A Wavelet-Based Method of Nearest Neighbor Pattern Classification Using Scale Sequential Matching

(Unclassified, distribution unlimited)

Charles D. Creusere Gary Hewer

Naval Air Warfare Center Weapons Division
China Lake, CA 93555

Abstract

In this method of pattern classification, a wavelet transform is used to extract features from the input signal which are then compared in a scale sequential manner (from coarse to fine) to a trained nearest neighbor codebook. At each scale, possible classification categories are eliminated until only one class is left. We apply this pattern classifier to the problem of pulse fingerprinting and analyze its performance in noise using Monte Carlo simulations. To make our classifier shift invariant, we process the input with an undecimated wavelet transform until the pulse edge is sensed and then start decimating the wavelet coefficients as appropriate to each scale.

1 Introduction

The classifier proposed in this paper is based on the multiresolutional or wavelet decomposition of a signal [1]-[2] and was partly inspired by work done by Devaney and Hisconmez in [3]. Their paper proposed a scale sequential approach to the problem of classification based on M-ary hypothesis testing and using log likelihood ratios at different scales. Because of the statistical nature of their method, the only results they presented were for highly artificial targets consisting of uniformly distributed random sequences. Our method, on the other hand, also uses a scale sequential matching approach but is based on nearest neighbor classifiers [4] which can be designed directly from training data a priori or can be created on-the-fly using clustering techniques. In addition, our hierarchical thresholding and searching methodology is considerably different than that proposed in [3]. Finally, the method proposed in our paper can also be used as a vector quantizer for signal compression and decompression [5] since an approximation of the original signal can be reconstructed by taking the inverse wavelet transform of the appropriate codebook vector.

2 Feature Extraction

Feature vectors are extracted from the input signal using a wavelet decomposition constructed from 2-band QMF banks. Such filter banks are both approximately orthogonal and linear phase; the first property makes it easier to determine how signal energy is partitioned by the transform while the latter simplifies the process of relating temporal events occurring at different scales. Relating such events would be far more difficult using, for example, a Daubechies orthogonal wavelet [6] because these have highly nonlinear phase characteristics. For the results presented in this paper, we have used a long 64-tap QMF bank (64D in [7]) in order to get good frequency discrimination at the different scales. While this filter may be longer than necessary, our preliminary results do indicate that good frequency discrimination is important in achieving reliable classification at lower signal to noise ratios.

Initially, all input waveforms are gain normalized since the method proposed here only tries to match shapes. If the gain of the signal also contains information useful to the classification process, it can be examined separately and used to eliminate classes before starting our procedure. To accomplish the actual feature extraction, the input signal is fed through an undecimated wavelet transform and the coarsest scale (lowest frequency band) is monitored. When the signal amplitude in this band reaches half of full scale, the algorithm concludes that it has found the leading edge of the pulse. Since the area in front of the leading edge also contains useful feature information, collection of the decimated wavelet coefficients begins a fixed number of samples in front of the identified leading edge. Samples are then collected long enough to guarantee complete coverage of the longest possible pulse. These sets of decimated wavelet coefficients thus form the feature vectors used by the scale sequential nearest neighbor classifier. A diagram of the entire process is shown in Fig. 1.

It is worthwhile here to highlight some important points. First, the process of finding the pulse edge is fairly robust with respect to noise because the decision is made at the coarsest scale. Second, the fact that linear phase filters are used allows us to precisely determine where the extraction of the feature vectors at the other scales should start (given the pulse edge determined at the coarsest scale) because each filtering stage imparts a fixed delay to its output samples. Finally, the use of approximately orthonormal filters allows for easier treatment of noise effects, especially white, Gaussian noise.

3 Scale Sequential Nearest Neighbor Classification

To design the classifier codebook, we first extract features from labeled training set data using the wavelet extraction technique described in the previous section. The codebook vectors for each class and at every scale are selected as the centroids of the appropriate subsets of these labeled features. In a conventional nearest neighbor classifier, the classification decision is based exclusively on the distance between the input feature vector and the codebook vectors: the class whose codebook vector is closest (usually in terms of Euclidean distance) to the input vector is selected. This is not necessarily the case with the proposed scale sequential classification method.

