EXPLOITING HIGH DIMENSIONAL DATA FOR SIGNAL CHARACTERIZATION AND CLASSIFICATION IN FEATURE SPACE

by

Mary L. Cassabaum

A Dissertation Submitted to the Faculty of the DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING In Partial Fulfillment of the Requirements For the Degree of DOCTOR OF PHILOSOPHY In the Graduate College THE UNIVERSITY OF ARIZONA

2004
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI Microform 3145051
Copyright 2004 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346
As members of the Final Examination Committee, we certify that we have read the
dissertation prepared by Mary Lou Cassabaum
entitled Exploiting High Dimensional Data For Signal Characterization
and Classification In Feature Space

and recommend that it be accepted as fulfilling the dissertation requirement for the
Degree of Doctor Of Philosophy

Jeffrey J. Rodriguez, Ph.D. 8-9-04
Robert Schowengerdt, Ph.D. 8/9/04
Robin N. Strickland, Ph.D. 8-9-04

Final approval and acceptance of this dissertation is contingent upon the
candidate's submission of the final copies of the dissertation to the Graduate College.
I hereby certify that I have read this dissertation prepared under my direction and
recommend that it be accepted as fulfilling the dissertation requirement.

Dissertation Director: Jeffrey J. Rodriguez, Ph.D. 8-9-04
STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under the rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of the source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department of the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: [Signature]
ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Jeffrey Rodriguez, for his support, encouragement and suggestions throughout this project. Many thanks to Dr. Robert Schowengerdt and Dr. Robin Strickland for participation on my defense committee and for their valuable feedback.

Many of my colleagues at work gave me encouragement throughout the graduate school process. Thanks are due Harry Schmitt for initially encouraging me to continue my education. Don Waagen provided suggestions for research topics, as well as many useful suggestions, discussions and technical guidance. Don Hulsey’s wealth of knowledge was very much appreciated. I am fortunate to work with a team of people whose contribution to my knowledge base and experience is too great to mention individually, but who are valued.
Dedicated to my husband, Bill, whose love, encouragement, patience and support made continuing my education possible. I would like to thank my children, Carri Ann, Michael, Tyler and Christina whose tolerance and support of a mom in school for years was evident. Finally, I would like to thank my parents, Paul and Marian Nelson, who instilled in me a desire for knowledge, a love of learning, and the confidence to go after my dreams.
TABLE OF CONTENTS

LIST OF FIGURES .................................................................................................................. 8
LIST OF TABLES .................................................................................................................... 11
ABSTRACT ............................................................................................................................ 13

CHAPTER 1 INTRODUCTION .................................................................................................. 14
  1.1 Signal Representation in Feature Space ................................................................. 15
  1.2 Selection of Classifiers and Required Input Hyperparameters ....................... 21
  1.3 Scope of Dissertation ............................................................................................. 23

CHAPTER 2 TRADE STUDY EXPERIMENTAL DATA ......................................................... 25
  2.1 Polarization Definitions ......................................................................................... 26
  2.2 Data ......................................................................................................................... 28

CHAPTER 3 CLASSIFIERS .................................................................................................. 31
  3.1 Bayesian Classifier ................................................................................................ 32
  3.2 Support Vector Machines ..................................................................................... 34

CHAPTER 4 A HEURISTIC FEATURE APPROACH .......................................................... 42
  4.1 Standard Definitions for Feature Calculations ..................................................... 44
     4.1.1 Target Extent ................................................................................................ 45
     4.1.2 Average Clutter ............................................................................................ 47
  4.2 Features .................................................................................................................... 47
  4.3 Analysis .................................................................................................................... 55

CHAPTER 5 WAVELET BASIS TRANSFORMATION .......................................................... 65
TABLE OF CONTENTS - continued

5.1 Wavelet-Based Approach to Projection to New Basis .......................... 66
5.2 Wavelet Transforms ........................................................................ 68

CHAPTER 6 BASIS SELECTION ................................................................. 77
6.1 Local Discriminant Basis .................................................................. 79
6.2 Kolmogorov-Smirnov Best-Basis Selection ...................................... 85
6.3 Best-Basis Algorithm Example ....................................................... 94
6.4 Kullback-Leibler Empirical Density Best Basis ................................. 105

CHAPTER 7 SVM HYPERPARAMETER OPTIMIZATION .......................... 106
7.1 Hyperparameter Selection Via Margin Distributions ....................... 107
7.2 Hyperparameter Selection Via Kernel Semi-Alignment .................... 111
7.3 SVM Hyperparameter Optimization Trade Study ......................... 119
7.4 Conclusion ..................................................................................... 130

CHAPTER 8 WAVELET BASIS RESULTS ............................................. 134
8.1 Procedure ...................................................................................... 134
8.2 Comparison of LDB and KS .......................................................... 136
8.3 Comparison of LDB, KS and KL with Reduced Sample Support ....... 139
8.4 Selection of SVM Kernel Hyperparameters on Measured Data ....... 146
8.5 Comparison of Bayesian Classifier and SVM ............................... 153

CHAPTER 9 CONCLUSIONS .................................................................. 157

APPENDIX ACRONYMS ....................................................................... 162

REFERENCES .................................................................................... 163
LIST OF FIGURES

Figure 1.1 Range profiles for two target classes ........................................ 16
Figure 2.1 Sample range profiles for training and test signatures ............... 28
Figure 3.1 Decision boundary for SVM ...................................................... 40
Figure 3.2 SVM separating hyperplane ..................................................... 41
Figure 4.1 Aspect angle dependence of target extent ................................. 46
Figure 4.2 Target extent ........................................................................... 46
Figure 4.3 Two class-conditional density functions with equal means .......... 55
Figure 4.4 Histograms of feature 5 ............................................................ 57
Figure 5.1 Sinusoids requiring analysis at different time resolutions ............ 67
Figure 5.2 Heisenberg rectangles of constant area ..................................... 68
Figure 5.3 Discrete wavelet decomposition ................................................. 73
Figure 5.4 Discrete wavelet packet decomposition .................................... 74
Figure 5.5 Image of wavelet coefficients ..................................................... 76
Figure 5.6 Image of wavelet packet coefficients ........................................ 76
Figure 6.1 Wavelet packet decomposition of signal of length 32 ............... 82
Figure 6.2 KS test statistic ....................................................................... 87
Figure 6.3 Comparison of ECDFs of two classes with and without outlier .... 88
Figure 6.4 Sample signals generated from the triangular waveforms .......... 95
Figure 6.5 Wavelet packet decomposition for triangular waveform .......... 97
Figure 6.6 LDB node scores over all classes and basis selected .................. 100
LIST OF FIGURES - continued

Figure 6.7  KS scores and resulting basis selected ............................................. 104
Figure 7.1  Decision boundaries with poor selection of kernel hyperparameter ................................................................. 108
Figure 7.2  Class-conditional margin density estimates for SVMs. ................. 110
Figure 7.3  Sample similarity matrix ................................................................. 114
Figure 7.4  Similarity matrices for various kernel width selections .................. 115
Figure 7.5  Subset of similarity matrix used for *semi-alignment* .................... 118
Figure 7.6  Similarity matrix used for *alignment* ............................................. 118
Figure 7.7  Kernel *semi-alignment* algorithm ............................................... 119
Figure 7.8  Kernel optimization performance evaluation steps ....................... 120
Figure 7.9  Sample checkerboard and quadboard data sets .......................... 121
Figure 7.10 Decision boundaries and associated class-conditional margin distributions for the checkerboard data .......................................................... 123
Figure 7.11 Classification efficacy error with *semi-alignment* for the checkerboard data .......................................................... 124
Figure 7.12 Classification efficacy error with *semi-alignment* for the quadboard data .......................................................... 126
Figure 8.1  Block diagram of feature selection and classification approach .. 135
Figure 8.2  Steps for analysis of classification efficacy ..................................... 135
Figure 8.3  Simulated data set results with full sample support ..................... 138
Figure 8.4  Measured data set results with full sample support ..................... 138
Figure 8.5  L1 classification efficacy for simulated data ............................... 142
Figure 8.6  L2 classification efficacy for simulated data ............................... 142
LIST OF FIGURES - continued

Figure 8.7  Minimax Classification efficacy for simulated data.......................... 143
Figure 8.8  Sup classification efficacy for simulated data................................. 143
Figure 8.9  L1 classification efficacy for measured data................................. 144
Figure 8.10 L2 classification efficacy for measured data................................. 144
Figure 8.11 Minimax classification efficacy for measured data....................... 145
Figure 8.12 Sup classification efficacy for measured data............................... 145
Figure 8.13 KS L1 classification results for \( \sigma \) selection techniques............. 149
Figure 8.14 KS L2 classification results of \( \sigma \) selection techniques............... 149
Figure 8.15 KS minimax classification results of \( \sigma \) selection techniques....... 150
Figure 8.16 KS sup classification results of \( \sigma \) selection techniques............. 150
Figure 8.17 KL L1 classification results of \( \sigma \) selection techniques............. 151
Figure 8.18 KL L2 classification results of \( \sigma \) selection techniques............. 151
Figure 8.19 KL minimax classification results of \( \sigma \) selection techniques....... 152
Figure 8.20 KL sup classification results of \( \sigma \) selection techniques............. 152
Figure 8.21 KS minimax SVM and Bayesian classifier results......................... 152
Figure 8.22 KS sup SVM and Bayesian classifier results................................. 155
Figure 8.23 Class-conditional densities of two features for the checkerboard data.................................................. 156
Figure 8.24 Checkerboard data quadratic decision boundary........................... 156
LIST OF TABLES

Table 2.1 Data polarization types ................................................................. 26
Table 2.2 Number of test samples per class for simulated data set ............... 29
Table 2.3 Number of test samples per class for measured data set .............. 30
Table 4.1 Feature notation .......................................................................... 45
Table 4.2 Feature equations ........................................................................ 54
Table 4.3 Individual feature classification results ...................................... 58
Table 4.4 Confusion matrix of feature 6-RL ............................................. 59
Table 4.5 Confusion matrix with 3 features ............................................... 60
Table 4.6 Confusion matrix with 2 features ............................................... 60
Table 4.7 Confusion matrix for feature set 1 ............................................. 62
Table 4.8 Confusion matrix for feature set 2 ............................................. 62
Table 4.9 Confusion matrix for feature set 3 ............................................. 63
Table 4.10 Confusion matrix for common features .................................. 63
Table 6.1 Dimension of class energy map .................................................. 82
Table 6.2 Sample matrix scores ................................................................. 90
Table 6.3 Dimension of wavelet packets ................................................. 92
Table 6.4 LDB node scores ..................................................................... 99
Table 6.5 KS individual wavelet packet sample point scores .................... 102
Table 6.6 KS minimax node scores ........................................................... 103
LIST OF TABLES – continued

Table 7.1  Relationship between kernel hyperparameter, SVM support vector
statistics, and PCC statistics for the checkerboard data set........... 125

Table 7.2  Relationship between kernel hyperparameter, SVM support vector
statistics, and PCC statistics for the quadboard data set.................. 127

Table 7.3  Reduced sample support results for the checkerboard data set........... 129

Table 7.4  Reduced sample support results for the quadboard data set................. 130
ABSTRACT

The challenge of target classification is addressed in this work with both feature extraction and classifier hyperparameter optimization investigations. Simulated and measured high-range resolution radar data is processed, features are selected, and the resulting features are given to a classifier. For feature extraction, we examine two techniques. The first is a supervised method requiring an "expert" to identify and construct features. The performance of this approach served as motivation for the second technique, an automated wavelet packet basis approach. For this approach, we develop the Kolmogorov-Smirnov best-basis technique that utilizes empirical cumulative distribution functions and results in improved classification performance at low dimensionality. To measure classification efficacy, we use a quadratic Bayesian classifier, which assumes a Gaussian distribution as well as a support vector machine. The support vector machine is a classifier, which has generated excitement and interest in the pattern recognition community due to its generalization, performance, and ability to operate in high dimensional feature spaces. Although support vector machines are generated without the use of user-specified models, required hyperparameters, such as kernel width, are usually user-specified or experimentally derived. We develop techniques to optimize selection of these hyperparameters. These approaches allow us to characterize the problem, ultimately resulting in an automated approach for optimization, semi-alignment.
CHAPTER 1
INTRODUCTION

A perennial challenge in digital signal and image processing is that of automatic target recognition (ATR) and object classification. In defense operations in particular, rapid object discrimination is critical. Synthetic aperture and millimeter wave radar have been used in a variety of defense applications such as reconnaissance and targeting of vehicles, aircraft, and fixed structures [11][35][41][52].

A technique that offers a reduction in processing requirements or an improvement in object discrimination can provide significant optimization advantages for real-time operations. For instance, for radar signals and synthetic aperture radar (SAR) images, target signatures vary dramatically with respect to aspect angle, often forcing the practitioner to characterize and store pose-dependent target models for classification. In operational environments, data storage and throughput requirements can impose a search for a reduction in the dimensionality of the feature space with the hope of reduced operational requirements. A technique that extracts the informative subspace, thereby maintaining the classification efficacy of the original data, is highly desirable.

Maturation of millimeter wave (MMW) radar technology has led to the potential for exploitation of range profile signatures in the detection and identification
of targets from the surrounding background [2][3]. This potential is realizable due to the MMW wavelengths and the corresponding high-range resolution obtainable at these frequencies. An additional benefit is the capability of MMW energy at key frequency bands to penetrate rain, dust, fog and smoke [5][27] and provide an associated narrow radar beam width pattern. The narrow radar beam width results in improved cross-range resolution with the accompanying increase in the signal-to-clutter ratio (SCR) (a). This increase in SCR is a significant benefit to detection/identification of targets due to the inherent suppression of terrain-induced clutter present in air-to-ground scenarios [5][27]. We seek to identify salient features from the range profiles derived from radar target signatures.

Research into feature extraction and classification techniques for MMW high-range resolution (HRR) radar data abounds [35][36][41][42][43][48][50][51][55][74][75][76]. While this problem has been studied extensively, it remains a challenge due to the complex and time-intensive methods of classical solutions.

1.1 Signal Representation in Feature Space

Feature selection involves developing a suite of statistics (in the broad sense) that summarize and maximize the discriminatory information of the original signals. If the set of features is "sufficient," the set provides all the necessary information content of the original signal. It is the goal of feature extraction to select a set of
statistics that provide "sufficiency" for discriminating between the target classes of interest; these are not necessarily the same statistics required for target signal characterization. In Figure 1.1, the range profiles for two different classes are displayed. It is these differences that we seek to exploit in the feature development process.

We consider two approaches to feature selection. The first utilizes classical feature extraction techniques and requires an "expert" with some knowledge of the type of data under question. This approach served to provide the motivation for the second approach, an automated method. An alternate approach to feature estimation is through the development of physical models. In fact, it might be argued that a thorough understanding of the physical phenomena involved should yield a feature set that captures the most information in the fewest coordinates. It is the lack of complete a priori knowledge that removes this choice from consideration.

Figure 1.1 Range vs amplitude profiles for target classes (a) Class 1. (b) Class 2.
Our first and more traditional method of determining features involves selection of salient features based on signal heuristics. For instance, we might experiment by selecting features based on some statistical method, adding additional features until the results are determined to be "good". As an example, feature 1 might be the signal mean; for feature 2 the standard deviation could be selected. We continue to add features based on higher order statistics or other combinations of the data until we get "good" results [22]. This transforms our signal from the signal space to a new feature space of reduced dimensionality. In general, this feature space is not a complete space; we cannot recover the signal from the feature space. An obvious disadvantage of this method is the requirement of an experienced feature estimator to select the most relevant features.

Our second method seeks an approach using a transformation to a new basis for feature extraction that removes the "expert" from the loop and provides a means of selecting features without specific knowledge of the target model. Additionally, the transient nature of energy in many real world signals has led to the desire for multi-resolution analysis to represent signals by both time and frequency characteristics. In target recognition problems in particular, it is often impractical to analyze data based solely on a choice of either frequency or time content of the signal. Rather, in order to characterize the signal efficiently, both localized time and frequency resolution is desired. Transforming the signal to a domain that provides
both time and frequency resolution can potentially produce superior representation of the signal in terms of dimensionality reduction and classification performance.

In the feature representation process using basis transformation, the data is converted from the signal space to a feature space via a projection to a new basis or coordinate system. A complete basis is composed of a set of orthogonal vectors and spans the original space. This definition leads to infinitely many possible bases. The objective is to decompose the signal into a new basis that well separates the given classes. As a secondary goal, we seek to exploit any information compression capability of the technique utilized, to offer the most pertinent information in the least coefficients, in order to reduce the dimensionality and increase the information content for this reduced dimension of coefficients.

How do we approach selecting a best basis? If extraction of frequency information is required, using the Fourier transform (FT) might provide a best basis. In this case, a sum of harmonic, sinusoidal kernels transforms the signal to the new space. An obvious drawback of the FT is its lack of localization due to the unbounded support of the kernels. The FT's frequency domain representation embodies the contribution of all input samples for the signal. For signals with transients, the windowed FT, or short-time FT, gives an excellent representation for some cases, but comes with its own set of limitations due to the windowing issues. The fixed scale of the windowed FT prevents it from automatically following the instantaneous frequency of rapidly varying signals. Therefore, for signals with both
localized time and frequency content of interest, this does not prove to be the most efficient basis in terms of characterizing the signal.

Other approaches for basis selection include the Karhunen-Loéve transform (KLT), or principal component analysis (PCA), which selects the eigenvectors associated with the largest eigenvalues as the features of choice. Selecting a small subset of these eigenvectors projects the data into a lower dimensional space while minimizing the reconstruction error in a least-squares sense [30][32]. While this approach results in data compression and the best least squares error representation of the data, it is not necessarily the best representation for discriminating between classes [30]. The Fisher linear discriminant [46] (Rayleigh quotient) seeks to find the best separation among classes by utilizing a difference of the means relative to the class covariances. Although this gives the optimal results for data with underlying Gaussian distributions, it can lead to poor results for multimodal data.

Decomposition of transient signals into an orthonormal library of time-frequency bases results in localization that has demonstrable benefits [18]. Wavelet orthonormal bases tend to realign and concentrate signal energy allowing the true information content to be concentrated and identified, leading to signal characterization and representation in a reduced feature space [8]. Wavelet packet bases provide a redundancy in the signal representation, which results in a multitude of possible bases. The need for an approach to automating basis selection becomes apparent due to the dimensionality of the basis selection process. Of particular in-
terest for the author is wavelet packet basis selection, exploiting the wavelet packets for enhanced class separability. The wavelet packet transform utilizes scaling and translation to allow for multi-resolution “zooming” in the time-frequency detail [45], making this an attractive option and the approach we utilize.

