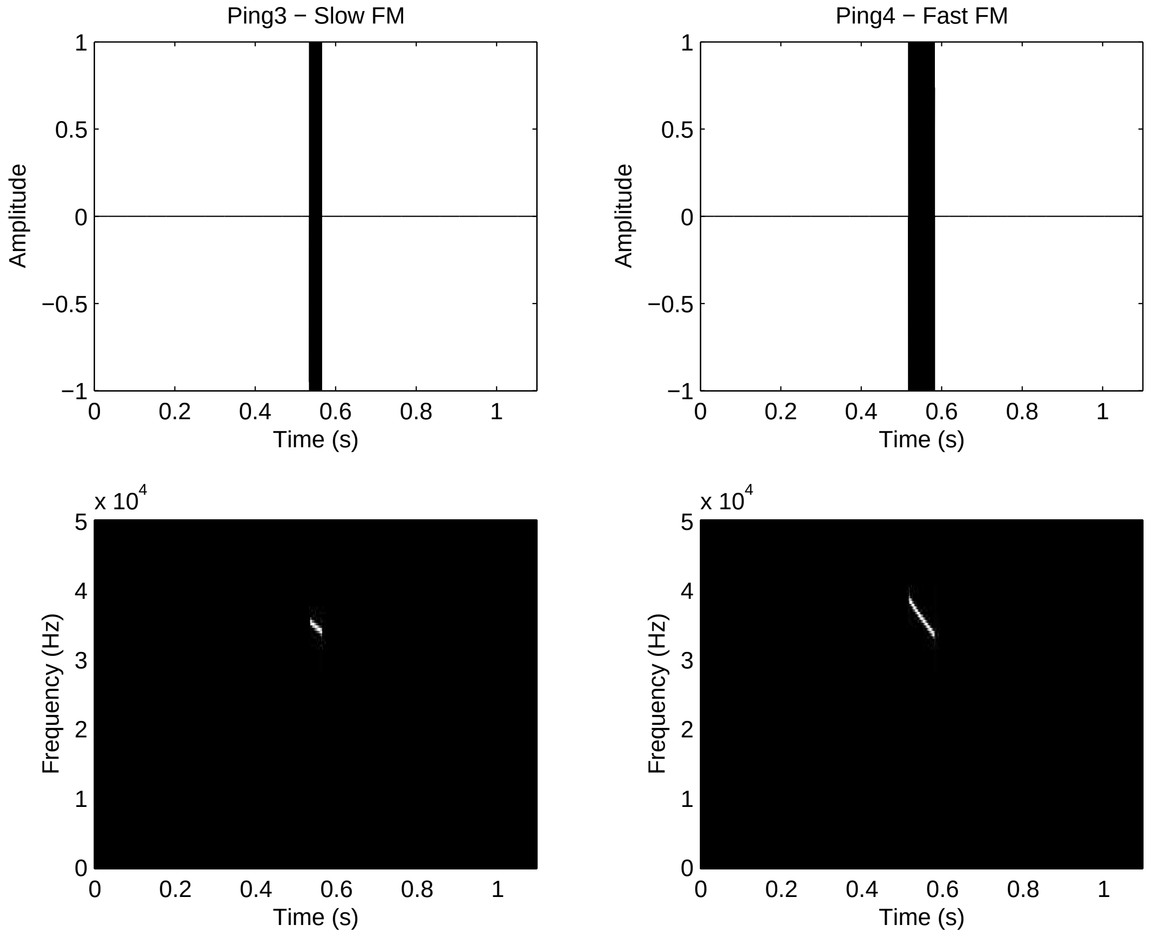

Figure 4: Time series and spectrograms of signal classes Ping 3 (left) and Ping 4 (right).