With scale sequential classification, the decision on how to classify an input vector is made in stages with a certain number of possibilities being eliminated at each stage (see Fig. 2). The process starts with the coarsest (lowest frequency) scale where the signal to noise ratio (SNR) is the highest and works toward the finest resolution (highest frequency) scale where the SNR is the lowest. At the coarsest scale, the codebook vector which best matches the input vector is found. The classes corresponding to that vector and to all other codebook vectors whose Euclidean distances are within T_0 of that vector are retained for consideration; all other classes are eliminated from future consideration. This process continues with each increasingly fine scale where the best match at that scale is chosen of the remaining contenders, and it terminates when only one choice is left.

The scale sequential nearest neighbor algorithm can be quantified as follows. Let c_i identify the i-th class, corresponding to the codebook vector $\mathbf{v}_i^j$ at scale j and define the sets C_j as

$$C_j \equiv \{c_i | \text{scale} = j\}, \tag{1}$$

i.e., the set of all classes still in contention at scale j. Starting from j = 0 and with C_0 initialized such that

$$\forall c_i \in C_0, \tag{2}$$

the scale sequential matching is accomplished by successively computing

$$\varepsilon_{opt}^j = \left\| \mathbf{v}_{opt}^j - \mathbf{v}_{in}^j \right\|^2 \le \varepsilon_i^j, \quad \forall i \quad \text{s.t.} \quad c_i \in C_j \tag{3}$$

and

$$C_{j+1} = \left\{ c_i \middle| (c_i \in C_j \cap \left| \varepsilon_i^j - \varepsilon_{opt}^j \right| < T_j) \right\}, \tag{4}$$

incrementing j until the set defined by Eq. (4) contains only one element. That final element identifies the selected classification for the input feature vector. The basic procedure outlined above is illustrated in Fig. 2.

Equations (3) and (4) warrant some explanation. The quantity ε_{opt}^j in Eq. (3) is simply the Euclidean distance between the input vector at scale j and the closest codebook vector still in consideration at that scale, as indicated by the inequality. This minimal Euclidean distance is then used in Eq. (4) along with a threshold T_j to eliminate any class possibilities which are far from the "best" choice. The process is repeated until the final class set has only one element.

The threshold T_j used in Eq. (4) is based on the variance of the codebook at scale j with adjustments for the change in the SNR from scale to scale. The SNR decreases as the scale increases because the signal power decreases exponentially while noise power stays constant. For the example illustrated in the next section, we have found that the signal power decreases by a factor proportional to 2^{2j} as the scale number j increases. Consequently, to have regions of equivalent uncertainty around the best match in all of the scales, we must increase the threshold by this exponential factor. This is not to say that the actual threshold will go up, however, since it is also a function of the variance which is constantly decreasing. Finally, we contract the threshold by a powers of 2 with increasing scale number so that the uncertainty interval, already normalize by the variance and the SNR decrease, will be halved at each step. This forces the algorithm to converge toward a single class-- it is still remotely possible that multiple classes will be selected, however. Putting these terms together, our threshold at scale j can be written as

$$T_j = \alpha \cdot 2^{j-1} \cdot \text{var}_j \tag{5}$$

where var_j is the variance of the codebook at scale j, α is a constant, and the threshold contraction has been combined with the exponential increase caused by the decreasing signal strength.

At this point, it is worth mentioning some of the advantages of the scale sequential nearest neighbor algorithm. First, it has a much lower search complexity than a conventional nearest neighbor classifier. Not only are a larger number of possible classes eliminated at each step in the process but also the complexity of the comparison (a vector inner product) increases with increasing scale. This means that at the coarse scales, which require the most comparisons, the complexity of each comparison is lowest while at the fine scales where few comparisons are made, it is higher (although still well below that of the standard nearest neighbor search). Another strength of this algorithm is its ability to handle noise. It is superior to more conventional techniques because it takes advantage of differences in SNR at the different scales by using only the finer scale information which is consistent with that derived from the coarser scales. Thus, unless a codebook vector matches reasonably well at the coarser scales, its class can not be selected based on evaluation of the finer scales. Increasing the threshold to account for signal at-

tenuation at finer scales allows the window to widen somewhat with decreasing scale to reduce the possibility of inadvertent eliminating an acceptable class candidate.