A novel approach to basis selection was introduced by Coifman and Wiener [20] to automatically select a best basis to represent the data. The original application of this procedure was for data compression; however, it has since been extended to discrimination, regression, signal de-mixing and controls. The Local Discriminant Basis (LDB) is an extension of this approach for application in a classification setting (Coifman and Saito [62][63][65]). We implement and analyze LDB and identify two limitations. First, LDB uses a type of class average approach for class representation that can be quite sensitive to outliers. An outlier is a data point that is far removed from the remaining data points and distorts the distribution, affecting the sample mean, while having little effect on the sample median [32]. Second, the scoring method used in the basis selection process uses an L1 norm over all pairs of classes, which can obfuscate individual class pair-wise separability. Our new approach uses empirical cumulative distributions with the Kolmogorov-Smirnov (KS) test statistic and pair-wise score matrices to address these limitations. In a complementary approach [64], Saito et al. have proposed a variation to the LDB algorithm that utilizes an alternative discriminant, the Kullback-Leibler (KL) distance, between empirically estimated class distributions (imple-
mented via average shifted histograms). This approach addresses the first limitation of LDB, but not the second.

1.2 Selection of Classifiers and Required Input Hyperparameters

In our investigation and analysis of salient features, we require a classification scheme to determine the classification efficacy. Classifier approaches abound. Parametric densities (like the Bayesian classifier) can be used with the requisite prior knowledge. Nonparametric approaches can be used with techniques such as Parzen windows [30] to estimate the density with no assumptions made on prior knowledge. Nearest neighbor based approaches result in twice the error rate of the Bayesian classifier. With $k$ nearest neighbor approaches, the results converge asymptotically to the Bayesian minimum error rate. In practice, we generally do not have adequate training samples to permit us to attain the asymptotic performance. Region based approaches include neural networks and support vector machines (SVMs).

We choose to use two classifiers. With knowledge of the true probability density functions (PDFs) and prior probabilities ($P(c)$), the Bayesian classifier is optimal [7] and results in a minimum data size for the classifier. The Bayesian classifier gives the minimum error rate with known PDFs and $P(c)s$. We estimate the PDFs and $P(c)s$, in our application of the classifier. Since we typically do not have the PDFs and $P(c)s$ for our feature space, we also considered an SVM. The
SVM finds the hyperplane that represents the maximal separation of the classes. Selection of the kernel input hyperparameters is critical to the performance of the SVM, directly impacting its generalization and classification efficacy. While our goal was not to determine the optimal classifier, we choose to spend considerable effort on the optimization of the input hyperparameters for the SVM classifier. We selected a Gaussian kernel and developed two approaches for selection of the Gaussian kernel width, $\sigma$.

The first technique uses class-conditional densities to set a range for training on values of $\sigma$. Kernel width is selected by minimizing the number of support vectors in the trained SVM over this range. Considerable insight into the behavior of the distribution of the margins is identified and characterized.

Our second technique amends recent work done by Cristianini et al. In [25], they define the concept of kernel alignment and discuss the selection of kernels and kernel hyperparameters as a normalized function of the Frobenius [46] inner product of the kernel Gram matrix and the outer product of the class labels. This simple, yet effective, statistic provides a measure for maximizing the within-class similarity (clustering) induced via the kernel and kernel hyperparameters, while penalizing the induced between-class similarity. This research amends the alignment approach with semi-alignment in a straightforward manner by applying the Frobenius inner product on a subset of the similarity matrix rather than on the entire matrix. By working with a subset of the matrix, we remove the within-class similarity of
the world class from consideration. For multi-class cases (greater than two classes), the statistic will no longer encourage the collection of classes in the world to look "alike". Unlike our first technique, this method does not require SVM training, thus eliminating a time-consuming step.

1.3 Scope of Dissertation

This study considers the identification, extraction and selection of features from MMW radar returns, providing the information necessary for automated target class identification for classifying vehicles on a battlefield. We begin in Chapter 2 with a description of the data sets that are used for the study. Chapter 3 contains a discussion of the classifiers used to determine classification efficacy. Of interest here is the SVM and a quadratic Bayesian classifier. Our initial feature extraction approach, utilizing classical heuristic feature selection, is presented in Chapter 4 along with analysis of this technique. The results of this method served simply to provide the motivation for the search for an automated feature extraction approach. The background necessary for the second approach, automated wavelet packet basis transformation feature selection, is presented in Chapter 5. In Chapter 6, we define LDB basis selection and our new basis selection technique based on a fitness function using a statistical test, the KS test, to measure the distributional differences between class-conditional probability densities in determining basis efficacy in a multi-class setting. In Chapter 7, we investigate and develop two novel tech-
niques for SVM hyperparameter selection. The first allows considerable insight into selection of the hyperparameter while the second method provides an automated approach that is applicable in a multi-class setting. Results of the wavelet-based feature extraction techniques are presented in Chapter 8, where we show that the KS best-basis approach that we developed results in improved performance at low dimensionality when compared to LDB. This is followed by concluding remarks in Chapter 9.
CHAPTER 2

TRADE STUDY EXPERIMENTAL DATA

Our research considers the extraction and selection of features from millimeter wave (MMW) radar returns providing the information necessary for automated target class identification from target signatures of scenes. For study purposes, real-beam MMW radar analysis [5] relies on identifying features that may be recovered from the range profile of the target under test. The range profile is derived from target signal returns and is a representation of the processed signal return versus range bin (distance to the target). Real-beam refers to a target signature acquired by a radar sensor, without the benefit of post-processing to adaptively shape the beam. Each signal will be defined for study purposes as a sample and is a \( 1 \times N \) target signature. We index into the samples by the range bin number or simply by the index of the sample. Features are estimated through a suite of discriminating signature features for the targets of interest, as defined in Chapter 4, and with wavelet based approaches as defined in Chapter 6. This chapter specifies the data used for testing classification efficacy.

The data sets used for testing the algorithms consisted of SAR and inverse SAR (ISAR) images. Two data sets were used: a fully polarimetric simulated four-class data set and a three-class single polarization measured data set. Detection of the target was pre-supposed in both cases, as this study was geared towards the
relative analysis of feature extraction and best-basis selection algorithmic performance. The complex target signatures were converted to real-valued magnitude profiles for processing and analysis. We begin by defining the polarization notation that is used throughout the paper followed by a description of both our simulated and measured data sets.

2.1 Polarization Definitions

Targets of interest for our simulated data set were characterized using fully polarimetric radar returns. We present a brief discussion of the polarimetric terms used throughout our study in this section. The polarization types and their respective polarization identifiers are presented in Table 2.1. Many good references are available for the interested reader [5][6][27][31].

<table>
<thead>
<tr>
<th>Transmit</th>
<th>Receive</th>
<th>Polarization Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal</td>
<td>Horizontal</td>
<td>HH</td>
</tr>
<tr>
<td>Vertical</td>
<td>Vertical</td>
<td>VV</td>
</tr>
<tr>
<td>Horizontal</td>
<td>Vertical</td>
<td>HV</td>
</tr>
<tr>
<td>Right Circular</td>
<td>Right Circular</td>
<td>RR</td>
</tr>
<tr>
<td>Right Circular</td>
<td>Left Circular</td>
<td>RL</td>
</tr>
</tbody>
</table>

A target radar cross-section (RCS), \( \sigma \), refers to the amount of power that the target reflects back to a sensor and is commonly described by the scattering matrix [1][27]:
\[
\sigma = \begin{bmatrix}
\sigma_{HH} & \sigma_{VH} \\
\sigma_{HV} & \sigma_{VV}
\end{bmatrix}
\] (2.1)

The scattering matrix is usually represented by a circular or linear orthogonal basis. Collecting the full polarimetric signal allows the investigator to exploit all the information relating to target RCS. The advantages of extracting the polarimetric nature of a particular target RCS for detection and identification have been documented [42][52][76]. The heuristic feature investigation considers several features that require full polarimetric returns, while, where available, the full polarimetric data is used for the wavelet basis approach.

Several polarizations of data will be referenced. The full polarimetric set of raw data consists of VV, HH and HV polarizations. The linear polarizations (HH, VV, HV) were converted to circular polarization, RR and RL, as follows:

\[
RR = \frac{1}{2} \left( \sigma_{HH} - \sigma_{VV} \right) + j \sigma_{HV}
\] (2.2)

\[
RL = \frac{1}{2} \left( \sigma_{HH} + \sigma_{VV} \right)
\] (2.3)

RR and RL are derived from HH, VV and HV and are viewed as the full polarimetric case. RR can be thought of as the even bounce polarization (co-pol), while RL
is the odd bounce (cross-pol). RR requires full polarimetric data (HH, VV and HV), while RL requires both HH and VV.

2.2 Data

The SAR and ISAR images were converted to real-beam range profiles by means of frequency domain processing. Sample range profiles for two classes were shown in Figure 1.1. It is the differences in these profiles, at all sensor-target orientations, that we seek to exploit for classification of the targets. An example of training and test range profiles at the same pose corresponding to the first target in Figure 1.1 is shown in Figure 2.1.

![Sample range vs amplitude profiles for class 1. (a) Training signature. (b) Test signature.](image)

Figure 2.1 Sample range vs amplitude profiles for class 1. (a) Training signature. (b) Test signature.
The first data set consisted of simulated fully polarimetric ISAR images for ground vehicles at six inch resolution. Four classes of similar size vehicles were chosen to test the algorithms in a realistic multi-class environment. The training data consisted of 1440 ISAR targets, 360 per class, at 1 degree aspect-angle increments. The performance, or test, data set consisted of 712 of the ISAR targets inserted into complex SAR clutter at random aspect angles and random locations. The images were converted into real-beam range profile representation and then were converted to RR and RL polarizations via (2.2) and (2.3). It was the RR and RL polarizations that were used for classification. The number of samples per class for this test set is shown in Table 2.2. The tools to simulate this data set were provided by Research Network Inc. of Atlanta, GA.

<table>
<thead>
<tr>
<th>Class</th>
<th>Number of samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>182</td>
</tr>
<tr>
<td>2</td>
<td>174</td>
</tr>
<tr>
<td>3</td>
<td>129</td>
</tr>
<tr>
<td>4</td>
<td>227</td>
</tr>
</tbody>
</table>

The second data set consisted of a three-class (single polarization) set of ground vehicle measured data of which 1080 targets, 360 per class, were selected.
for training leaving 337 images for testing. The original ISAR images were converted to real-beam range profiles for algorithm performance testing. The breakdown by class for the test set is shown in Table 2.3.

<table>
<thead>
<tr>
<th>Class</th>
<th>Number of samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>112</td>
</tr>
<tr>
<td>2</td>
<td>109</td>
</tr>
<tr>
<td>3</td>
<td>116</td>
</tr>
</tbody>
</table>

For both data sets, target pose information is not considered, either in the training or testing phases, resulting in extraction of features that are representative of class differences across vehicle angular aspects. Aspect-angle variations present an important challenge in classification with radar signal signatures, since the signatures exhibit a high degree of aspect-angle dependence [5][27]. Desired features include those that exhibit little dependence on aspect angle with respect to the separation of classes. In our quest for angle-independent features, we group vehicle signal returns representing all aspects of a class into the same training set. With no knowledge of aspect angle, we exploit signal characteristics that are common at all aspects, thus forcing the selection of features exhibiting pose independence.
We now concern ourselves with how we assign a signal to a category. Volumes have been written about classification theory and approaches. Methods vary from the very simple to complex. The approach that yields the best results for a particular problem can often be elusive. While the goal of this work is not finding the optimal classifier, we seek to select a classifier to evaluate relative classification efficacy that quantifies the differences in our feature extraction techniques. With a set of training sample pairs, \( (x_i, y_i) \ldots (x_m, y_m) \), we map the data, \( x \), onto class labels, \( y \), to obtain a classifier function that determines the class label of future (test) samples. To this end, we consider two approaches to classification, a Bayesian classifier and the SVM classifier. The Bayesian classifier is known to give the minimum error when the prior probabilities and the class-conditional densities are known. As a rule of thumb, we use 5\( \cdot \)D training samples for adequate sample support [60], where D is the dimensionality of the data. With a set of 10 features, this results in 50 training samples; however, with data that is 128 dimensional, 640 training samples are generally needed. For the problems of interest for this study, it is rare to have 640 training samples, resulting in an ill-posed problem due to the high dimensionality. The SVM provides an alternate approach that allows the data to be mapped non-linearly to a higher dimensional space where a decision boundary is
defined and then mapped back to the lower dimensional space. There is an inherent reduction in dimensionality with this approach.

3.1 Bayesian Classifier

How likely is an event? Do we have this information prior to the event actually occurring? Bayesian decision theory assumes that we have \textit{a priori} knowledge of both the class probabilities and class-conditional densities. We use standard notation for Bayes' rule. Here, \( p(x|c_i) \) is the class-conditional density for the feature vector \( x \) given class \( i \), and \( P(c_i) \) is the probability mass or prior probability term. We begin our discussion of Bayesian classification with Bayes' rule:

\[
P(c_i | x) = \frac{p(x | c_i)P(c_i)}{\sum_{j=1}^{m} p(x | c_j)P(c_j)}
\]  

(3.1)

where \( m \) is the total number of classes. Here we see that the prior probabilities, \( P(c_i) \), and class-conditional densities, \( p(x|c_i) \), are converted to \textit{a posteriori} probabilities, \( P(c_i|x) \). The Bayesian classifier allows expression of the classification results probabilistically, and results in the minimum average probability of error. We implement a quadratic form of the classifier. For the data sets of interest for this study, the class-conditional densities and prior probabilities are unknown. As is often the case, one can assume that the features are independent and normally distributed, allowing us to estimate a Bayesian classifier with a Gaussian kernel. The lack of knowledge of prior probabilities can be handled by assigning equal
probabilities to each class. Note that the denominator is essentially a scaling term, which we will omit in subsequent references without loss of generality. The multivariate Gaussian distribution is

\[
p(x) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{(x - \mu)^T \Sigma^{-1} (x - \mu)}{2}\right)
\]

(3.2)

Here \(x\) is the data (a group of column vectors), \(\mu\) is the data mean for each dimension, \(\Sigma\) is the covariance matrix of the data, \(\Sigma^{-1}\) is its inverse, \(|\Sigma|\) is the determinant, and \(d\) is the dimensionality of the data. We express this in terms of the class-conditional density:

\[
p(x | c_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{(x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i)}{2}\right)
\]

(3.3)

We seek a discriminant function that maximizes \(p(x | c_i)\). The discriminant function, \(g_i(x)\), allows us to accomplish this.

\[
g_i(x) = \ln p(x | c_i) + \ln P(c_i)
\]

\[
= -\frac{d}{2} \ln(2\pi) - \frac{1}{2} \ln |\Sigma_i| - \frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) + \ln P(c_i)
\]

\[
= -\frac{d}{2} \ln(2\pi) - \frac{1}{2} x^T \Sigma_i^{-1} x + \left(\Sigma_i^{-1} \mu_i\right)^T x - \frac{1}{2} \mu_i^T \Sigma_i^{-1} \mu_i - \frac{d}{2} \ln |\Sigma_i| + \ln P(c_i)
\]

(3.4)

We modify the function further by noting that the term \(-\frac{d}{2} \ln(2\pi)\) is independent of \(i\) and omit it from the final form of \(g_i(x)\), which is expressed as
Following the notation of [30], this can be expressed as

\[ g_i(x) = x^T W_i x + w_i^T x + w_{i0} \]  

(3.6)

where

\[ W_i = \frac{1}{2} \Sigma_i^{-1} \]

\[ w_i = \Sigma_i^{-1} \mu_i \]

\[ w_{i0} = -\frac{1}{2} \mu_i^T \Sigma_i^{-1} \mu_i - \frac{1}{2} \ln |\Sigma_i| + \ln P(c_i) \]

The test data is declared to be the class that results in the maximum value of \( g_i(x) \).

This formulation allows separation into terms that can be calculated after training \( (W_i, w_i, \text{and } w_{i0}) \), but prior to classification, for application to the test data \( x \) during classification. We note that the Gaussian assumption means that the discriminant function is completely characterized by its first two moments:

\[ \mu_i = \text{Statistical mean vector of the training data set for the } i^{th} \text{ class.} \]

\[ \Sigma_i = \text{Statistical covariance matrix of the training data set for the } i^{th} \text{ class.} \]

### 3.2 Support Vector Machines

SVMs are classification and regression algorithms developed by Vapnik [70] and others [1][9] and were inspired from theoretical concepts of statistical learning...
theory. SVMs work by mapping input patterns to an alternative high-dimensional space where linear separation becomes possible by the use of a nonlinear transformation, and finding the optimal hyperplane in that space that separates the patterns of each class. Optimality is defined to be the identification of the hyperplane that results in the maximum separation between the pairs of classes [16] and depends on the type of support vector training optimization [24] and the associated constraints imposed. The SVM margin of a training set refers to the boundaries that are used to select the most difficult patterns to classify. The cost of misclassification, C, defines the margin type for the SVM. We cannot simultaneously maximize the support vector margin and minimize the error but rather control the tradeoff between the two with the value of C. A C set to infinity is a hard margin SVM while a lower value of C results in a soft margin SVM. Large values of C can cause overtraining of the data, resulting in poor generalization of the SVM. For the data of interest for this study, it was found that there was considerable flexibility in the value of C without differences noted in the classification efficacy. Therefore, C was set to a value of 10. For other data applications, the value may have a greater impact on performance of the SVM and may need to be studied in greater detail. In general, a hyperplane is produce which maximally separates the closest patterns from each class. SVMs have generated excitement and interest in the pattern recognition community due to their generalization, performance, and ability to operate in high-dimensional feature spaces. Although SVMs are generated without the use
of user-specified models, a kernel and its associated hyperparameters, such as the Gaussian kernel width, are usually user specified or experimentally derived. Determining valid hyperparameters for a Gaussian kernel is addressed in Chapter 7.