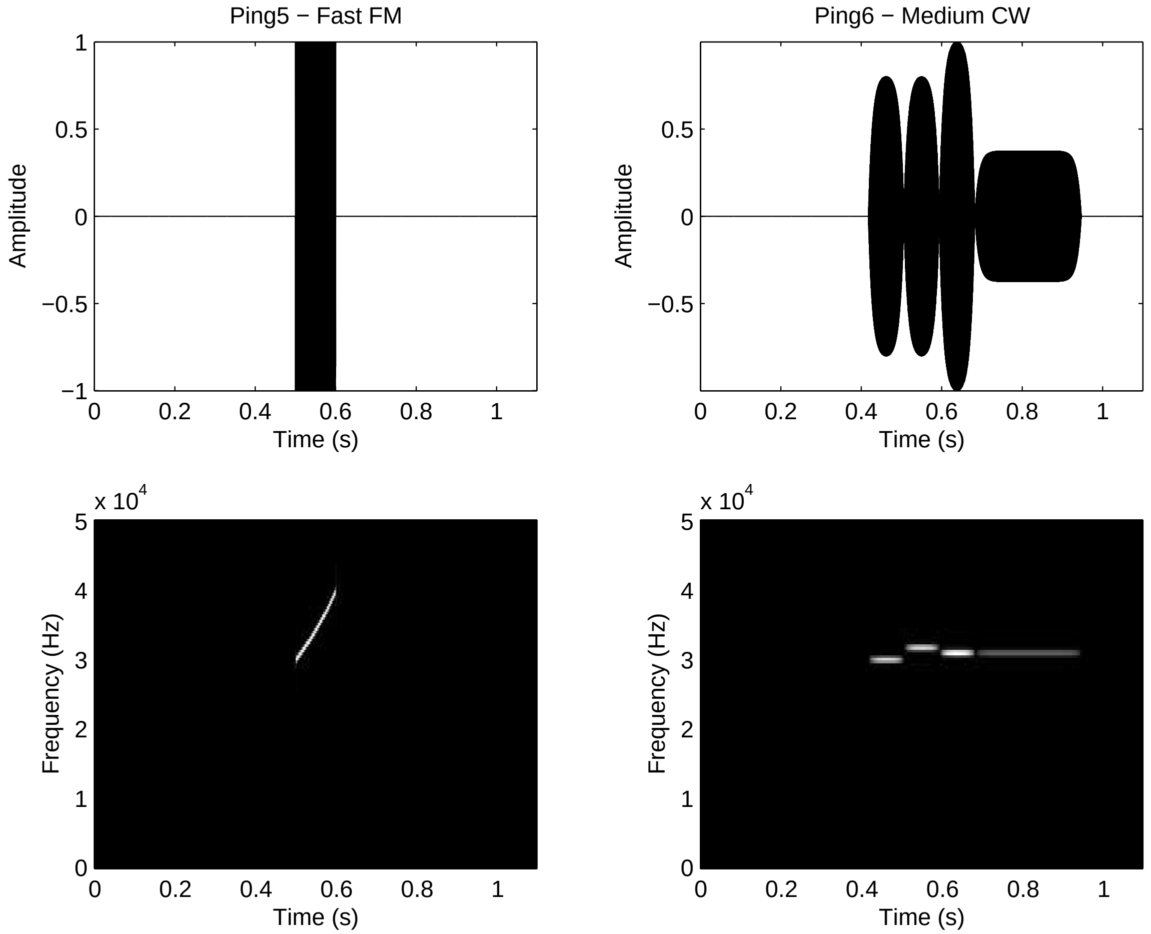

Figure 5: Time series and spectrograms of signal classes Ping 5 (left) and Ping 6 (right).