4 Example: Pulse Fingerprinting

To illustrate this approach to pattern classification, we apply it to the problem of fingerprinting individual post-detection radar pulses. The objective here is to sense the onset of a pulse and to then analyze that pulse in an attempt to determine what type of radar generated it. For this example, we have designed a scale sequential nearest neighbor classifier from training data which has a total of 20 classes. Figure 3 shows the feature vectors for an input pulse with ((a), (b)) and without ((c), (d)) additive noise versus the correct codebook vectors (the circles) for the two coarsest scales. Using a rough optimization procedure, we have determined that a value of $\alpha = 0.1$ provides good results for most of the classes. For the Monte Carlo simulations listed in Table 1, we performed 100 trials of the experiment, using randomly generated additive white, Gaussian noise and selecting input vectors from within the training set randomly as well. Listed along the top of the table are a variety of signal to noise ratios (in decibels) and the different classes are listed vertically; the numbers in the interior are the percentages of correct classification with the accuracy computed to 90% [8]. How well this classifier performed varied a great deal from class to class--some classes are very similar to each other. One class in particular, class 7, performed very poorly even with no noise added to the input pulse (SNR = ∞). Examining the pulses in this class's training set, we have noted that they are very dissimilar, explaining the classifier's poor performance. In fact, if a clustering algorithm had been used to partition the training sets, these vectors would certainly not have been grouped together. For this classifier, however, the actual clustering was done based on a priori knowledge of how the pulses were generated, leading to large differences in classification performance amongst the various classes.

For comparison, we have considered two other low complexity pulse classification schemes, both of them using the nearest neighbor concept. The first is the direct application of a nearest neighbor classifier, where the leading edge of the input pulse is located and that pulse is compared to shift-normalized codebook vectors (which are just pulses themselves). In the absence of noise this method performs very well (95 % correct classification in class 1), but its performance degrades rapidly when noise is added (at SNR < 30dB, 0% correct classification in class 1). Comparing this result with Table 1, we see that the scale sequential classifier degrades much more gracefully with increasing noise level, still correctly classifying pulses in class 1 66% of the time even at an SNR of only 5 dB.

The other scheme we have considered here uses a frequency domain pulse classification method. Again, a codebook is created but this time in the frequency domain. The correlation is then done through frequency domain multiplication with the various codebook vectors followed by an inverse Fourier Transform. This type of classifier, as it turns out, fails almost every time. In retrospect, this is not surprising since discrimination between pulses classes relies largely on small time transients around pulse edges which are entirely lost in the frequency domain

5 Conclusions and Future Directions

We have presented here a new algorithm for doing pattern classification which implements nearest neighbor classification in a multiresolutional hierarchy to achieve excellent performance at reduced computational complexity. To demonstrate its utility, we have applied this algorithm to the problem of single pulse identification and shown that it achieves far superior results to other low complexity classifiers.

Much work remains to be done in this area, however. The complexity of filter banks used to implement the wavelet transform can probably be reduced by using either shorter QMF banks or linear phase biorthogonal banks [9]. In addition, the use of pre-detection and/or wider bandwidth radar pulses could greatly improve classification performance. Finally, it would be worthwhile to build an adaptive clustering algorithm into the classifier to allow the system to better handle new signals generated by source outside the training set (or to do away with training entirely). In short, we believe that the classification algorithm presented here is simple and effective, providing many potential avenues for growth and future use.

References

[1] S. Mallat, "Multiresolutional approximations and wavelet orthonormal bases of $L^2(R)$," *Trans. American Math Soc.*, Vol. 315, 1989, pp. 69-87.

[2] S. Mallat, "Theory for multiresolutional signal decomposition: the wavelet representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 2, 1989, pp. 674-693.

[3] A.J. Devaney and B. Hisconmez, "Wavelet signal processing for radar target identification a scale sequential approach," *Proc. SPIE Vol. 2242 Wavelet Applications*, April 1994, pp. 389-399.

[4] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, New York, 1973, pp. 97-105.

[5] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, Boston, MA, 1992, pp. 307-401.

[6] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Comm. on Pure and Applied Mathematics*, Vol. XLI, 1988, pp. 909-996.

[7] J.D. Johnston, "A filter family designed for use in quadrature mirror filter banks," *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, April 1980, pp. 291-294.

[8] E. Kreyszig, *Introductory Mathematical Statistics, Principles and Methods*, John Wiley & Sons, New York, 1970, pp. 188-190.