Prior to our definition of SVMs, we present a brief explanation of the kernel trick, which we exploit in the implementation of SVMs. The kernel trick [34][69] is a method of implicitly computing dot products in feature space by casting a kernel function in terms of dot products given in

\[ \langle \Phi(x_i), \Phi(x_j) \rangle = K(x_i, x_j) \]  \hspace{1cm} (3.7)

An excellent tutorial on the kernel trick is given in [69]. We define our training examples as \((x_i, y_i), \ldots, (x_n, y_n)\), where \(x_i\) are the sample points and \(y_i\) are the labels. Our interest is in avoiding actually operating in a high dimensional space by mapping into the high dimensional feature space, \(\mathcal{F}\), using dot products. This can be accomplished by selecting a feature space where the dot product can be evaluated directly from the input signal space, \(x\), and a kernel function, \(K\) using (3.7). We are not interested in determining the dot product functions, but rather selecting a kernel for which the kernel trick equation holds.

We form a Gram matrix by evaluating the input data with the kernel, \(K\). We will refer to this Gram matrix as the similarity matrix, \(G\), which is, in essence, a clustering of the data where the resulting matrix values give a similarity measure between classes. This positive, semi-definite matrix has nonnegative, real eigen-
values and is symmetric with all the necessary training information for the SVM contained in the similarity matrix:

\[ G(i,j) = K(x_i, x_j) \quad 1 \leq i, j \leq n \quad (3.8) \]

where \( n \) represents the total number of training samples. We operate on this matrix.

We now begin with a formal definition of the SVM. Given a training set \( S = \{(x_i, y_i), \ldots, (x_n, y_n)\} \), composed of \( n \) \( D \)-dimensional patterns \( x_i \in X \) and associated class labels \( y_i \in \{-1, 1\} \), a Support Vector Machine is a linear function of the form

\[ f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b \quad (3.9) \]

Variables \( \alpha_i \) are Lagrange multipliers, whose values are derived via maximizing

\[ L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad (3.10) \]

subject to the constraints,

\[ \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha_i \geq 0 \quad \forall i = 1, \ldots, n \quad (3.11) \]

All training patterns with non-zero \( \alpha_i \) are defined as support vectors. Support vectors are the set of patterns that lie closest to the hyperplane in the projected feature space. These represent the most difficult patterns to classify or those that are the most informative for classification purposes [30]. Training patterns associ-
ated with zero-valued Lagrange multipliers ($\alpha_i=0$) are non-contributors to the evaluation of the optimal hyperplane, $f(x)$. Ideally, the number of support vectors is less than the number of training patterns, inherently performing a reduction of the data required for SVM representation. The bias term, $b$, is the offset of the hyperplane from the feature space origin. $K$ is the kernel function, which defines the mapping to the feature space as a function of the training data. Favorable kernel functions include the linear, polynomial, and Gaussian or radial-basis function kernels, and are defined respectively as

$$K(u,v) = u \cdot v$$

(3.12)

$$K_p(u,v) = (u \cdot v + 1)^p$$

(3.13)

$$K_r(u,v) = \exp\left(-\frac{|u - v|^2}{2\sigma^2}\right)$$

(3.14)

For classification problems, the separating hyperplane defines the decision boundary, and the assignment of a test pattern, $x$, is determined by which side of the hyperplane the pattern falls on in the feature space as defined by the signed form of (3.9) in

$$f(x) = \text{sgn}\left[\sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b\right]$$

(3.15)
where $i$ is indexed over all training samples. Positive values with this formulation of $f(x)$ indicate a class match between the training and test samples. Recall that $\alpha_i = 0$ for non-support vector samples, so that the number of non-zero terms is less than or equal to $n$ giving an inherent reduction in dimensionality with a well chosen SVM. The SVM training is performed with constrained quadratic optimization. Approaches to SVM optimization generally use a gradient descent (e.g. Kernel-Adatron) [24] approach or attempt some sort of chunking (e.g. sequential minimization optimization) [37][57]. See [24][37][57] for algorithmic implementations.

In a multi-class setting, SVM training is performed as a sequence of two-class problems, i.e. object versus everything else (world). For a $k$ class problem, $k$ SVMs are generated during training, one for each class. Classification of a new test pattern consists of assigning the new pattern to the class with the largest positive SVM value as shown in

$$c(x) = \arg \max_{1 \leq i \leq m} f_i(x)$$

where $m$ is the number of classes. For each input test vector, $x$, a score is calculated for each trained class. The class with the largest value is declared to be the class of the test data vector.

The margin of an example $(x_i, y_i)$ with respect to a real-valued function, $f$, is defined [23]:

This is the distance of each sample point from the decision boundary or hyperplane. Note that a point is correctly classified if $y_i > 0$ since $y_i$ is positive for the class of interest and negative for the world class, as is the value of $f(x_i)$. The distribution of margins of a training set, $S$, is defined as the margin distribution of $f$ with respect to $S$. The class label changes the sign of the margin values associated with the $\{-1\}$ labeled patterns. To ease analysis of the margin distributions associated with each class, one can separate the training set, $S$, into class-conditional sets, $S_i$ and $S_{-i}$, and define a class-conditional or signed margin:

$$y_i = y_i f(x_i)$$

(3.17)

Figure 3.1 Decision boundary for SVM. $f(x) = 0$ (solid line) defines the boundary between the class of interest and the world. Dashed lines indicate margin (distances) to support vector points ($y_i = f(x_i)$).
In the SVM framework, the signed margin corresponds to the value of the signed distance of the pattern to the optimal hyperplane, which is equivalent to the projection of the feature vector in the high-dimensional space onto the vector defining the hyperplane. Figure 3.1 illustrates this concept in a two-dimensional case.

When referring to the entire training set, the margin of the training set is defined to be the lines corresponding to $y_i f(x_i) = 1$. Figure 3.2 shows a two-dimensional representation of the separating hyperplane and margins associated with the support vectors. Note that only the points closest to the margins are retained as support vectors (those corresponding to margins $\leq 1$), thus reducing the complexity of the SVM.

Figure 3.2 Two-dimensional representation of separating hyperplane for two classes. Support vectors are defined as those points corresponding to margin values $\leq 1$. 
CHAPTER 4

A HEURISTIC FEATURE APPROACH

We begin by presenting a case study of feature selection based on signal heuristics (see [10]). With an “expert” driving the selection, we determine relevant features based on the expert’s intuition and/or past experience. The features selected were classical target features and proved to be suboptimal in terms of classification efficacy while requiring considerable human “tweaking” for a particular target set. This study showed the difficulty of heuristic feature selection and provided the motivation for the remainder of the work presented in this paper.

For heuristic features, we might experiment by selecting statistical features (such as mean and variance) and then add additional features such as target extent. We continue adding features until the results are determined to be “good”. Due to the aspect-angle dependence of radar data, desired features include those that exhibit little dependence on aspect angle with respect to the separation of classes.

We define a suite of potential features for investigation in this chapter. The suite of candidate features includes features that measure classical statistics [48][59], as well as classical radar techniques such as target extent relative to several distinct thresholds, including both clutter and peak response perspectives [61]. Investigating the target extent is particularly interesting with aspect-angle independent analysis. A target at an aspect of 90 degrees typically has a different extent
than the same target at 180 degrees. In the development of the features, the advantages of full polarimetric data can be exploited for many features. Frequently, features make use of the polarimetric characteristics of differences that may be present between classes of objects [42][52][76]. Using a second polarization essentially results in an additional set of the single polarization features. Ratios of characteristics of the polarimetric signatures can be exploited. Additional techniques considered include cross-correlation and matched filtering to produce estimates of features [27][40][44][53]. Exploiting spectral characteristics of the target signatures leads to several features [27][44].

Once a set of features has been identified, we seek the “best” subset of features. Several automated approaches for feature selection exist [32], but generally no approach to date can guarantee extraction of the optimal set [22]. We chose to keep the human “expert” in the loop and used feature histograms to drive our feature selection process. Those features that appeared to have the best separation between classes for the full training set were selected as candidates. The techniques mirror classical radar measures such as target extent, as well as common signal processing techniques such as template matching for the range profile of a target, with the feature generalized across all aspect angles to a single vector representation. The selected features included several derived from full polarimetric data, as well as several that use data of a single polarization. We further refined the feature
selection process by measuring classification efficacy for candidate features. A discussion of the features follows.

4.1 Standard Definitions for Feature Calculations

We first identify some terminology used in the development of our feature set. Information that is common to multiple features is presented; several features utilize target extent and average clutter. These are defined, followed by a discussion of individual features. Many of the features are determined relative to target specific thresholds and filter window sizes. Although calculation of these values could be automated, the initial thresholds and window sizes were determined experimentally for our data set. The results of this heuristic feature approach proved to be a motivation for continued investigation for a new feature extraction approach, and automation was not implemented.

The symbolic terminology used in the definition of these features is defined here. Some of the notation is quite standard, but is specified in Table 4.1 for clarity.
Table 4.1 Feature notation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{x}$</td>
<td>Target signal mean</td>
</tr>
<tr>
<td>$x(i)$</td>
<td>Individual target return sample</td>
</tr>
<tr>
<td>$\bar{x}_{clut}$</td>
<td>Clutter signal mean</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Target signal standard deviation</td>
</tr>
<tr>
<td>$\sigma_{clut}$</td>
<td>Clutter signal standard deviation</td>
</tr>
<tr>
<td>$k$</td>
<td>Truncated signal length</td>
</tr>
<tr>
<td>$NTE_i$</td>
<td>Near target extent, index $i$ indicates the method of declaring the $NTE$</td>
</tr>
<tr>
<td>$FTE_i$</td>
<td>Far target extent, index $i$ indicates the method of declaring the $FTE$</td>
</tr>
<tr>
<td>$x_{peak}$</td>
<td>Target signal peak</td>
</tr>
<tr>
<td>$x_{peak2}$</td>
<td>Second largest target signal peak</td>
</tr>
</tbody>
</table>

4.1.1 Target Extent

Target extent refers to the length of the target relative to a threshold. For many targets, the extent differs greatly with aspect or viewing angle (see Figure 4.1). The depression angle of the sensor also contributes to variations in the extent. The height of the target can be a factor and, in many cases, must be considered in order to determine the impact of range layover on the target extent. Range layover occurs when target returns from elevated targets appear closer to the sensor than scatterers directly below [5]. A look-up table might be utilized where target extent is based on predictable changes such as those mentioned.
Near target extent ($NTE$) is determined to be the closest range bin in which the target resides according to the respective feature definition. Far target extent ($FTE$) is the farthest range bin from the sensor in which the target resides. Figure 4.2 depicts the determination of target extent. The range bin can be thought of as a distance partitioning of the x-axis. The $NTE$ and $FTE$ can be determined relative to several thresholds, which will be apparent in the description of specific features.

Figure 4.1 Target extent is dependent on aspect viewing angle. (a) Vehicle at 0° aspect angle. (b) Vehicle at 90° aspect angle.

Figure 4.2 Target extent = $FTE - NTE$. 
4.1.2 Average Clutter

Average clutter is determined from data residing \( n \) bins away from the peak of the target, where \( n \) is selected such that no target returns are introduced into the clutter calculation. The value of \( n \) is data dependent, but can be calculated experimentally simply by allowing \( n \) to be large enough to ensure a target-free region. Clearly, using average clutter in the calculation of a feature requires training on sample sets at all clutter levels of interest. This can be efficiently managed by first determining the clutter level for a test sample and then using a look-up table of features from the training set at the specified clutter level.

4.2 Features

We next describe the suite of discriminating features that we derive from the radar signal returns. The first eight are developed from a single polarization; adding the second polarization results in a second set of eight features. The remainder are formed using the full polarimetric data. It is from our full set of features that we will select a subset for classification.

The first feature is the statistical mean of the truncated target signal return:

\[
F1 = \bar{x} = \frac{1}{k} \sum_{i=1}^{k} x(i)
\]  

(4.1)

We truncate the signal length to \( \pm k/2 \) bins centered at the signal peak in the calculation of this feature. Recall that \( k \) was determined experimentally for the data set.
The coefficient of variation, $C_v$, is a measure of the relative scatter, or dispersion, of data with respect to the mean. It has found application in interpretation of data in many fields, such as immunology [59]. The $C_v$ is calculated as

$$F2 = C_v = \frac{\sigma}{x}$$

(4.2)

where we again work with the truncated signal as described for feature 1 in order to determine the mean and standard deviation.

Feature 3 is the target extent of the input data from a clutter perspective. We first calculate the average clutter as described in Section 4.1.2. Beginning $k/2$ bins prior to the peak target response and proceeding in towards the peak, the SCR is determined by dividing the current range bin response by the average clutter. Once this value exceeds a predetermined threshold, the $NTE_i$ is declared. The $FTE_i$ is determined by starting $k/2$ bins past the peak target response and proceeding towards the peak in the same manner. The target extent is defined as the distance in range bins between the $FTE_i$ and the $NTE_i$:

$$F3 = FTE_i - NTE_i$$

(4.3)

A second target extent can be determined by setting a new definition of the threshold for $NTE$ and $FTE$. Feature 4 is the target extent of the input data from a peak signal response perspective. Beginning $k/2$ bins prior to the peak target response and proceeding inwards towards the peak, the ratio of the peak response to the current range bin response is determined. Once this value is less than a prede-
terminated threshold, the $NTE_2$ is declared. The $FTE_2$ is determined by starting in a likewise manner $k/2$ bins past the peak target response and proceeding towards the peak. This target extent is again defined as the distance in range bins between the $FTE_2$ and the $NTE_2$:

$$F4 = FTE_2 - NTE_2$$

(4.4)

For feature 5, the discrete Fourier transform (DFT) is used to average the low-frequency components of the signal return for target discrimination. We begin by finding the peak response of the signal return. The sequence is truncated to a length of $N$ (integer power of 2) centered around the peak. A DFT of length $N$ is calculated on this truncated sequence and normalized by the dc component. Finally, we average the first $M < N$ frequency components to determine feature 5:

$$F5 = \frac{1}{M} \sum_{k=0}^{M-1} \frac{\left| X(k) \right|}{\left| X(0) \right|}$$

(4.5)

where

$$X(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n)e^{-\frac{2\pi nk}{N}}$$

(4.6)
Feature 6 [61] uses a moving average filter, of window length $m$, to pre-process the signal return, and the mean for each windowed segment replaces the raw value for the center point of the window. From this filtered return (which we will call $x$), the target area is determined to be the portion of the signal $\pm k/2$ bins centered at the signal peak. The clutter portion consists of $n$ samples of the filtered return, both before and after the target, separated from the signal by $p$ guard bins, where $p$ is set to be large enough to insure that the clutter and target returns are well separated. The values of $k$, $m$, $n$, and $p$ can be determined experimentally or can be automated. They were determined experimentally for this data set. The first and second moments of the data and clutter samples form this feature. A new value, $s(i)$, is computed for each point in the windowed signal based on:

$$s(i) = [\bar{x}(i) - \bar{x}_{clut}(i)] \ [\sigma(i) - \sigma_{clut}(i)]$$ \hspace{1cm} (4.7)

The feature is calculated from $s$, and is the amplitude difference between the maximum and minimum responses of $s$ given by:

$$F6 = 10 \log_{10}(s_{peak} - s_{min})$$ \hspace{1cm} (4.8)

For feature 7, we investigate the relationship between the ratio of the peak signal response to the second strongest peak. After finding the peak response, the first nulls both prior to and after the peak are found and then the second peak in each direction is determined. We select the maximum of these second peaks to use
for the calculation. The logarithm of the peak response to the maximum of these second peaks is feature 7:

\[ F7 = 20 \log_{10} \left( \frac{x_{\text{peak}}}{x_{\text{peak2}}} \right) \] (4.9)

Preprocessing the signal returns via the moving average filter attenuates many spurious portions of the signal return [53]. This feature uses the moving average filter as described for feature 6. The mean of the clutter is considered to be the clutter level, while a peak value is determined from the target region. This feature is the logarithm of the ratio of the peak moving averaged filtered signal response to the mean clutter level:

\[ F8 = 20 \log_{10} \left( \frac{x_{\text{peak}}}{x_{\text{clut}}} \right) \] (4.10)

The last three features use the full polarimetric data. Feature 9 uses a standard signal processing tool, correlation, to develop a metric for the polarimetric channels. This feature is defined as the peak response of the outputs of the cross-correlation of the RR and RL channels. We start with data sequences of the RR and RL returns, truncated around the peak responses to a length of \( k \) (again determined experimentally), and remove the mean from each sequence. Let \( x_{\text{RR}} \) be the
signal return from the RR polarization (Section 2.1) and \( x_{rl} \) the RL return. Similarly, we let \( \mu_{rr} \) be the RR polarization mean and \( \mu_{rl} \) be the RL mean. The polarimetric data sequences are individually transformed to the Fourier domain after removing their respective means, and then multiplied together to form one sequence on which we calculate the inverse DFT (IDFT):

\[
\text{IDFT}\{X(k)\} = \text{IDFT}\{ \text{DFT}(x_{rr} - \mu_{rr}) \text{DFT}(x_{rl} - \mu_{rl}) \} \tag{4.11}
\]

The maximum value in dB is \( F_9 \).

\[
F_9 = 20 \log_{10}(\text{IDFT}\{X(k)\})_{\text{max}} \tag{4.12}
\]

The cross-correlation should be higher for targets than for clutter since one would expect higher correlation of the returns for the targets across both RR and RL channels than for natural environmental clutter returns. This enhances the signal return for this feature.

Feature 10 is calculated by finding the peak response for the signal return as well as the mean clutter value. The signal is then truncated to \( \pm k/2 \) bins around the peak (including the peak). This sequence is formed for both polarimetric channels. The mean clutter value is removed from each point in the sequence and feature 10 is the calculated via

\[
F_{10} = 20 \log_{10} \left( \frac{1}{k} \sum_{i=1}^{k} \frac{x_{rl}(i) - x_{clut}(RL)}{x_{rr}(i) - x_{clut}(RR)} \right) \tag{4.13}
\]
Note that we take the logarithm to represent the value for this feature in a preferred range, but this can also be calculated without using the logarithm.