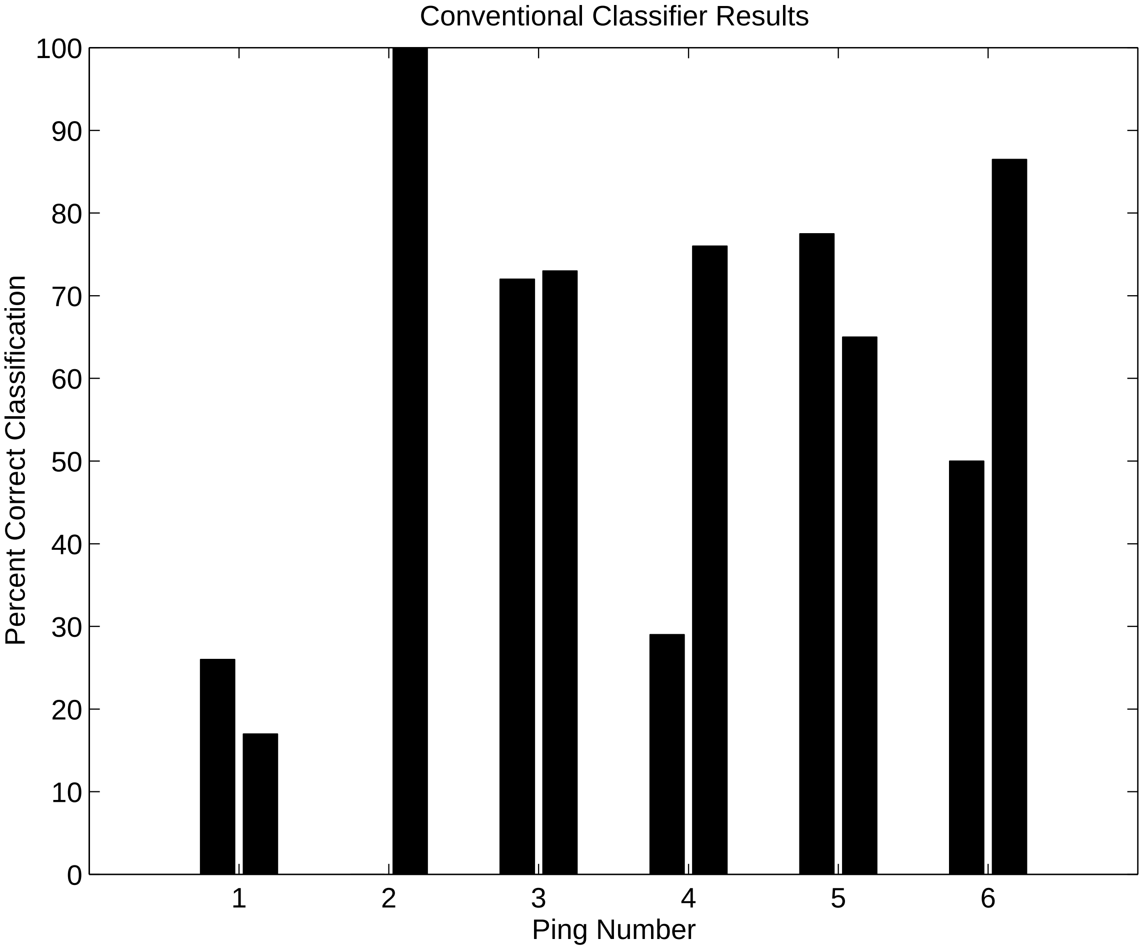

Figure 6: Percent correct classification for conventional classifier. Bar to the left of Ping number is performance without application of rules (generative classifier) and bar to right is that with the application of rules (hybrid classifier).

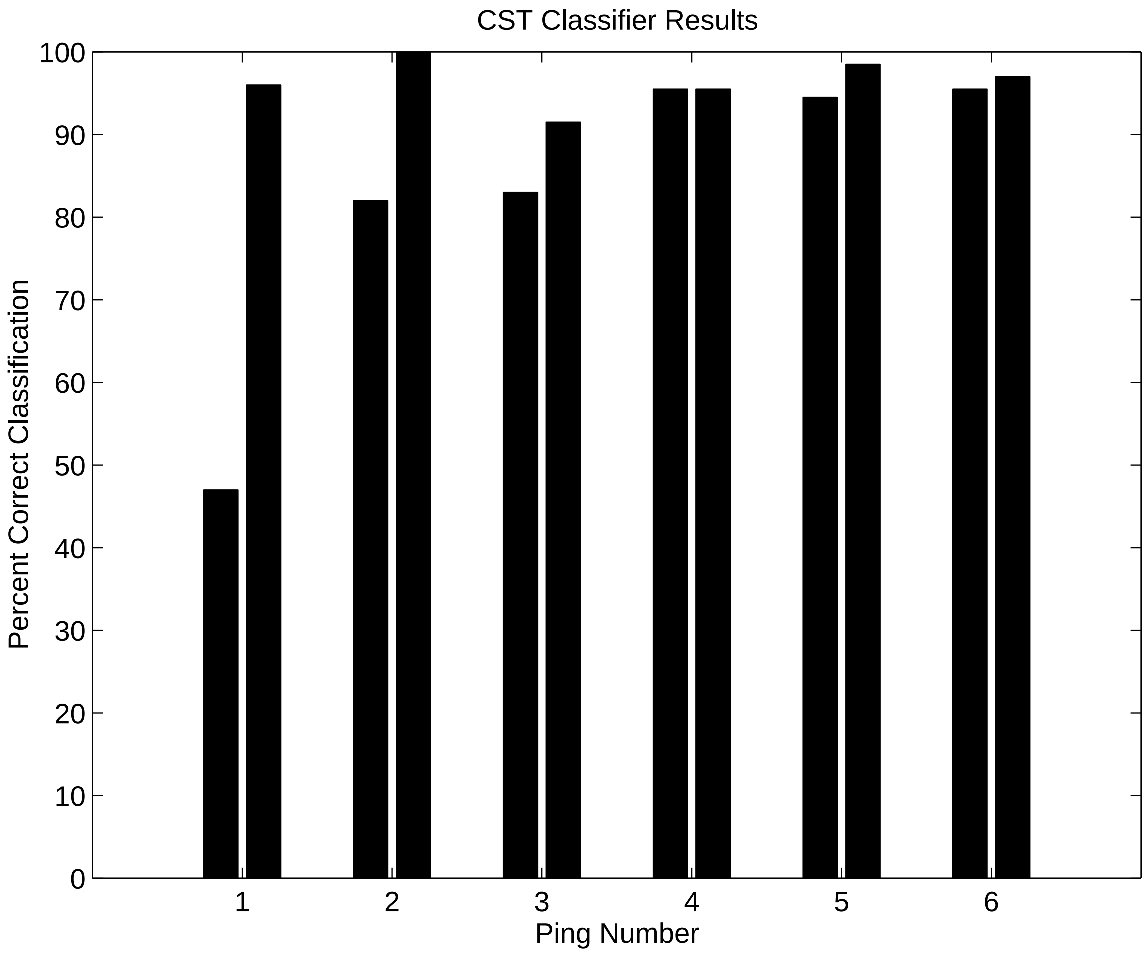

Figure 7: Percent correct classification for CST classifier. Bar to the left of Ping number is performance without application of rules (generative classifier) and bar to right is that with the application of rules (hybrid classifier).