[9] I. Daubechies, *Ten Lectures on Wavelets*, SIAM, Philadelphia, PA, 1992, pp. 251-288.

Table 1: Monte Carlo Results.

Class	5	10	20	30	40	∞
1	66±8	79±7	89±5	98±2	96±3	96±3
2	21±7	35±8	57±8	75±7	83±6	87±6
3	7±4	17±6	31±8	59±8	92±4	100
4	23±7	27±7	45±8	69±8	76±7	76±7
5	22±7	46±8	59±8	78±7	93±4	100
6	52±8	66±8	75±7	85±6	91±5	90±5
7	37±8	32±8	41±8	49±8	51±8	50±8
8	25±7	35±8	50±8	81±6	93±4	100
9	76±7	80±7	90±5	88±5	87±6	85±6
10	12±5	25±7	34±8	51±8	70±8	100
11	67±8	69±8	94±4	94±4	98±2	100
12	18±6	22±7	41±8	73±7	85±6	90±5
13	14±6	12±5	35±8	67±8	86±6	100
14	62±8	81±6	96±3	100	100	100
15	21±7	34±8	36±8	59±8	81±6	100
16	39±8	43±8	61±8	72±7	82±6	100
17	34±8	33±8	56±8	68±8	82±6	100
18	86±6	89±5	94±4	96±3	96±3	100
19	46±8	51±8	76±7	94±4	100	100
20	49±8	56±8	48±8	49±8	68±8	69±8

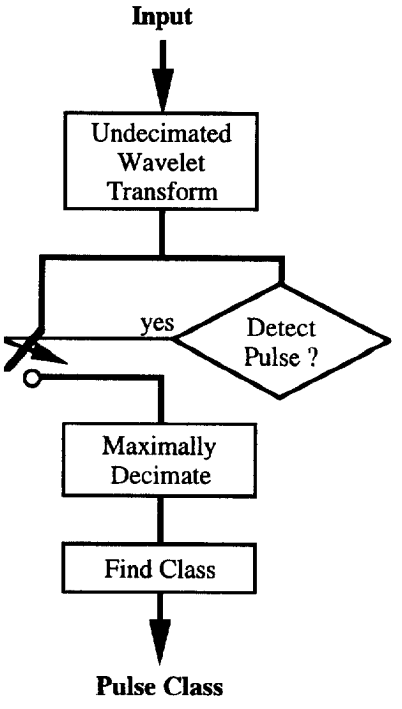

Figure 1: Wavelet-based feature extraction algorithm. The block labeled 'Find Class' represents the scale sequential classification algorithm.

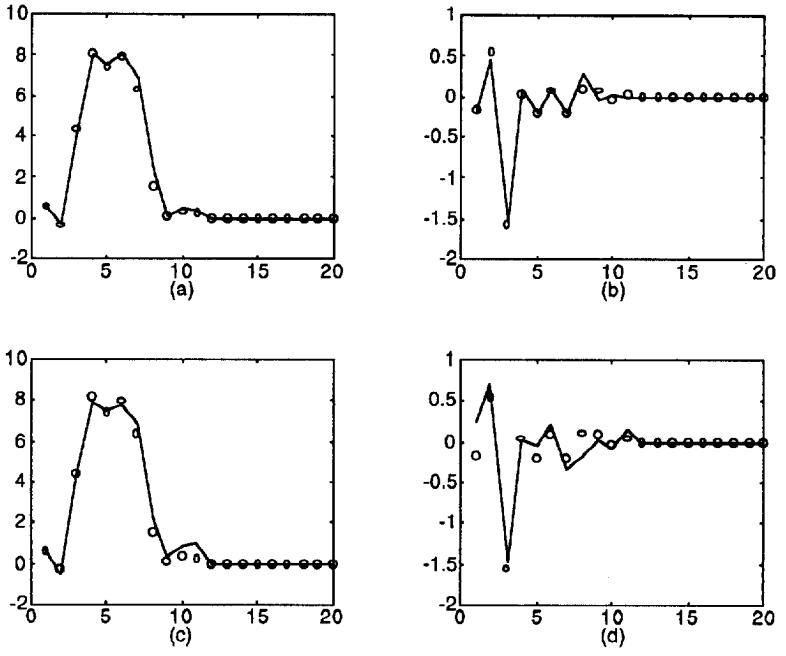

Figure 3: Feature vectors for scales one and two with the circles indicating the correct codebook vectors. (c) and (d) contain additive white noise (SNR = 20 dB).