We form a matched filter type feature in order to explore a classic signal processing technique for development of this final feature [40]. An average response is formed from the full polarization training data in order to create the matched filter. After aligning the samples across polarizations, each sample is truncated to \( k \) points and a point-by-point summation across polarizations of the data is computed. These are averaged across all samples of the training data to form the response. We calculate the matched filter via

\[
I(n) = \frac{1}{m} \sum_{i=1}^{m} (x_{RR}(n) + x_{RL}(n)) \\
\forall n = 1, \ldots, k
\]  

(4.14)

where \( m \) is the number of training samples, and \( n \) is the index into each sample. We implement the matched filter in the Fourier domain and transform back to the time domain. Setting \( j \) to be the index of the first data point to cross an experimentally derived threshold, we implement this feature as:

\[
F11 = N - j
\]  

(4.15)

where \( N \) is the DFT length.

The feature equations are given in Table 4.2.
Table 4.2 Feature equations. Features 1 – 8 result in two sets of features (RR and RL), while features 9-11 require full polarimetric data.

<table>
<thead>
<tr>
<th>Feature Number</th>
<th>Polarization</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RR and RL</td>
<td>( F1 = \bar{x} = \frac{1}{k} \sum_{i=1}^{k} x(i) )</td>
</tr>
<tr>
<td>2</td>
<td>RR and RL</td>
<td>( F2 = C_v = \frac{\sigma}{\bar{x}} )</td>
</tr>
<tr>
<td>3</td>
<td>RR and RL</td>
<td>( F3 = FTE_1 - NTE_1 )</td>
</tr>
<tr>
<td>4</td>
<td>RR and RL</td>
<td>( F4 = FTE_2 - NTE_2 )</td>
</tr>
<tr>
<td>5</td>
<td>RR and RL</td>
<td>( F5 = \frac{1}{M} \sum_{k=0}^{M} \frac{</td>
</tr>
<tr>
<td>6</td>
<td>RR and RL</td>
<td>( F6 = 10 \log_{10}(s_{peak} - s_{min}) )</td>
</tr>
<tr>
<td>7</td>
<td>RR and RL</td>
<td>( F7 = 20 \log_{10} \left( \frac{x_{peak}}{x_{peak,2}} \right) )</td>
</tr>
<tr>
<td>8</td>
<td>RR and RL</td>
<td>( F8 = 20 \log_{10} \left( \frac{x_{peak}}{x_{clut}} \right) )</td>
</tr>
<tr>
<td>9</td>
<td>Full</td>
<td>( F9 = 20 \log_{10}(IDFT{X(k)})_{max} )</td>
</tr>
<tr>
<td>10</td>
<td>Full</td>
<td>( F10 = 20 \log_{10} \left( \frac{1}{k} \sum_{i=1}^{k} \frac{x_{RL}(i) - x_{clut(RL)}}{x_{RR}(i) - x_{clut(RR)}} \right) )</td>
</tr>
<tr>
<td>11</td>
<td>Full</td>
<td>( F11 = N - j )</td>
</tr>
</tbody>
</table>
4.3 Analysis

The features in this section are, in general, data dependent, requiring an "expert" to select features for a particular scenario. For instance, using feature 1 (mean) may discriminate well for targets that exhibit differences in their mean values but would, of course, perform poorly with classes of objects that appear similar in mean value. In the two-class example in Figure 4.3, the classes would appear inseparable using mean value alone, but, in fact, are substantially separable. An expert would be able to select features depending on the particular classes of interest; however, "good" features for one data set could have poor classification results for a second classification problem. We present our results with this feature approach in this section.

![Figure 4.3 Two class-conditional density functions with equal means.](image)

We test the features on the simulated four-class data set of ground vehicles discussed in Chapter 2, while we use the Bayesian classifier for determining classification efficacy as discussed in Chapter 3. We refer the reader to those chapters
and discuss our results in this section. These results will be presented in a confusion matrix format. The confusion matrix provides a statistically valid way of measuring the performance of the classification algorithms. Let \( P(C_i | C_j) \) be the empirical probability that an observed object of class \( C_j \) is classified as being a member of class \( C_i \). The confusion matrix indicates the performance measure of each individual class. For a four-class system, the \( 4 \times 4 \) confusion matrix would be:

\[
\mathbf{X} = \begin{bmatrix}
P(C_1 | C_1) & P(C_2 | C_1) & P(C_3 | C_1) & P(C_4 | C_1) \\
P(C_1 | C_2) & P(C_2 | C_2) & P(C_3 | C_2) & P(C_4 | C_2) \\
P(C_1 | C_3) & P(C_2 | C_3) & P(C_3 | C_3) & P(C_4 | C_3) \\
P(C_1 | C_4) & P(C_2 | C_4) & P(C_3 | C_4) & P(C_4 | C_4)
\end{bmatrix}
\]  

An ideal classifier would yield a confusion matrix equal to the identity matrix. We modify the confusion matrix to present results in percent correct classification (PCC) format. The PCC is an average over all classes; the terms on the diagonal are summed and then divided by the total number of test cases.

In order to select a subset of features for use, we began by generating histograms of the features for each class across all aspect angles. For the first eight features, two sets of features were generated, one for each polarization. Several features exhibited better separation of the classes in the histograms than others and initially looked like good candidates for class separation. This was borne out when looking at the single-feature classification results in Table 4.3. For example, ob-
serving the histograms of features 5-RL and 5-RR (Figure 4.4) we see that the class separation is significantly greater for 5-RL than for feature 5-RR. The PCC for 5-RL is 43.4%, while that for 5-RR is close to guessing for this four-class case. As can be seen from Table 4.3, the best PCC from a single feature is 50%. This leads to the conclusion that a single feature will not provide adequate information for classification and drives the search for the best combination of features.

From the individual classification results, we might select feature 6-RL as our first feature for the combined set of features. The confusion matrix for this feature is given in Table 4.4. Although the PCC is the greatest for any individual feature, we note that class 2 is missed 100% of the time in our classification with this
feature. Looking at the histogram in Figure 4.4 (a), we see that the distribution of class 2 is “hidden” in the other three classes. For our next feature, we might select a feature that separates class 2 well from the other classes.

Table 4.3 Individual feature classification results.

<table>
<thead>
<tr>
<th>Feature Number</th>
<th>Polarization</th>
<th>PCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RL</td>
<td>36.1</td>
</tr>
<tr>
<td>1</td>
<td>RR</td>
<td>20.0</td>
</tr>
<tr>
<td>2</td>
<td>RL</td>
<td>23.2</td>
</tr>
<tr>
<td>2</td>
<td>RR</td>
<td>33.6</td>
</tr>
<tr>
<td>3</td>
<td>RL</td>
<td>36.3</td>
</tr>
<tr>
<td>3</td>
<td>RR</td>
<td>21.8</td>
</tr>
<tr>
<td>4</td>
<td>RL</td>
<td>36.9</td>
</tr>
<tr>
<td>4</td>
<td>RR</td>
<td>30.1</td>
</tr>
<tr>
<td>5</td>
<td>RL</td>
<td>43.4</td>
</tr>
<tr>
<td>5</td>
<td>RR</td>
<td>28.9</td>
</tr>
<tr>
<td>6</td>
<td>RL</td>
<td>50.0</td>
</tr>
<tr>
<td>6</td>
<td>RR</td>
<td>39.3</td>
</tr>
<tr>
<td>7</td>
<td>RL</td>
<td>23.5</td>
</tr>
<tr>
<td>7</td>
<td>RR</td>
<td>36.7</td>
</tr>
<tr>
<td>8</td>
<td>RL</td>
<td>32.3</td>
</tr>
<tr>
<td>8</td>
<td>RR</td>
<td>18.9</td>
</tr>
<tr>
<td>9</td>
<td>na</td>
<td>29.2</td>
</tr>
<tr>
<td>10</td>
<td>na</td>
<td>34.5</td>
</tr>
<tr>
<td>11</td>
<td>na</td>
<td>36.1</td>
</tr>
</tbody>
</table>
Table 4.4 Confusion matrix of feature 6-RL, PCC = 50%.

<table>
<thead>
<tr>
<th>F6-RL</th>
<th>Class 1 observed</th>
<th>Class 2 observed</th>
<th>Class 3 observed</th>
<th>Class 4 observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 Truth</td>
<td>56.6</td>
<td>0</td>
<td>31.9</td>
<td>11.5</td>
</tr>
<tr>
<td>Class 2 Truth</td>
<td>30.4</td>
<td>0</td>
<td>36.8</td>
<td>32.8</td>
</tr>
<tr>
<td>Class 3 Truth</td>
<td>13.2</td>
<td>0</td>
<td>61.2</td>
<td>25.6</td>
</tr>
<tr>
<td>Class 4 Truth</td>
<td>0.9</td>
<td>0</td>
<td>22.4</td>
<td>76.7</td>
</tr>
</tbody>
</table>

From the histograms, it was determined that feature 3-RL separated class 2 from classes 3 and 4 while feature 5-RL showed a level of separation between classes 1 and 2. Selecting these features in addition to feature 6-RL, we chose to classify and obtained the results shown in Table 4.5. The results for all classes are now biased towards class 1, resulting in a lower overall PCC of 37.2%. The addition of the two features degraded our results from the single-feature case with feature 6-RL. We remove feature 3 from consideration and examine the results of features 5-RL and 6-RL (shown in Table 4.6). Here the PCC is lower than for either of the features individually (feature 5-RL PCC = 43.4%, feature 6-RL PCC = 50%). “Good” individual features do not necessarily combine to make good pairs of features [22][30]. An exhaustive search of $n$ features results in $2^n-1$ possible combinations of features. With eight pairs of single polarization features and three full polarization features, we have $n = 19$, resulting in 524,287 possible combinations for an exhaustive search to find the best set of features. This is an intractable problem
with classifiers that require a significant amount of training time (such as the SVM). In order to have a manageable problem, we have reduced the number of features to 16 (\(2^{16} - 1 = 65,535\) possible combinations) and chosen to use a minimum-error Bayesian classifier, which allows evaluation of classification efficacy in a “reasonable” time frame. By observing the histograms and confusion matrices, three of the single polarization features (F3-RR, F4-RL and F7-RR) were omitted from the exhaustive search.

Table 4.5 Confusion matrix with 3 features, PCC = 37.2%.

<table>
<thead>
<tr>
<th>F3-RL, F5-RL, F6-RL</th>
<th>Class 1 observed</th>
<th>Class 2 observed</th>
<th>Class 3 observed</th>
<th>Class 4 observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 Truth</td>
<td>94.0</td>
<td>0</td>
<td>0</td>
<td>6.0</td>
</tr>
<tr>
<td>Class 2 Truth</td>
<td>66.7</td>
<td>0</td>
<td>31.0</td>
<td>2.3</td>
</tr>
<tr>
<td>Class 3 Truth</td>
<td>77.5</td>
<td>0</td>
<td>17.7</td>
<td>7.8</td>
</tr>
<tr>
<td>Class 4 Truth</td>
<td>31.3</td>
<td>0</td>
<td>35.7</td>
<td>33.0</td>
</tr>
</tbody>
</table>

Table 4.6 Confusion matrix with 2 features, PCC = 31.3%.

<table>
<thead>
<tr>
<th>F5-RL, F6-RL</th>
<th>Class 1 observed</th>
<th>Class 2 observed</th>
<th>Class 3 observed</th>
<th>Class 4 observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 Truth</td>
<td>40.1</td>
<td>40.1</td>
<td>5.5</td>
<td>14.3</td>
</tr>
<tr>
<td>Class 2 Truth</td>
<td>23.0</td>
<td>49.4</td>
<td>1.2</td>
<td>26.4</td>
</tr>
<tr>
<td>Class 3 Truth</td>
<td>6.2</td>
<td>50.4</td>
<td>20.9</td>
<td>22.5</td>
</tr>
<tr>
<td>Class 4 Truth</td>
<td>0.9</td>
<td>81.9</td>
<td>0.9</td>
<td>16.3</td>
</tr>
</tbody>
</table>
The results from the three best cases of the exhaustive search are shown in Tables 4.7 – 4.9. For the four-class case of ground vehicles of similar size with one-dimensional signal returns, we are able to classify with 70% accuracy with this technique using only eight features. We note that this assumes no prior knowledge about the probabilities of each target in the Bayesian classifier; all targets are assigned a 25% prior probability. Moreover, the true PDFs are unknown; these have been estimated as Gaussian. Five of the features are common to all three best-case feature sets, which raises the question, how well do the five features alone classify this data set? The results (Table 4.10) show a strong bias towards class two and a degradation of the results to no better than guessing with this subset. This example illustrates the difficulty that arises in selecting exemplary features in an ad hoc fashion. If we found a 25% PCC with these five features, we most likely would not continue adding additional features to this set, but would begin a new search for the best features. It is probable that we would miss the best combination of features.
Table 4.7 Confusion matrix for feature set 1, PCC = 70.9%.

<table>
<thead>
<tr>
<th>F3-RL, F5-RL, F6-RL, F7-RL, F8-RL, F8-RR, F9, F11</th>
<th>Class 1 observed</th>
<th>Class 2 observed</th>
<th>Class 3 observed</th>
<th>Class 4 observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 Truth</td>
<td>81.3</td>
<td>9.3</td>
<td>8.8</td>
<td>0.6</td>
</tr>
<tr>
<td>Class 2 Truth</td>
<td>35.1</td>
<td>61.5</td>
<td>3.4</td>
<td>0</td>
</tr>
<tr>
<td>Class 3 Truth</td>
<td>14.7</td>
<td>6.2</td>
<td>64.4</td>
<td>14.7</td>
</tr>
<tr>
<td>Class 4 Truth</td>
<td>13.2</td>
<td>7.9</td>
<td>17.2</td>
<td>73.7</td>
</tr>
</tbody>
</table>

Table 4.8 Confusion matrix for feature set 2, PCC = 70.1%.

<table>
<thead>
<tr>
<th>F3-RL, F5-RL, F5-RR, F6-RL, F6-RR, F7-RL, F8-RR, F11</th>
<th>Class 1 observed</th>
<th>Class 2 observed</th>
<th>Class 3 observed</th>
<th>Class 4 observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 Truth</td>
<td>90.6</td>
<td>3.9</td>
<td>3.9</td>
<td>1.6</td>
</tr>
<tr>
<td>Class 2 Truth</td>
<td>51.1</td>
<td>42.0</td>
<td>5.2</td>
<td>1.7</td>
</tr>
<tr>
<td>Class 3 Truth</td>
<td>16.3</td>
<td>4.6</td>
<td>66.7</td>
<td>12.4</td>
</tr>
<tr>
<td>Class 4 Truth</td>
<td>2.2</td>
<td>3.5</td>
<td>17.2</td>
<td>77.1</td>
</tr>
</tbody>
</table>
Table 4.9 Confusion matrix for feature set 3, PCC = 70.1%.

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Class 1 observed</th>
<th>Class 2 observed</th>
<th>Class 3 observed</th>
<th>Class 4 observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 True</td>
<td>68.1</td>
<td>16.0</td>
<td>14.8</td>
<td>1.1</td>
</tr>
<tr>
<td>Class 2 True</td>
<td>30.5</td>
<td>62.6</td>
<td>5.8</td>
<td>1.1</td>
</tr>
<tr>
<td>Class 3 True</td>
<td>7.0</td>
<td>9.3</td>
<td>69.0</td>
<td>14.7</td>
</tr>
<tr>
<td>Class 4 True</td>
<td>0.4</td>
<td>4.9</td>
<td>16.7</td>
<td>78.0</td>
</tr>
</tbody>
</table>

Table 4.10 Confusion matrix for common features among best subsets of all features. PCC = 25.0%.

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Class 1 observed</th>
<th>Class 2 observed</th>
<th>Class 3 observed</th>
<th>Class 4 observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 True</td>
<td>10.4</td>
<td>73.1</td>
<td>4.4</td>
<td>12.1</td>
</tr>
<tr>
<td>Class 2 True</td>
<td>1.2</td>
<td>89.0</td>
<td>0.6</td>
<td>0.2</td>
</tr>
<tr>
<td>Class 3 True</td>
<td>21.6</td>
<td>64.3</td>
<td>3.0</td>
<td>10.1</td>
</tr>
<tr>
<td>Class 4 True</td>
<td>20.3</td>
<td>79.7</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

The features resulting in the best classification performance include those derived with single polarimetric data as well as two features that utilize full polarization data. The features selected were specific to this data set, and a new exhaus-
tive search would be required for a new data set. When we group similar classes together (classes one and two become class A and classes three and four become class B), the classification performance improves to 91%. Classes one and two are similar enough to be confusers of each other as are classes three and four, adding difficulty to the classification process. This is not unlike a situation one encounters in many classification problems. This technique could be used to declare the data as class A or class B, and then additional features could be extracted that are more specific to the subclasses.

In many cases, the number of features renders an exhaustive search intractable. This, in addition to the requirement that an expert select data-specific features, has led to our search for an automated feature selection technique. We begin the background for that in the next chapter.
CHAPTER 5

WAVELET BASIS TRANSFORMATION

We seek to project our signal space to a more informative subspace via a basis transformation. Infinitely many possible bases exist that meet the basis transformation criteria of independence and spanning the space. A powerful framework for signal analysis evolves due to the multi-resolution basis transformation properties provided by the wavelet transform. The pyramidal implementation of the discrete wavelet transform provides a computationally efficient method of transformation into the wavelet space. Due to the over-complete or redundant nature of the discrete wavelet packet transform, an input signal can be decomposed onto a multitude of possible bases [28][45]. It is this redundancy that enabled the introduction of best-basis algorithms [19][20][65]. These best-basis algorithms employ cost functions as discriminants to prune the wavelet packet tree to a complete basis in a manner that is tailored to a particular application. Inherent in the wavelet transformation is reduction of dimensionality with information concentrated in a small number of wavelet coefficients, making it particularly attractive for data compression. Moreover, the multi-resolution capabilities of the transform allow for more effective spectral analysis of signals with high-frequency transients than the more common method of spectral analysis using the Fourier transform.
For signals that are localized in extent, decomposition of the signal into an orthonormal library of time-frequency bases results in localization that has demonstrable benefits [18]. Wavelet decomposition provides this localization while spanning the original space. Wavelet orthonormal bases tend to realign and concentrate signal energy, thereby allowing for the desired signal characterization and representation in a reduced feature space [8]. In this chapter, we provide the background for projection into a wavelet packet space. We summarize wavelet and wavelet packet decomposition, the latter being our approach for transformation to feature space.

5.1 Wavelet-Based Approach to Projection to New Basis

The use of the wavelet transform has been motivated by the desire to have adaptive capabilities in the characterization of signals. From Figure 5.1, it is clear that, for general applications, finer time resolution is required for analysis of high-frequency signals, while lower frequency signals can often be considered constant over short time intervals, so that the time resolution need not be so fine. The frequency localization of the wavelet transform is controlled by the scale or frequency parameter. Figure 5.2 shows two sample Heisenberg rectangles for a wavelet transform. Note that this is an example of the Heisenberg uncertainty principle; fixing the value of one parameter (\(\sigma_m\), frequency detail, or \(\sigma_t\), time or space detail) completely determines the value of the second parameter, thus restricting the
time/frequency energy distribution since the area of the Heisenberg box remains fixed [45]. The multi-resolution capabilities of this transform are apparent; the area remains constant while the time and frequency resolution changes for different scales, $s$. A large value of $s$ results in low time resolution with high frequency resolution while decreasing $s$ gives greater time resolution at the expense of frequency resolution.

Figure 5.1 Sinusoids requiring analysis at different time resolution when present as transients in the signal of interest.
5.2 Wavelet Transforms

We present a brief introduction to wavelet transform concepts. Many excellent references are available that provide comprehensive treatment of the topic. Among those are [28][29][45] and [73]. Several key concepts are presented here. The wavelet transform is an inner product of a signal with a kernel function resulting in a time-frequency representation of the input signal. A wavelet kernel ($\psi$) is energy preserving and satisfies the property that it have zero average value:

$$\int_{-\infty}^{\infty} \psi(t) dt = 0$$  \hspace{1cm} (5.1)

The wavelet transform requires selection of a kernel function. Examples of wavelet
transform kernels include the Beylkin, Coiflet, and Daubechies. Beylkin filters place the roots for the frequency response close to the Nyquist frequency on the real axis. This has the effect of maximizing the power in the desired band. Coiflets provide the same number of vanishing moments in both the wavelet and the scaling functions and result in a high level of compressibility. The Daubechies wavelets maximize the smoothness of the scaling function with a minimum size for a given number of vanishing moments [45], which results in high compressibility for piecewise linear signals. Selection of the wavelet kernel can be tailored to the particular application.

The wavelet transformation defines local time-frequency density with a measure of the localized energy for each Heisenberg rectangle in the time-frequency plane. In order to cover the plane, the wavelet atom is dilated or scaled by $s$ and translated by $u$:

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)$$  \hspace{1cm} (5.2)$$

The Fourier transform of the wavelet exposes the capacity of the wavelet for scaling the frequency information with the parameter $s$:

$$\hat{\psi}_{u,s}(\omega) = e^{-i\omega u} \sqrt{s}\hat{\psi}(s\omega)$$  \hspace{1cm} (5.3)$$

Note that the area resolved by the scaling is a constant; the time-honored trade-off
between time and frequency holds for wavelet transforms as we adaptively “zoom” in on time and frequency information. The wavelet transform of the signal, \( f(t) \), is given by the correlation of the signal with a wavelet kernel:

\[
Wf(u,s) = \int_{-\infty}^{\infty} f(t) \frac{1}{\sqrt{s}} \psi^* \left( \frac{t-u}{s} \right) dt
\]  

(5.4)

Of particular interest is the discrete wavelet transform with an orthonormal basis, which provides the framework for our basis transformation. Mallat [45] introduced a pyramidal multi-resolution algorithm for discrete wavelet transforms that, like the fast Fourier transform, is a fast, linear operation making the wavelet transform practical for implementation. Computation of the discrete wavelet transform is based on a discrete wavelet kernel,

\[
\psi_j[n] = \frac{1}{\sqrt{a^j}} \psi \left( \frac{n}{a^j} \right)
\]

(5.5)

and a scaling function,

\[
\phi_n[n] = \frac{1}{\sqrt{a^j}} \phi \left( \frac{n}{a^j} \right)
\]

(5.6)

where \( a^j \) represents the scale parameter for level \( j \).

Discrete wavelet transformation is applied via the following circular convolution:
\begin{align}
Wf[n,a^j] &= \sum_{m=0}^{N-1} f[m] \psi^*[m-n] \\
Lf[n,a^j] &= \sum_{m=0}^{N-1} f[m] \phi^*[m-n]
\end{align}

(5.7) (5.8)

where $Wf[n,a^j]$ represents the wavelet transform (high-pass) operation and $Lf[n,a^j]$ represents the scaling (low-pass) operation.

Actual implementation of the discrete wavelet transform is done via quadrature mirror filter (QMF) pairs that form an orthogonal basis, satisfying the following constraints:

\begin{align}
\left| \hat{g}(\omega) \right|^2 + \left| \hat{g}(\omega + \pi) \right|^2 &= 2 \\
\hat{g}(\omega) \hat{h}^*(\omega) + \hat{g}(\omega + \pi) \hat{h}^*(\omega + \pi) &= 0
\end{align}

(5.9) (5.10)

The QMFs are generated for the wavelet selected, and then used to create filter banks. QMFs decompose a discrete signal into separate frequency bands; a pair of these filter banks can be thought of as a high-pass filter (HPF), $g$, and a low-pass filter (LPF), $h$. We note that for perfect reconstruction of the original signal, two additional constraints are imposed on the design of the QMFs:
\[ \begin{align*} 
\hat{h}^* (\omega + \pi) \hat{h}(\omega) + \hat{g}^* (\omega + \pi) \hat{g}(\omega) &= 0 \\
(5.11) \\
\hat{h}^* (\omega) \hat{h}(\omega) + \hat{g}^* (\omega) \hat{g}(\omega) &= 2 \\
(5.12) 
\end{align*} \]

The frequency-domain relationship between \( g[n] \) and \( h[n] \) is given by

\[ \hat{g}(\omega) = e^{-i\omega \hat{h}^* (\omega + \pi)} \]  
(5.13)

The filter coefficients are determined by

\[ h[n] = \left( \frac{1}{\sqrt{2}} \phi \left( \frac{t}{2} \right) \phi(t - n) \right) \]  
(5.14)

\[ g[n] = \left( \frac{1}{\sqrt{2}} \psi \left( \frac{t}{2} \right) \phi(t - n) \right) \]  
(5.15)

Note that these filters are dilated by 2. In addition, by taking the inverse Fourier transform of (5.13), we see a simple relationship between the coefficients of (5.14) and (5.15).

\[ g[n] = (-1)^{i\pi} h[1 - n] \]  
(5.16)
The one-dimensional pyramidal algorithm of the discrete wavelet transformation, illustrated in Figure 5.3, is shown as a sequence of filtering and down-sampling operations. Equation (5.14) is analogous to the impulse response of the LPF (block H) while (5.15) can be thought of as the impulse response of the HPF (block G). Note that the output of block G, commonly referred to as the "details," is retained as the wavelet transform coefficients for that level of the decomposition, while the output of block H is passed on to the next octave of filter banks for further processing, iterating the decomposition and subsampling process until the desired level of decomposition is reached. The result is an orthogonal subspace decomposition of the data vector with no redundancy.

Figure 5.3 Discrete wavelet decomposition. Output of the HPF (G) is retained. Output of the LPF (H) is further decomposed.
The discrete wavelet packet transform is a generalization of the discrete wavelet transform, which partitions the output of all filters to the desired octave level as shown in Figure 5.4. Both the details and the smoothed portions (output of the LPF) of the data vector are passed through the pair of QMFs at each level, resulting in an overcomplete or redundant representation of the input signal. There are on the order of $2^{n/2}$ possible orthogonal bases for a wavelet packet decomposition, where $n$ is the input signal length.

Figure 5.5 shows the wavelet coefficients for a wavelet decomposition of a one-dimensional signal. The coefficients (down sampled output from filtering the LPF signal of the previous level) are shown. To have a complete basis, the high-pass (details) coefficients would be used as well as the low-pass output at the last level of decomposition. The basis selection technique is pre-defined. Figure 5.6 shows the wavelet packet coefficients for the same signal that produces an over-complete frame where each level contains a complete basis of the original space.
In fact, from this wavelet packet decomposition, there are more than $2^{64}$ possible complete bases that could be selected. It is the redundancy of the wavelet packet decomposition that allows application-specific selection of a complete basis and necessitates pruning the wavelet packet tree via a discriminant or cost function to obtain the desired representation for a particular application. Various cost functions have been utilized as the discriminant to search for the best basis. Our research results in an approach that provides an alternative to local discriminant basis (LDB) for selection of a basis for representation of the data for classification with a reduced set of coefficients. Both LDB and our new approach are addressed in the next chapter.
Figure 5.5  Image of wavelet coefficients for decomposition of a one-dimensional example signal. Fine spatial information at top level, fine frequency information at bottom level. Black indicates no additional filtering of the output of the previous level filter.

Figure 5.6  Image of wavelet packet coefficients for decomposition of a one-dimensional example signal. Fine spatial information at top level, fine frequency information at bottom level.
CHAPTER 6

BASIS SELECTION

We begin this section with an explanation of the manner in which we define our terminology. A basis is a set of vectors that span the original data space. Bases refers to multiple basis sets, which together are over-complete (redundant); a basis is complete (non-redundant representation of the signal that entirely describes the original signal). The basis can be pruned to be under-complete containing only a partial description of the signal. The term node is used to refer to a dyadic subspace (down sampled after filtering). For instance, for a signal of length sixteen, the first dyadic level of decomposition contains two nodes, each of length eight. At the next level, there are four nodes of length four, while level three contains eight nodes of length two.

Utilizing wavelet packet decomposition provides a multitude of possible complete orthonormal bases. The over-determined nature of the wavelet packet decomposition requires adaptive pruning of the wavelet packet tree to determine the best-basis [14][20][63][65] representation for a particular signal. Representation of the signal in the wavelet packet domain results from projections onto new coordinates; the goal is to determine from these redundant bases the best basis. The ideal best basis is dependent upon the application. For example, for data compression, we might seek a basis that maximizes the information content of a few coeffi-
cients and minimizes the energy, or information content, in the remaining coefficients thereby reducing the dimensionality of the problem. For a classification problem, we would desire a basis that provides improved separability between the classes as the primary objective, with dimensionality reduction a close second. Automating basis selection for classification is the subject of this chapter.

A novel approach to basis selection was introduced by Coifman and Wick-erhauser [20]; their best-basis technique provided an automated approach to selecting an orthonormal basis to represent the data for data compression. This was extended to the classification setting by Coifman and Saito [62][63][65] with a technique that they refer to as LDB.

In this chapter, we explore best-basis approaches addressed in our research. These include the original LDB technique and our alternative basis selection technique based on a fitness function using a statistical test, the Kolmogorov-Smirnov (KS) test, to measure the distributional differences between class-conditional empirical cumulative distribution functions (ECDFs) in determining efficacy of basis selection in a multi-class setting. We briefly discuss Saito's extension to LDB [64], which uses the Kullback-Leibler (KL) [39] divergence (relative entropy) as the test statistic and applies it to estimates of the probability density function.
6.1 Local Discriminant Basis

The LDB method is a powerful algorithmic framework that was originally developed by Coifman and Saito in 1994 [19][65] as a best-basis technique for analyzing object classification problems. Using LDB as a feature extraction algorithm allows for selection of a basis and the resulting LDB coefficients or features. From the entire set of LDB features, a greatly reduced subset can often be used for classification. We present an overview of the LDB technique and refer the reader to [65] for a comprehensive treatment of the subject.

We begin by cataloging a basis library consisting of functions localized in the time-frequency plane. A dictionary of bases is selected from the library to decompose each class into over-complete bases, and an energy map is formed for each class. The class energy maps are, in essence, averages of the wavelet packet decompositions of all the signals in the class. Class energy maps are used in a pair-wise fashion to prune to an orthogonal basis by applying a distance measure to estimate maximal class separation. LDB uses an additive pair-wise class discriminant, \( D(p',p^k) \), such as relative entropy:

\[
D(p', p^k) = \sum_{i=1}^{n} p'_i \log \frac{p'_i}{p^k_i}
\]

where \( p' \) and \( p^k \) represent the wavelet packet coefficients for separate classes and \( n \) is the length of the class sequences in the current node being scored.
We calculate the overall node value, \( D(p) \) using an L1 vector normalization scheme of the class pair-wise values:

\[
D(p) = \sum_{j=1}^{m-1} \sum_{k=j+1}^{m} D(p^j, p^k)
\]

(6.2)

where \( m \) is the number of classes. It is this vector of scores that is used to prune the wavelet packets to a complete basis.

Coefficients or features obtained from mapping the data into the orthonormal basis are ordered based on the separability of classes. The entire set of features can be used as inputs to the classifier; however, using a subset of features reduces the dimensionality and often improves the performance by reducing the noise in the signal. This algorithm has been applied to a range of classification challenges \[11\][15][33][64][66][67] (see also [17]). We summarize the original LDB algorithm with the following steps and follow with a step-by-step example of both LDB and our new KS technique in Section 6.3.

**Step 0:** Select a wavelet family based on the desired characteristics (see Section 5.2).

**Step 1:** Given a training set of \( m \) classes, expand the training signals, \( x_i^j \), (\( i \) indexed over all training samples for class \( j \)) onto a library of re-
dundant orthonormal wavelet packets, \( w_{s,f,t} \), indexed by scale, frequency, and time/position (i.e. calculate the wavelet packets of the input signals).

**Step 2:** Compute the time-frequency energy map of class \( j \), denoted as \( \Gamma^j(s,f,t) \), and specified by the normalized sum of the wavelet packets for class \( j \):

\[
\Gamma^j(s,f,t) = \frac{\sum_{t=1}^{N_j} \langle w_{s,f,t}, x_i \rangle}{\sum_{t=1}^{N_j} \| x_i \|^2}
\]

(6.3)

where \( N_j \) is the number of training samples of class \( j \) and \( \langle \rangle \) represents the inner product operation. Note that it is the inner product of the input signal with the wavelet packet vectors that produces the wavelet packets of the input signal. We sum the node values at indices \((s,f)\) over all \( t \) in the node for an overall node score, \( \Gamma^j(s,f) \) (score indexed by scale and frequency band at that scale). For example, for a class with 360 training samples of length 32, the signals are expanded dyadically to five levels and then averaged by class. The dimension of the wavelet packets would be \( 360 \times 32 \times 5 \) for all training samples of this class; once the packets are averaged by class, the dimension is \( 1 \times 32 \times 5 \) for each class. If we consider the energy map,
Figure 6.1 Wavelet packet decomposition of signal of length 32. Nodes at each scale are outlined in white. Dyadic decomposition becomes finer with less coefficients in each node as the level of decomposition increases. There are multiple frequency bands at each scale.

<table>
<thead>
<tr>
<th>Level Number (index $s$)</th>
<th>Number of frequency bands (index $f$)</th>
<th>Number of points summed to form the score at each frequency band</th>
<th>Energy map score dimension by level, $1 \times$ number of nodes ($2^s$) for a single class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>16</td>
<td>$1 \times 2$</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>8</td>
<td>$1 \times 4$</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>4</td>
<td>$1 \times 8$</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>2</td>
<td>$1 \times 16$</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>1</td>
<td>$1 \times 32$</td>
</tr>
</tbody>
</table>

$\Gamma^s(s,f)$, and look at the indices at each level, we see the packet decomposition and frequency band splits by level in Figure 6.1 and Table 6.1.
Step 3: Compute an overall class energy map node score for each subspace using an L1 norm:

$$
\Gamma^j(s, f; \cdot) = \sum_{t=1}^{n/2^S} D(\Gamma^j(s, f, t))
$$

with \( f = 1, ..., 2^s \), \( s = S, ..., 1 \), where \( S \) is the maximum scale (depth) of the library tree, \( 2^s \) is the number of nodes at the current scale \( s \), and \( n \) is the signal length. Compute the composite score as a summation of the pairwise comparison scores. This is a discriminant function specified on the energy maps for each dyadic subspace to determine an L1 based-discriminant score, \( \Delta_{s,f} \), for each subspace,

$$
\Delta_{s,f} = \sum_{j=1}^{m-1} \sum_{k=j+1}^{m} D(\Gamma^j(s, f; \cdot), \Gamma^k(s, f; \cdot))
$$

again using \( m \) as the number of classes.

Step 4: Determine the best basis or spanning space by comparing the \( \Delta_{s,f} \) with \( \Delta_{s+1,2f} + \Delta_{s+1,2f+1} \) (\( \Delta_{s,f} \)'s subspace tree descendants). By starting at the highest scale (level 5 of our previous example) and moving to the lowest (level 1), select the subspace representatives with the maximum overall score. This results in the best basis.
Step 5: Order the basis functions (wavelet packet vectors), \( w_{s,j,n} \), of the selected best basis by their classification efficacy. Select the best \( k \ll n \) features to provide to a classifier.

Two issues of concern surface with the LDB approach. The first occurs in the formation of the energy maps in LDB. This process replaces the distributional characteristics of the training set with an estimate of the overall class average power projected by the training data onto the bases. Energy map comparisons between classes therefore consist of comparisons of the differences in average power or a comparison between the class means of the wavelet packet transformed samples. Higher moments are unavailable for exploitation; furthermore, the mean can be susceptible to training outliers. The second concern arises in the scoring for the basis selection process. In considering the discrimination method for pruning the LDB basis, we note that the L1 summation of the pair-wise discriminants can result in a large overall score that may not necessarily result in the best basis since the summation removes knowledge of the individual pair-wise scores. A case where there are many small but not insignificant pair-wise values could result in a larger overall score than one in which there is a single large pair-wise discriminant and many insignificant terms that well separate two of the classes. This could lead to selection of a suboptimal basis. An example of this is given in the next section.
where the new normalization techniques are discussed. Our new basis selection algorithm addresses both of these issues.

6.2 Kolmogorov-Smirnov Best-Basis Selection

This section introduces the best-basis approach investigated in our research and defines the algorithmic differences between it and LDB [72]. Our goal is to define a basis selection approach that maximizes the discriminatory information content in a "small" number of coefficients, attempting to increase our ability to distinguish among the classes while simultaneously reducing dimensionality. Additionally, we wish to address the two issues that arose with the LDB algorithm. First, we define an alternative discriminant function for measuring class pair-wise classification efficacy that utilizes the empirical cumulative distribution function, ECDF, of the wavelet packets. Our technique removes the averaging effect that occurs with LDB and permits retention of individual signal contributions resulting in the ability to exploit higher order statistics. The second issue, that of converting pair-wise scores to more meaningful overall scores, is addressed via scoring matrices and the identification of alternative multi-class normalizations.