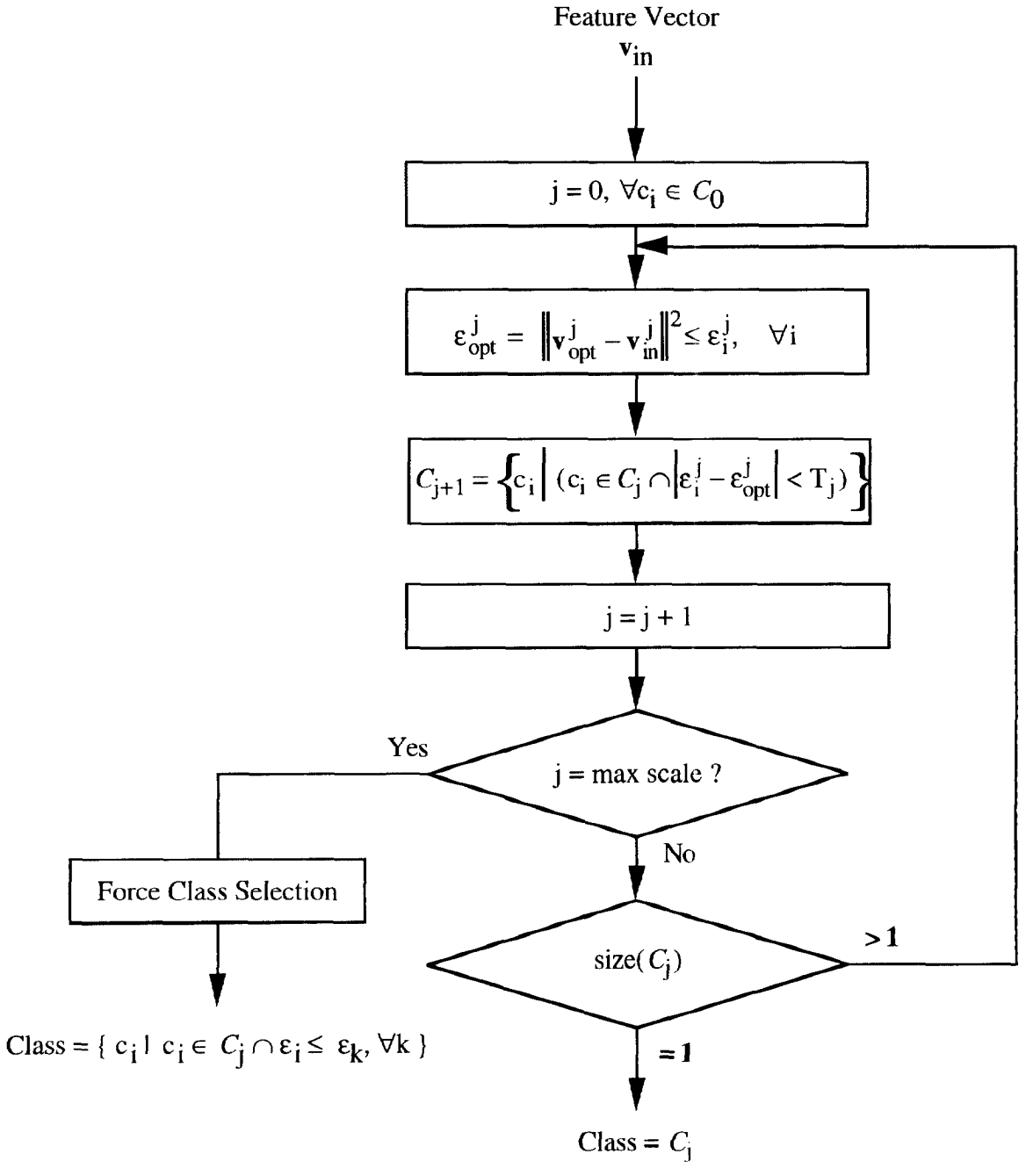

Figure 2: The scale sequential nearest neighbor classification algorithm. Codebook and input feature vectors are labeled **v** with the superscript denoting scale and the subscript specifying the source (e.g., codebook class or input vector).