Desirable characteristics for any discriminant function include the minimization of the required assumptions for its use while maximizing the efficacy of the test. The two-sample KS test [38][77] provides a model-free (distribution-free) approach to measure the deviations between two samples. The KS test is a nonpara-
metric statistical test with a two-sample variant that quantifies whether two samples were produced from the same underlying probability density, \( f(x) \). We use the critical value produced by the KS test in a manner that estimates the differences in two distributions. The KS test identifies when samples come from a common distribution; failing to reject that the two distributions are from a common distribution serves to allow the conclusion that samples are consistent with a single probability density, rather than to offer proof [77]. In using the KS test to select the basis, we identify the largest KS values (those indicating the least likelihood that the test samples are from the same distribution). The ECDF of a sample, \( \hat{F}(x) \), can be defined as the proportion of samples less than or equal to \( x \), and is computed from the respective samples via

\[
\hat{F}(x) = \frac{1}{n} \sum_{j=1}^{n} u(x - x_j)
\]  
(6.6)

where \( n \) is the total number of samples, and

\[
u(x - x_j) = \begin{cases} 
1 & \text{if } x - x_j \geq 0 \\
0 & \text{else}
\end{cases}
\]
Note that the ECDF is a scaled step function, individually defined at each sample location, where all values take on multiples of $1/n$. The KS test statistic, $D$, is the maximum absolute deviation between $\hat{F}_1(x)$ and $\hat{F}_2(x)$.

$$D(\hat{F}_1, \hat{F}_2) = \max_x \left( |\hat{F}_1(x) - \hat{F}_2(x)| \right)$$  \hfill (6.7)

See Figure 6.2. This measure of separation makes no assumptions concerning the form of the underlying density function and is robust to outliers. For example, Figure 6.3 (a) and (b) show ECDFs of two samples. The solid line distribution is the same in both (a) and (b). The dashed distribution in (a) has a single outlier while the dashed distribution in (b) is identical with the exception of the outlier. The KS test statistic in both (a) and (b) is 0.5; the L1 norm for the outlier example, (a),

Figure 6.2 KS test statistic, $D$, maximum distance between ECDFs of two distributions.
is 2.5 while that for (b), with no outliers, is 1.7. Using the KS test statistic caused the outlier to have no impact on the overall test.

The second issue addressed by our approach is that of the loss of pair-wise discriminant information in LDB. Through the utilization of a score matrix, we retain the pair-wise class separability scores. Forming an $m \times m$ cube of discriminant matrix statistics, $D_k$, where $m$ is again the number of classes, we have some liberty in determining a mapping from the pair-wise scores to an overall score. A natural general formulation consists of the p-norm, $L_p$, given by

$$\|D\|_p = \left( \sum_{k=1}^{m(m-1)/2} |D_k|^p \right)^{1/p} \quad (6.8)$$
This formula allows the investigation of alternative normalizations. Popular values for $p$ include 1, 2, and $\infty$ (sup norm), which is given as:

$$\|D\|_\infty = \max_k |D_k|$$

(6.9)

where $|D_k|$ is the standard definition, the magnitude of $D_k$. We investigate the use of these $p$-norms and also a minimax normalization given by $\|D\|_{\text{min}} = \min_k |D_k|$.

These norms are discussed briefly.

The overall score for each node with the L1 norm (Manhattan or city block distance) results from the summation of the individual pair-wise discriminants. This is the approach used in LDB and can cause loss of information regarding good separability for a single pair of classes. The L2 norm is the familiar Euclidean distance, or the square root of the sum of the squares of the pair-wise scores for each node. The sup norm assigns the maximum or best pair-wise separation score to the overall score for each node. The final normalization scheme under investigation for overall score selection is the minimax norm that selects the minimum pair-wise value from the score matrix as the node score. The net effect of this normalization is to minimize the risk associated with selection of the node by selecting the minimum pair-wise discriminant value. The features are later ranked by maximum values, hence the name minimax. Although choosing the minimum might seem counterintuitive at first glance, it allows the user to select features with the best-worst-case efficacy, thereby minimizing the risk.
An example of the application of these norms follows for two sample scoring matrices, $S(1)$ and $S(2)$, with a three-class case of pair-wise class separation scores.

$$S(1) = \begin{bmatrix} 0 & 0.3 & 0.1 \\ 0.3 & 0 & 0.7 \\ 0.1 & 0.7 & 0 \end{bmatrix}, \quad S(2) = \begin{bmatrix} 0 & 0.4 & 0.5 \\ 0.4 & 0 & 0.3 \\ 0.5 & 0.3 & 0 \end{bmatrix}$$

The overall node scores for each of these examples are shown in Table 6.2 for the scoring techniques discussed. Due to the symmetry of the matrix, the scores below the diagonal are not included in the overall score.

<table>
<thead>
<tr>
<th>Example</th>
<th>$L1$ score</th>
<th>$L2$ score</th>
<th>Min score</th>
<th>Sup score</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S(1)$</td>
<td>1.1</td>
<td>0.77</td>
<td>0.1</td>
<td>0.7</td>
</tr>
<tr>
<td>$S(2)$</td>
<td>1.2</td>
<td>0.71</td>
<td>0.3</td>
<td>0.5</td>
</tr>
</tbody>
</table>

As the table clearly shows, the overall score for $S(1)$ varies between 0.1 and 1.1 while that for $S(2)$ varies from 0.3 to 1.2. Depending on which norm is used for scoring, the overall score results in selection of a different basis. For instance, using an $L1$ norm, node $S(2)$ would be selected, while a sup norm would result in the selection of $S(1)$. This can be exploited to result in better classification performance by permitting the user to inject knowledge about the class separation problem for specific data sets. For instance, a sup norm might be selected so that the basis favors single superior pair-wise class separations at the expense of overall class
separation. A minimax can be selected to minimize the overall risk of misclassifica-
tion for a case where all classes are difficult to separate.

Incorporating the KS test as our discriminant function for estimation of pair-wise class separation and the alternative node scoring scheme, we replace the original LDB algorithm with the following:

**Step 0:** Select a wavelet family based on the desired characteristics. (see Section 5.2).

**Step 1:** Given a training set of \( m \) classes, expand the training signals, \( x_i^j \) (indexed over all training samples for class \( j \)) onto a library of redundant orthonormal wavelet packets, \( w_{sfb} \), indexed by scale, frequency, and time/position. Calculate the wavelet packets, \( a' \), where \( a \) are determined for each training sample of class \( j \).

\[
a'^j(s,f,t) = \frac{\langle w_{sfb} , x_i^j \rangle}{\|x_i^j\|} \tag{6.10}
\]

At this level, we do not sum the node \( s,f \) over \( t \) for an overall node score for the frequency subspace as we did with LDB. The score is indexed by \( s \) (scale), \( f \) (frequency band at the particular scale, \( s \)), and \( t \) (index into the frequency subspace). For example, for a single class with 360 training
samples of length 32, the signal can be expanded dyadically to five levels. The dimension of $a'_{i}$ would be $360 \times 32 \times 5$ for all training samples. If we consider a single training sample of $32 \times 5$ and look at the indices at each level, we see the following by level.

Table 6.3 Dimension of wavelet packets.

<table>
<thead>
<tr>
<th>Level Number (index $s$)</th>
<th>Number of frequency bands (index $f$)</th>
<th>Number of points in each frequency band (index $t$)</th>
<th>Score dimension by level (1 x index) for a single class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>16</td>
<td>$1 \times 32$</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>8</td>
<td>$1 \times 32$</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>4</td>
<td>$1 \times 32$</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>2</td>
<td>$1 \times 32$</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>1</td>
<td>$1 \times 32$</td>
</tr>
</tbody>
</table>

**Step 2:** Compute the K-S test statistic using (6.7) across all wavelet packet indices $(s, f, t)$. For each $s, f, t$ (each point in the wavelet packet frame), form an $m \times m$ score matrix of class pair-wise discriminant measures:

$$D^{jk}(s, f, t) = D\left(\hat{F}(a^{j}(s, f, t)), \hat{F}(a^{k}(s, f, t))\right) \quad (6.11)$$

where $j$ and $k$ are class indices. Select a norm technique to use on the $m \times m$ score matrices to determine a score for each point in the wavelet packet frame. For LDB, the norm applied here was L1; any of the new norm techniques can be applied. This results in an $L \times n$ score frame ($L$ is the dyadic level of the data, $L = \log_{2}(n)$, $n =$ signal length). Select a norm technique to determine the overall score for each node or frequency band. These scores
are represented as $A_{s,f}$ where $s$ is the scale and $f$ is the frequency band at that scale as shown in Figure 6.1.

*Step 3:* Prune the node-based scores to a complete basis. This is accomplished by comparing the $A_{s,f}$ with $A_{s+1,2f} + A_{s+1,2f+1}$. We again start at the highest scale ($L = \log_2(n)$) and move to the lowest ($L = 1$) in selecting the basis with the maximum overall node score.

*Step 4:* Order the basis functions, $w_{s,f,n}$ of the selected basis by their classification efficacy (coefficient magnitude). Select the best $k<<n$ features to provide to a classifier.

By using the KS approach, we are able to exploit the distributional characteristics of the individual sample projections, *i.e.* $\langle w_{s,f,n} x \rangle$, since these are not averaged in *Step 1* as they are in the energy map approach of LDB’s *Step 1*. This approach requires more memory than the LDB approach, since a wavelet packet decomposition is stored for each signal. The energy map approach of LDB results in cumulative summations of wavelet packets ($\Sigma \langle w_{s,f,n} x |^2 \rangle$) by class, hence requiring the retention in memory of only one wavelet packet decomposition per class. Finally, we are able to exploit the ability to look at pair-wise discriminants between classes rather than simply considering a summation of the scores.
6.3 Best-Basis Algorithm Example

We provide an example with triangular waveforms, which was examined in [65]. We investigate this three class case to show the basis selection process for LDB and KS. The signals were generated by the following

\[
x'(i) = uh_1(i) + (1-u)h_2(i) + \epsilon(i)
\]

\[
x^2(i) = uh_1(i) + (1-u)h_3(i) + \epsilon(i)
\]

\[
x^3(i) = uh_2(i) + (1-u)h_3(i) + \epsilon(i)
\]

letting \( i = 1, \ldots, 32 \), (index into the sample/signal) and defining

\[
h_1(i) = \max(6 - |i - 7|, 0),
\]

\[
h_2(i) = h_1(i - 8)
\]

\[
h_3(i) = h_1(i - 4)
\]

We let \( u \) be a uniform random variable on \((0,1)\), and \( \epsilon \) is a normally distributed random variable with zero mean and unit variance. Figure 6.4 shows examples of the signals, \( x' \) (class one), \( x^2 \) (class two), and \( x^3 \) (class three). We generate 100 random signals for each class. These signals are used to show the process for the wavelet packet best-basis selection approaches.
We begin with Step 0 and select a wavelet family. We select the Daubechies 4 wavelet, which maximizes the smoothness of the scaling function with a minimum size wavelet. The filter coefficients of the LPF ($h[n]$) are [-0.1294 0.2241 0.8365 0.483]. The corresponding HPF ($g[n]$) coefficients can be found from (5.16) and are [0.483 -0.8365 0.2241 0.1294]. This step is the same for both LDB and KS.

In Step 1, we compute the wavelet packets. We begin by convolving the input signal with the wavelet filters and downsampling:

$$s = a \ast h[2n]$$

$$d = a \ast g[2n]$$
where \( a \) is the original signal, \( s \) (smoothed portion) is the output of the LPF, \( d \) (detail portion) is the output of the HPF, and * denotes convolution. We filter the output of each filter \((s \text{ and } d)\), with both filters (see Figure 5.4) to obtain

\[
\begin{align*}
ss &= s \ast h[2n] \\
sd &= s \ast g[2n] \\
ds &= d \ast h[2n] \\
dd &= d \ast g[n]
\end{align*}
\]

We continue the decomposition until we have the output of the filters of length one. These wavelet packets are organized in an \( k \times n \) table, where \( n \) is the signal length (32 for our example) and \( k \) is the number of dyadic levels for the signal length \( k = \log_2(n) = 5 \). This was shown in Figure 6.1 and is shown again in Figure 6.5 with example frequency bands labeled. Each training sample (signal) is decomposed individually into the wavelet packets resulting in \( N^j \) wavelet packets for class \( j \), where \( N^j \) represents the number of training samples for class \( j \).

When referencing individual coefficients, we refer to them by the filters that were used for their generation and an index into that filter. At level 1, there are 16 coefficients for the output of each filter \((h[n] \text{ and } g[n])\). This decreases dyadically at each level until the last level has one coefficient for each filter output. For example, the first 16 coefficients in level 1 would be referred to by \( s \) and a position
index (1-16), while the last 16 would be referred to by $d$ and a position index (1-16).

For the last level, level 5, the first two coefficients would be $s_{ssss}$ (index 1) and $s_{sssd}$ (index 1). LDB and KS approaches are the same up to this point. They will now be discussed separately, beginning with LDB.

Beginning with Step 2 for LDB, we calculate the energy map, $\Gamma^j(s,f,t)$, for each class using (6.3), which is repeated here for convenience.

\[
\Gamma^j(s,f,t) = \frac{\sum_{i=1}^{N_j} \langle w_{f,b} \cdot x_i \rangle^2}{\sum_{i=1}^{N_j} \| x_i \|^2}
\]

Recall that for this example three class case, there are 100 samples for each class ($N^1 = N^2 = N^3 = 100$), and each sample is of length 32. After this step, we have one $32 \times 5$ class energy map for each class.
For Step 3, we calculate the overall node scores for each frequency band or node for each class using an L1 norm. The relative entropy between pairs of the nodes is calculated from the class overall nodes scores in a pair-wise fashion using (6.1) and then summed for all class pair-wise combinations (equation (6.5)). The overall nodes scores for this data set are given in Table 6.4 with the scores for each node listed in the place of the first coefficient for the node. These are also shown in Figure 6.6 (a). We evaluate the node scores to determine the best basis starting at the highest scale (level 5). For instance, the $\text{ssss}$ frequency band score is 0.06705, while the sum of its two immediate descendants ($\text{sssss}$ and $\text{ssssd}$) is 0.13355 ($0.08667 + 0.04687$) resulting in selection of the descendants at that level. This continues to level 1, with each best score in the frequency band saved for comparison with the scores at the ascendant levels. Once the scores have been compared at all levels, the nodes associated with those maximum values are flagged as the best basis representation for separating the classes. The LDB best basis for these three classes of data is shown in Figure 6.6 (b). Note that the basis selected represents the finest frequency subbands. Finally, the coefficients of the basis are ordered based on their magnitude.
Table 6.4 LDB node scores over all classes.

<table>
<thead>
<tr>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
<th>Level 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.04210</td>
<td>0.08717</td>
<td>0.07006</td>
<td>0.06705</td>
<td>0.08667</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.04687</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.34792</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.31543</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.34376</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.01278</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.01023</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00277</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.01699</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.01669</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.02860</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00652</td>
</tr>
<tr>
<td>0.00255</td>
<td>0.00594</td>
<td>0.00770</td>
<td>0.00628</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00930</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00356</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00682</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00131</td>
</tr>
<tr>
<td>0.00060</td>
<td>0.00181</td>
<td>0.00335</td>
<td>0.00766</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00049</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00569</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00517</td>
</tr>
<tr>
<td>0.00479</td>
<td>0.00451</td>
<td>0.00622</td>
<td>0.00651</td>
<td>0.00774</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00574</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00594</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00687</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00521</td>
</tr>
<tr>
<td>0.00229</td>
<td>0.00262</td>
<td>0.00255</td>
<td>0.00270</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00207</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00038</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00359</td>
</tr>
<tr>
<td>0.00531</td>
<td>0.00521</td>
<td>0.00711</td>
<td>0.00671</td>
<td>0.00823</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00292</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00276</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00308</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00553</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00530</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00473</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00617</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00615</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00679</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.00556</td>
</tr>
</tbody>
</table>
Figure 6.6 (a) LDB node scores over all classes. White indicates largest values. (b) Resulting basis selected. White indicates selected basis.
For the KS best basis, we compute the wavelet packets in the same manner as LDB. We begin the KS discussion at Step 2, after formation of the wavelet packets. We form ECDFs on the wavelet packet decomposed signal on a class-wise basis. For this example, we have 100 training signals decomposed into wavelet packets of dimension $32 \times 5$ resulting in $32 \times 5$ ECDFs with 100 sample points for each class. The KS test statistic is calculated for all pairs of classes and a matrix $(m \times m)$ of scores is formed for each of the $32 \times 5$ points in the wavelet packet. We now select a normalization scheme (L1, L2, minimax or sup). We use the minimax for this example. The $m \times m$ matrix of scores at each point in the $32 \times 5$ score data set is evaluated based on the minimax norm and the minimum of each pair-wise score is selected to represent that point (i.e., $D(s,f,t) = \min(D^k(s,f,t))$). After this evaluation, the $32 \times 5 \times m \times m$ score data set is reduced to an overall score matrix that is $32 \times 5$. A norm technique is selected for evaluation of the scores in each individual node or frequency band to overall node scores. The KS score matrix is given in Table 6.5 with the scores also shown in Figure 6.7 (a). In Table 6.6, we show the node scores with the minimax norm applied (score is left in its original position). The selected basis is given in Figure 6.7 (b). Notice that the basis selected includes more variation in the frequency bands than the basis that was selected for LDB. We will see in Chapter 8, that this provides better classification performance when operating in reduced dimensionality. The coefficients of the basis are ordered based on their magnitude.
Table 6.5  KS individual wavelet packet sample point scores over all classes.

<table>
<thead>
<tr>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
<th>Level 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.10000</td>
<td>0.09000</td>
<td>0.12000</td>
<td>0.14000</td>
<td>0.07000</td>
</tr>
<tr>
<td>0.07000</td>
<td>0.07000</td>
<td>0.08000</td>
<td>0.10000</td>
<td>0.12000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.19000</td>
<td>0.15000</td>
<td>0.07000</td>
<td>0.07000</td>
</tr>
<tr>
<td>0.08000</td>
<td>0.09000</td>
<td>0.10000</td>
<td>0.12000</td>
<td>0.11000</td>
</tr>
<tr>
<td>0.06000</td>
<td>0.07000</td>
<td>0.13000</td>
<td>0.09000</td>
<td>0.15000</td>
</tr>
<tr>
<td>0.17000</td>
<td>0.09000</td>
<td>0.16000</td>
<td>0.14000</td>
<td>0.11000</td>
</tr>
<tr>
<td>0.12000</td>
<td>0.10000</td>
<td>0.12000</td>
<td>0.18000</td>
<td>0.19000</td>
</tr>
<tr>
<td>0.12000</td>
<td>0.09000</td>
<td>0.07000</td>
<td>0.18000</td>
<td>0.12000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.06000</td>
<td>0.09000</td>
<td>0.08000</td>
<td>0.09000</td>
</tr>
<tr>
<td>0.06000</td>
<td>0.13000</td>
<td>0.12000</td>
<td>0.11000</td>
<td>0.12000</td>
</tr>
<tr>
<td>0.11000</td>
<td>0.07000</td>
<td>0.12000</td>
<td>0.08000</td>
<td>0.08000</td>
</tr>
<tr>
<td>0.08000</td>
<td>0.14000</td>
<td>0.09000</td>
<td>0.09000</td>
<td>0.06000</td>
</tr>
<tr>
<td>0.09000</td>
<td>0.13000</td>
<td>0.10000</td>
<td>0.09000</td>
<td>0.08000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.17000</td>
<td>0.12000</td>
<td>0.10000</td>
<td>0.05000</td>
</tr>
<tr>
<td>0.09000</td>
<td>0.08000</td>
<td>0.12000</td>
<td>0.10000</td>
<td>0.12000</td>
</tr>
<tr>
<td>0.09000</td>
<td>0.09000</td>
<td>0.11000</td>
<td>0.11000</td>
<td>0.07000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.11000</td>
<td>0.09000</td>
<td>0.09000</td>
<td>0.08000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.06000</td>
<td>0.12000</td>
<td>0.10000</td>
<td>0.08000</td>
</tr>
<tr>
<td>0.12000</td>
<td>0.21000</td>
<td>0.07000</td>
<td>0.08000</td>
<td>0.07000</td>
</tr>
<tr>
<td>0.08000</td>
<td>0.14000</td>
<td>0.09000</td>
<td>0.08000</td>
<td>0.09000</td>
</tr>
<tr>
<td>0.14000</td>
<td>0.08000</td>
<td>0.10000</td>
<td>0.14000</td>
<td>0.06000</td>
</tr>
<tr>
<td>0.16000</td>
<td>0.07000</td>
<td>0.21000</td>
<td>0.13000</td>
<td>0.22000</td>
</tr>
<tr>
<td>0.09000</td>
<td>0.11000</td>
<td>0.08000</td>
<td>0.19000</td>
<td>0.18000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.13000</td>
<td>0.07000</td>
<td>0.12000</td>
<td>0.12000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.07000</td>
<td>0.09000</td>
<td>0.15000</td>
<td>0.09000</td>
</tr>
<tr>
<td>0.11000</td>
<td>0.14000</td>
<td>0.08000</td>
<td>0.08000</td>
<td>0.14000</td>
</tr>
<tr>
<td>0.11000</td>
<td>0.09000</td>
<td>0.12000</td>
<td>0.13000</td>
<td>0.15000</td>
</tr>
<tr>
<td>0.09000</td>
<td>0.11000</td>
<td>0.14000</td>
<td>0.09000</td>
<td>0.11000</td>
</tr>
<tr>
<td>0.09000</td>
<td>0.12000</td>
<td>0.08000</td>
<td>0.08000</td>
<td>0.13000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.11000</td>
<td>0.13000</td>
<td>0.13000</td>
<td>0.05000</td>
</tr>
<tr>
<td>0.09000</td>
<td>0.08000</td>
<td>0.10000</td>
<td>0.09000</td>
<td>0.09000</td>
</tr>
<tr>
<td>0.13000</td>
<td>0.10000</td>
<td>0.15000</td>
<td>0.07000</td>
<td>0.10000</td>
</tr>
</tbody>
</table>
Table 6.6 KS minimax node scores over all classes.

<table>
<thead>
<tr>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
<th>Level 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.07000</td>
<td>0.08000</td>
<td>0.10000</td>
<td>0.12000</td>
<td>0.07000</td>
</tr>
<tr>
<td>0.07000</td>
<td>0.07000</td>
<td>0.07000</td>
<td>0.11000</td>
<td>0.11000</td>
</tr>
<tr>
<td>0.06000</td>
<td>0.09000</td>
<td>0.15000</td>
<td>0.18000</td>
<td>0.19000</td>
</tr>
<tr>
<td>0.07000</td>
<td>0.08000</td>
<td>0.09000</td>
<td>0.12000</td>
<td>0.12000</td>
</tr>
<tr>
<td>0.06000</td>
<td>0.09000</td>
<td>0.18000</td>
<td>0.22000</td>
<td>0.18000</td>
</tr>
<tr>
<td>0.07000</td>
<td>0.12000</td>
<td>0.07000</td>
<td>0.08000</td>
<td>0.06000</td>
</tr>
<tr>
<td>0.10000</td>
<td>0.09000</td>
<td>0.08000</td>
<td>0.07000</td>
<td>0.05000</td>
</tr>
<tr>
<td>0.06000</td>
<td>0.08000</td>
<td>0.12000</td>
<td>0.12000</td>
<td>0.12000</td>
</tr>
<tr>
<td>0.06000</td>
<td>0.07000</td>
<td>0.12000</td>
<td>0.14000</td>
<td>0.09000</td>
</tr>
<tr>
<td>0.08000</td>
<td>0.08000</td>
<td>0.18000</td>
<td>0.22000</td>
<td>0.18000</td>
</tr>
<tr>
<td>0.07000</td>
<td>0.09000</td>
<td>0.11000</td>
<td>0.13000</td>
<td>0.05000</td>
</tr>
<tr>
<td>0.08000</td>
<td>0.10000</td>
<td>0.09000</td>
<td>0.09000</td>
<td>0.09000</td>
</tr>
</tbody>
</table>
Figure 6.7 (a) KS individual scores over all classes. White indicates largest values. (b) Resulting basis selected. White indicates selected basis.
6.4 Kullback-Leibler Empirical Density Best Basis

We conclude this chapter by briefly mentioning a complementary approach that was recently published by Saito. In [64], Saito describes an extension to his original LDB algorithm. He addresses the problem of the first issue that arises with LDB, that of the averaging effect of the class energy maps, and noted that the energy map approach does not always represent the data well. The new approach utilizes average shifted histograms (ASH) [68] to estimate probability densities for the classes and uses the Kullback-Leibler divergence as the discriminant. His technique continues the summation method (L1 normalization) for the discriminant scores as in the original LDB. In our implementation of his technique, we extend the scoring to include our score matrix with alternative normalizations for comparison purposes with our KS approach.
CHAPTER 7

SVM HYPERPARAMETER OPTIMIZATION

Support vector machines (SVMs) are generated without the use of user-specified models. Selection of the kernel hyperparameters is critical to the performance, directly impacting the generalization and classification efficacy of the SVM [12][13][24][25][71]. An automated procedure for hyperparameter selection is clearly desirable given the intractable problem of exhaustive search methods. Selection of the kernel hyperparameters is discussed by Cristianini et al. [24][25][26]. Their Kernel-Adatron [24] approach performs a search for the optimal selection of the Gaussian kernel hyperparameter value by starting with a “small” value of \( \sigma \) and iteratively increasing the value through a predefined range and computes a theoretical upper bound on the error. Determining criteria for “small” was outside the scope of their research. In [25], Cristianini et al. define the concept of kernel alignment, and discuss selection of kernels and kernel hyperparameters as a function of the kernel matrix.

In this chapter, we investigate the kernel width hyperparameter, \( \sigma \), for a Gaussian kernel. The kernel width hyperparameter is frequently selected on an experimental basis with iterative training of the SVM on various values of \( \sigma \) until “good enough” results are obtained. Training an SVM with Gaussian kernel hyperparameter values that are too small results in over-fitting or memorization of the
training set, while values that are too large produce SVMs that cannot distinguish between classes either for the training or test data set. Figure 7.1 illustrates this behavior. In (a), the true checkerboard decision boundaries are shown. In (b) and (c), the over-fit and under-fit cases are shown, both of which lead to poor classification efficacy of the SVM. We investigate techniques to identify an approach for a simple, computationally efficient algorithm for selection of the Gaussian kernel width hyperparameter, which concurrently minimizes between-class similarity and maximizes within-class similarity. To that end, we develop two approaches for hyperparameter selection. Our initial approach investigates analyzing the SVM training data distribution of the margins for a Gaussian kernel in order to guide the kernel hyperparameter selection process by minimization of the number of support vectors [71]. The second approach seeks an algorithm that identifies clusters with high similarity within class and low similarity between classes, acting directly on the kernel matrix rather than iterating the SVM training, thereby significantly decreasing the time to identify the desired value for $\sigma$ [12].

7.1 Hyperparameter Selection Via Class-Conditional Margin Distributions and Minimization of Number of Support Vectors

What is needed, and forms the essence of this section, is the identification of a simple approach that can differentiate between the conditions of over-fitting
Figure 7.1 Decision boundaries for two-class checkerboard problem. (a) True decision boundaries. (b) Over-fit case. (c) Under-fit case.

and under-fitting for SVM training with Gaussian kernels (Figure 7.1), thereby leading to an optimal range in the iterative search for \( \sigma \). We demonstrate a simple visualization approach for identifying whether a particular value of \( \sigma \) is too large or too small. We accomplish this by looking at the class-conditional margin distributions associated with the training set projected onto the vector defined by the SVM.

The distribution of margins generated by an SVM is literally the projection of the data onto the hyperplane (see (3.9)) defined by the SVM in the feature space; the margins define a one-dimensional representation of the data from the viewpoint
of the “optimal” decision boundary. The distribution of margins can be estimated and visualized by simple classical techniques. We chose to use Parzen [54] kernel density estimates, a classic nonparametric density estimation approach, to generate density estimates of the class-conditional SVM margin distributions. Parzen kernel density functions have the same kernel width selection issue as SVMs. Scott [68] demonstrates, under the assumption of an underlying normal distribution, the kernel width that minimizes the mean integrated squared error can be approximated as

\[ h \approx 1.06 \hat{s} n^{-\frac{1}{5}}, \]  

(7.1)

where \( \hat{s} \) is the standard deviation of the class-conditional margin values and \( n \) is the number of samples. The kernel width, \( h \), is computed independently for each class. We use equation (3.17) for Parzen width determination, but, to avoid numerical issues, we also set the lower bound of \( h \) to be \( 10^{-2} \). The class-conditional margin distributions associated with the SVMs of Figure 7.1 are shown in Figure 7.2. Note that the class-conditional densities when the SVM Gaussian \( \sigma \) value is too small are two impulse functions (centered at \( \pm 1 \)), whereas the distributions overlap significantly when the value of \( \sigma \) is too large. We seek a value for \( \sigma \) that separates the classes (densities) without either of these extremes. For both these conditions, virtually all the training patterns are support vectors of their respective SVMs, as their margin distributions are contained in the region \( \gamma \leq 1 \) as was
discussed in Chapter 3. In both cases, the classification efficacy of the SVM is poor. In fact, both over- and under-fitting hyperparameters produce equivalently poor SVM classification performance, which precludes SVM performance as an indicator to differentiate an over-fitting from an under-fitting hyperparameter. Likewise, the number of support vectors produced during training with an ill-chosen hyperparameter does not distinguish between these conditions. For the SVMs illustrated in Figure 7.2, the number of support vectors was the same, with all samples included as support vectors.

By examining plots (as in Figure 7.2) of the class-conditional margin distributions associated with the training set mapped onto the vector defined by the SVM, an over-fit or under-fit condition is readily declared (by the physical separation or overlap of the margin data) and a range for the kernel width hyperparameter, $\sigma$, can
be identified. Although the class data must be trained in this range to experimen-
tally determine the desired value for $\sigma$, the initial search range can be significantly
limited, thereby decreasing the number of iterations of SVM training required.
Moreover, this method provides insight into the separability of classes with the
SVM.

Once a range for the search is established, we iterate the training towards the
minimization of the number of support vectors. After finding an initial minimim,
we can continue to iterate in "small" increments about that point until the smallest
number of support vectors is realized. In practice, we set our iterations to some
maximum level in order to bound the problem. This maximum number of itera-
tions is driven by the specific problem under investigation.

7.2 Hyperparameter Selection Via Kernel Semi-Alignment

Our second approach seeks an algorithm that identifies clusters with high
similarity within class and low similarity between classes, by employing unsuper-
vised kernel hyperparameter selection. Our approach acts directly on the kernel
(similarity) matrix rather than iterating on SVM training, thereby significantly de-
creasing the time to identify the desired value for $\sigma$. Statistical methods are applied
to the similarity matrix to determine kernel optimization in an unsupervised fashion.
This preprocessing framework removes the requirement for iterative SVM training.
In [25] and [26], Cristianini et al. define the concept of kernel alignment, and discuss determination of kernels and kernel hyperparameters as a normalized function of the Frobenius inner product of the kernel matrix and the outer product of the class labels. Cristianini defines kernel alignment as

\[
A = \frac{\langle K, yy \rangle_F}{\sqrt{\langle K, K \rangle_F \langle yy', yy \rangle_F}},
\]

(7.2)

where \( K \) is the kernel or similarity matrix, \( y \) is the class label (with \( y' \) its transpose), and \( \langle \cdot \rangle_F \) denotes that the Frobenius norm is computed on the resultant matrix. The Frobenius norm is defined as

\[
\|X\|_F = \left( \sum_{i=1}^{m} \sum_{j=1}^{n} |x_{ij}|^2 \right)^{1/2} \quad \forall i, j
\]

(7.3)

for a matrix of dimension \( m \times n \). The Frobenius norm is computed on the similarity or kernel matrix in order to determine the alignment and adapt the kernel. Alignment is a simple, yet effective, statistic that provides a measure for maximizing the within-class similarity (clustering) induced via the kernel hyperparameters, while penalizing between-class similarity induced by the same kernel hyperparameters.

The alignment statistic as defined does not differentiate between within-class clustering of the class of interest and within-class clustering of the remaining classes grouped together as a single class, the world. While alignment is effective for true two-class problems, when there are more than two classes, the statistic can
introduce bias due to the clustering when attempting to maximize the similarity of the *world* data vectors.

This research amends the *alignment* approach with *semi-alignment* in a straightforward manner by applying a Frobenius inner product on a subset of the similarity matrix rather than on the entire matrix. By using a subset of the matrix, we remove the within-class similarity of the *world* class from consideration. For multi-class cases (more than two classes), the statistic will no longer encourage the collection of classes in the *world* to look “alike.” Although this may decrease the sample support (number of training samples used) for a true two-class case, it removes the induction of a false bias caused by the treatment of disparate classes of the *world* as a single class.

We form a similarity matrix on the training data using the kernel trick with the Gaussian kernel given in (3.14) and expressed as a function of $\sigma$:

$$K_\sigma(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$$  \hspace{1cm} (7.4)

Figure 7.3 shows a two-class similarity matrix. Note that the texture seen is due to the checkerboard nature of our data set and is seen in subsequent similarity matrices for the checkerboard and quadboard problems. The quadrants on the diagonal represent within-class similarities while the anti-diagonal quadrants represent between-class similarities. We desire large values in the within-class quadrants and small values in the between-class quadrants.
Sample similarity matrices for a two-class case for four values of \( \sigma \) are shown in Figure 7.4. For the first plot, (a), the value for \( \sigma \) (0.1) is too small for this data set resulting in comparable within and between-class similarity. In (b), \( \sigma = 0.4 \), the plot shows high within-class similarity and low between-class similarity. For both (c) and (d), the value of \( \sigma \) is too large and is beginning to form large enough clusters that the classes look “alike” resulting in similar within-class and between-class similarity. For all cases except (b), \( \sigma = 0.4 \), it would be difficult to distinguish between samples from the two classes. The separability information of the classes is contained entirely in these similarity matrices, rendering training of the SVM for hyperparameter selection unnecessary. Our approach follows.

![Sample similarity matrix](image)

**Figure 7.3** Sample similarity matrix. White indicates large magnitudes while black symbolizes small magnitudes.
We begin with formation of a similarity matrix with an initial guess for $\sigma$. We calculate the Frobenius norm as the test statistic on a sample weighted subset of the matrix. The empirical semi-alignment of the kernel matrix, $K$, and the class labels, $yy'$, is our test statistic, $f(\sigma)$.
\[ f(\sigma) = \langle K_{\sigma}, y'y' \rangle = \frac{-n_w}{n} \sum_{y_i = y_j} K_{\sigma}(x_i, x_j) + \frac{n_c}{n} \sum_{y_i \neq y_j} K_{\sigma}(x_i, x_j) \] (7.5)

The kernel matrix, \( K_{\sigma}(x_i, x_j) \), is a function of \( \sigma \); \( i \) and \( j \) are indexed only over the class of interest, \( y = 1 \), in the first summation. In the second summation, \( i \) is indexed over the class of interest and \( j \) is indexed over the world. By defining \( f(\sigma) \) as the negative of \( \langle K_{\sigma}, y'y' \rangle \), we can utilize a gradient descent approach to expedite the search for the minimum value of the function. Weighting the function by the ratios \( n_w/n \) and \( n_c/n \) (where \( n_c \) is number of samples in the class of interest, \( n_w \) is number of samples in world and \( n \) is the total samples) allows adjustment of the statistic when class sizes are not equal. As it is a monotonic adjustment, the weighting does not affect the resultant \( \hat{\sigma}_{opt} \), but allows maintenance of a normalized scale for comparison purposes. Note that we have not carried the denominator of (7.2) forward in (7.5). The formulation of (7.2) results in a correlation coefficient, which is not necessary for our statistic.

Figure 7.5 shows possible subsets of the matrix used to calculate the semi-alignment test statistic for both two and four-class cases. Class C1 is shown as the class of interest for both scenarios. In the two-class case, C2 is the world while, in the four-class case, classes C2 - C4 are grouped together as the world. Any of the classes could have been designated as the class of interest. As is shown in the plots, the semi-alignment method uses only the class of interest and the between-class data from the matrix. Using the subset of the similarity matrix removes the group-
ing of the world classes compared to each other as a single class and instead compares a grouping of each class only to the class of interest. Depending on the data set, this can avoid the problem of selecting a value for \( \sigma \) that induces a false similarity on the classes, which make up the world class. In Figure 7.6, the groupings of the alignment similarity matrices are shown. The two-class similarity matrix hints to a similar result as would be seen with the semi-alignment approach. It shows two single within-class groupings and two between-class comparison regions. For semi-alignment, only half of the data is used, resulting in better sample support for the alignment approach with a two-class case. Since sample support is generally an issue in classification, alignment could prove to be a favorable approach for some true two-class cases. Generally, however, we consider recognition problems with more than two classes. The four-class alignment approach groupings are shown in Figure 7.6 (b). We see that all three world classes are forced into a single class grouping. The search for the optimal value for \( \sigma \) now clusters these classes into a single class and seeks high within-class similarity in the cluster.
Figure 7.5 Subset of similarity matrix used for semi-alignment calculation with weighted Frobenius norm. Cross-hatched is the within-class group; single-hatched is the between-class group. (a) Two-class case. (b) Four-class case.

Figure 7.6 Similarity matrix with weighted Frobenius norm, used for alignment calculation. Cross-hatched is the within-class group; single-hatched is the between-class group. (a) Two-class case. (b) Four-class case.
Our approach is summarized in Figure 7.7.

**Step 1:** Initialize $\sigma$. Starting with a "small" value of $\sigma$, use a bracketing technique (discussed in [58]) to find appropriate minimum and maximum values for $\sigma$ search.

**Step 2:** Calculate $f(\sigma)$ and $\frac{df}{d\sigma}$. Note that $\frac{df}{d\sigma}$ is the derivative of $f(\sigma)$ and can be calculated by using the derivative of the kernel function in (7.5). Search over bracketed values found in step 1 for a minimum function value using a gradient descent approach.

**Step 3:** Iterate steps 1-2 for an estimate of the "optimal" value of $\sigma$ for each class.

Figure 7.7 Kernel semi-alignment algorithm.

### 7.3 SVM Hyperparameter Optimization Trade Study

To characterize the efficacy of our approach, we investigated two-class and four-class classification problems with specified underlying distributions. We studied our two techniques. The first was selection of a range of $\sigma$ for training the SVMs through visualization of the class-conditional margins and subsequent selection based on the minimum number of support vectors. Secondly, we applied our semi-alignment kernel optimization algorithm to obtain an estimate for $\sigma$. We also compared the alignment approach to our semi-alignment approach. Additional values of $\sigma$ were pre-determined for use in the evaluation phase in order to compare
classification performance of $\hat{\sigma}_{opt}$ obtained by our current methods to that obtained through iterating over a range of values. Our approach for the testing phase is presented in Figure 7.8.

| Step 1: Calculate $\hat{\sigma}_{opt}$ (Figure 7.7). |
| Step 2: Specify additional “reasonable” values of $\sigma$ for testing based on prior knowledge of the data. |
| Step 3: Train SVM on all values of $\sigma$ specified (see Chapter 3). |
| Step 4: Classify test data with SVM (see Chapter 3). |
| Step 5: Compare classification efficacy for $\hat{\sigma}_{opt}$ obtained in step 1 with $\sigma$ values from step 2. |

Figure 7.8 Kernel optimization performance evaluation steps.

Our first data set, the checkerboard data, is a two-dimensional pattern space consisting of two classes distributed in a $4 \times 4$ cell checkerboard pattern. We generated 500 Monte Carlo data sets by utilizing a random uniform number generator ([0,1]) for two features (x and y) and offsetting the features by the current cell location. Each set consisted of samples that were assigned to the appropriate class for training data and test data. For our training set, random sets of 320 training patterns (20 per cell) were generated; the test set consists of 4800 samples for evaluation of classification efficacy. A second data set was generated, a quadboard pattern, which is a four-class data set in an $8 \times 8$ cell pattern. This set consists of 200
random sets of 1280 training patterns, again 20 per cell. The corresponding test set has 19,200 samples. Sample data sets for each of these are shown in Figure 7.9.

![Figure 7.9 Sample data sets with 100 samples shown per cell. (a) Checkerboard. (b) Quadboard.](image)

We tested four sample margin values for the checkerboard problem to investigate the relationship between kernel size and margin distributions. This presents us with an easily visualized environment in which we can observe the class-conditional margin distributions. Figure 7.10 shows a sample data set with the decision boundaries generated by the SVM and the corresponding Parzen density estimates of the margin distributions. We started with a small value ($\sigma = 0.05$, plot (a)), which shows complete separation of the training classes indicating that over-training would occur with this value. We used this as a lower bound for $\sigma$ and iterated the SVM training with $\sigma = 4.0$ (shown in (d)). At this value, the margins of
different classes overlap considerably. This could have been used as an upper bound, but we chose to further restrict the range and iterated with $\sigma = 1.0$ in order to decrease the number of iterations. From the plots at this value, (c), we see that the decision boundaries would not able to generalize well enough for clear definition of the checkerboard classes (margins overlap by 37%), indicating that this is too large of a value for $\sigma$. This does give us an upper bound for the search for $\sigma$ by iterative training. By training on values for $\sigma$ in the range $0.05 \leq \sigma \leq 1$, we are able to control the number of iterations. We adjusted the value of $\sigma$ towards the minimization of the number of support vectors in this range and determined a good value at $\sigma = 0.4$ (plot (b)).
Figure 7.10  Checkerboard data set with decision boundaries (left) and associated SVM margin distributions (right).
(a) $\sigma = 0.05$. (b) $\sigma = 0.4$. (c) $\sigma = 1.0$. (d) $\sigma = 4.0$. 
Figure 7.11 Classification efficacy error with *semi-alignment* (0.33 ≤ σ_{opt} ≤ 0.39) and pre-specified values for σ for the two-class checkerboard data.

Figure 7.11 shows a scatter plot of the classification error rate for various values of σ. Table 7.1 tabulates the association between the kernel hyperparameter (σ), the mean and standard deviation of the number of support vectors in the associated SVM, and the corresponding mean and standard deviation for the classification efficacy results for the checkerboard problem. The iterative training approach resulted in a minimum number of support vectors at σ = 0.4. The classification efficacy was at a near maximum at this value. Our iterative training support vector minimization approach selects this value for σ. The last two rows show the results
for $\hat{\sigma}_{opt}$ with semi-alignment and alignment techniques. The semi-alignment value of $\hat{\sigma}_{opt}$, as determined by our algorithm, is in the range of $0.33 \leq \hat{\sigma}_{opt} \leq 0.39$ with a mean of 0.35 and standard deviation of 0.0099 for our random data set trials for this two-class case. The alignment algorithm resulted in essentially the same results. Note that Figure 7.11 shows that the error rate for $\hat{\sigma}_{opt}$ is in the neighborhood of the optimal value obtained by our iterative training support vector minimization approach.

Table 7.1 Relationship between kernel hyperparameter $\sigma$, the SVM support vector (s.v.) statistics, and PCC statistics for two-class checkerboard problem (500 Monte Carlo iterations).

<table>
<thead>
<tr>
<th>Kernel $\sigma$ Value</th>
<th># s.v. mean</th>
<th># s.v. stdev</th>
<th>PCC mean</th>
<th>PCC stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>316</td>
<td>2.1</td>
<td>81.6</td>
<td>0.03</td>
</tr>
<tr>
<td>0.10</td>
<td>279</td>
<td>5.1</td>
<td>90.4</td>
<td>0.87</td>
</tr>
<tr>
<td>0.20</td>
<td>147</td>
<td>5.2</td>
<td>91.3</td>
<td>0.83</td>
</tr>
<tr>
<td>0.30</td>
<td>101</td>
<td>5.1</td>
<td>91.6</td>
<td>0.89</td>
</tr>
<tr>
<td>0.40</td>
<td>95</td>
<td>5.7</td>
<td>92.3</td>
<td>0.86</td>
</tr>
<tr>
<td>0.50</td>
<td>99</td>
<td>6.1</td>
<td>92.4</td>
<td>0.89</td>
</tr>
<tr>
<td>0.60</td>
<td>113</td>
<td>6.4</td>
<td>91.9</td>
<td>0.95</td>
</tr>
<tr>
<td>0.70</td>
<td>135</td>
<td>6.7</td>
<td>90.8</td>
<td>1.10</td>
</tr>
<tr>
<td>1.0</td>
<td>228</td>
<td>6.6</td>
<td>79.4</td>
<td>1.58</td>
</tr>
<tr>
<td>2.0</td>
<td>298</td>
<td>3.4</td>
<td>56.0</td>
<td>2.14</td>
</tr>
<tr>
<td>4.0</td>
<td>302</td>
<td>3.0</td>
<td>51.2</td>
<td>0.03</td>
</tr>
<tr>
<td>semi-alignment</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.33 \leq \hat{\sigma}_{opt} \leq 0.39</td>
<td>97</td>
<td>5.4</td>
<td>92.0</td>
<td>0.87</td>
</tr>
<tr>
<td>alignment</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.34 \leq \hat{\sigma}_{opt} \leq 0.39</td>
<td>97</td>
<td>5.4</td>
<td>91.7</td>
<td>0.86</td>
</tr>
</tbody>
</table>
Figure 7.12 Classification efficacy error with *semi-alignment* 
(0.58 ≤ \( \hat{\sigma}_{\text{opt}} \) ≤ 0.67) and pre-specified values for \( \sigma \) for the four-class quadboard data.

The scatter plot of the results for the four-class quadboard case are shown in Figure 7.12 with the details of the results shown in Table 7.2. The iterative training support vector minimization approach results in selection of 0.5 as the value for the kernel width. This value of \( \sigma \) results in 305 support vectors and the highest classification efficacy. *Semi-alignment* results for \( \hat{\sigma}_{\text{opt}} \) obtained for our four-class case were in the range 0.58 ≤ \( \hat{\sigma}_{\text{opt}} \) ≤ 0.67 with a mean of 0.61 and a standard deviation of 0.015. For the *alignment* approach, the range for the optimal value of \( \sigma \), found over multiple trials, varies more widely than with the *semi-alignment* approach.
The results show $0.75 \leq \hat{\sigma}_{opt} \leq 1.37$ with a mean value of $\sigma = 0.90$ and a standard deviation of 0.15. The value for $\hat{\sigma}_{opt}$ is larger for this approach than was found by semi-alignment and the classification efficacy degraded by seven percent. The alignment approach clusters the world data together forcing it to appear as a single class. This requires the kernel width to be larger than optimal and induces a false "alikeness" on the world data. While the PCC for the iterative training method is slightly higher than for the semi-alignment approach for this particular data set, the automation of semi-alignment removes the requirement for SVM training thus making it preferable.

<table>
<thead>
<tr>
<th>Kernel $\sigma$ Value</th>
<th># s.v. mean</th>
<th># s.v. stdev</th>
<th>PCC mean</th>
<th>PCC stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>1096</td>
<td>52.1</td>
<td>76.4</td>
<td>4.96</td>
</tr>
<tr>
<td>0.10</td>
<td>1134</td>
<td>9.1</td>
<td>88.8</td>
<td>0.47</td>
</tr>
<tr>
<td>0.20</td>
<td>655</td>
<td>7.7</td>
<td>89.8</td>
<td>0.46</td>
</tr>
<tr>
<td>0.30</td>
<td>418</td>
<td>6.0</td>
<td>90.5</td>
<td>0.50</td>
</tr>
<tr>
<td>0.40</td>
<td>324</td>
<td>6.1</td>
<td>91.1</td>
<td>0.48</td>
</tr>
<tr>
<td>0.50</td>
<td>305</td>
<td>6.2</td>
<td>91.3</td>
<td>0.49</td>
</tr>
<tr>
<td>0.60</td>
<td>313</td>
<td>6.4</td>
<td>90.8</td>
<td>0.51</td>
</tr>
<tr>
<td>0.70</td>
<td>350</td>
<td>7.7</td>
<td>89.2</td>
<td>0.56</td>
</tr>
<tr>
<td>1.0</td>
<td>438</td>
<td>9.8</td>
<td>79.8</td>
<td>0.44</td>
</tr>
<tr>
<td>2.0</td>
<td>644</td>
<td>2.5</td>
<td>69.4</td>
<td>1.28</td>
</tr>
<tr>
<td>4.0</td>
<td>960</td>
<td>142</td>
<td>50.1</td>
<td>9.85</td>
</tr>
<tr>
<td><strong>semi-alignment</strong></td>
<td><strong>317</strong></td>
<td><strong>7.1</strong></td>
<td><strong>90.6</strong></td>
<td><strong>0.51</strong></td>
</tr>
<tr>
<td>$0.58 \leq \hat{\sigma}_{opt} \leq 0.67$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Alignment</strong></td>
<td><strong>511</strong></td>
<td><strong>24.4</strong></td>
<td><strong>83.5</strong></td>
<td><strong>1.54</strong></td>
</tr>
<tr>
<td>$0.75 \leq \hat{\sigma}_{opt} \leq 1.37$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Although the two-class case results are essentially identical for alignment and semi-alignment, we see benefit for using the semi-alignment approach for multiple classes of data. We next investigate the effect of sample support on the two techniques, alignment and semi-alignment, by considering results with reduced numbers of training samples. Table 7.3 shows performance results with 16, 32, 80, 160 and 320 training samples for the checkerboard problem. The performance degrades as the sample support decreases, but the results with the two techniques are essentially identical. With 16 training samples, the PCC is close to guessing. The results of the quadboard (four-class case) are shown in Table 7.4. Again, we find degradation of performance as the sample support decreases. The semi-alignment approach consistently outperforms the alignment technique with this multiple class case until the sample support has decreased to 1 sample per cell. At this sample level, the semi-alignment approach results are similar to guessing (PCC = 26.5%), while the alignment approach performs slightly better (PCC = 36.3%). Alignment clustering provides more sample support. Even though the clusters are too large to result in good classification efficacy, the results are better than for the semi-alignment approach when the sample support is so low that there is no benefit from the technique. At this sample support level, generalization of the training data for test would not be effective with either semi-alignment or alignment.
Table 7.3 Results for variations in the training sample support for the two-class case.

<table>
<thead>
<tr>
<th>Number of training samples</th>
<th>Technique</th>
<th># s.v. mean</th>
<th># s.v. stdev</th>
<th>PCC mean</th>
<th>PCC stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 (1/cell)</td>
<td>semi-alignment</td>
<td>16</td>
<td>0</td>
<td>51.0</td>
<td>0.04</td>
</tr>
<tr>
<td>16 (1/cell)</td>
<td>alignment</td>
<td>16</td>
<td>0</td>
<td>50.7</td>
<td>0.03</td>
</tr>
<tr>
<td>32 (2/cell)</td>
<td>semi-alignment</td>
<td>31.7</td>
<td>0.50</td>
<td>72.3</td>
<td>0.08</td>
</tr>
<tr>
<td>32 (2/cell)</td>
<td>alignment</td>
<td>31.7</td>
<td>0.55</td>
<td>71.2</td>
<td>0.08</td>
</tr>
<tr>
<td>80 (5/cell)</td>
<td>semi-alignment</td>
<td>54.0</td>
<td>3.6</td>
<td>83.2</td>
<td>1.8</td>
</tr>
<tr>
<td>80 (5/cell)</td>
<td>alignment</td>
<td>53.9</td>
<td>3.6</td>
<td>83.1</td>
<td>1.8</td>
</tr>
<tr>
<td>160 (10/cell)</td>
<td>semi-alignment</td>
<td>69.1</td>
<td>3.9</td>
<td>87.9</td>
<td>1.3</td>
</tr>
<tr>
<td>160 (10/cell)</td>
<td>alignment</td>
<td>69.0</td>
<td>3.9</td>
<td>87.9</td>
<td>1.3</td>
</tr>
<tr>
<td>320 (20/cell)</td>
<td>semi-alignment</td>
<td>97</td>
<td>5.4</td>
<td>92.0</td>
<td>0.87</td>
</tr>
<tr>
<td>320 (20/cell)</td>
<td>alignment</td>
<td>97</td>
<td>5.4</td>
<td>91.7</td>
<td>0.86</td>
</tr>
</tbody>
</table>
### Table 7.4 Results for variations in the training sample support for the four-class case.

<table>
<thead>
<tr>
<th>Number of training samples</th>
<th>Technique</th>
<th># s.v. mean</th>
<th># s.v. stdev</th>
<th>PCC mean</th>
<th>PCC stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 (1/cell)</td>
<td><em>semi-alignment</em></td>
<td>64</td>
<td>0</td>
<td>26.5</td>
<td>0.04</td>
</tr>
<tr>
<td>64 (1/cell)</td>
<td><em>alignment</em></td>
<td>32</td>
<td>0.2</td>
<td>36.3</td>
<td>0.05</td>
</tr>
<tr>
<td>128 (2/cell)</td>
<td><em>semi-alignment</em></td>
<td>102</td>
<td>5.6</td>
<td>73.8</td>
<td>0.02</td>
</tr>
<tr>
<td>128 (2/cell)</td>
<td><em>alignment</em></td>
<td>43</td>
<td>5.3</td>
<td>69.5</td>
<td>0.05</td>
</tr>
<tr>
<td>320 (5/cell)</td>
<td><em>semi-alignment</em></td>
<td>130</td>
<td>3.6</td>
<td>81.9</td>
<td>0.01</td>
</tr>
<tr>
<td>320 (5/cell)</td>
<td><em>alignment</em></td>
<td>67</td>
<td>4.7</td>
<td>76.7</td>
<td>0.03</td>
</tr>
<tr>
<td>640 (10/cell)</td>
<td><em>semi-alignment</em></td>
<td>197</td>
<td>5.2</td>
<td>86.8</td>
<td>0.01</td>
</tr>
<tr>
<td>640 (10/cell)</td>
<td><em>alignment</em></td>
<td>116</td>
<td>7.3</td>
<td>80.6</td>
<td>0.01</td>
</tr>
<tr>
<td>1280 (20/cell)</td>
<td><em>semi-alignment</em></td>
<td>317</td>
<td>7.1</td>
<td>90.6</td>
<td>0.51</td>
</tr>
<tr>
<td>1280 (20/cell)</td>
<td><em>alignment</em></td>
<td>511</td>
<td>24.4</td>
<td>83.5</td>
<td>1.54</td>
</tr>
</tbody>
</table>

### 7.4 Conclusion

In our exploitation of the SVMs for classification, we develop two approaches to the selection of the Gaussian kernel width. The first uses class-conditional densities to set a range, and then investigates the number of support vectors in the trained SVM by training on a variety of kernel widths. When the number of support vectors is at a minimum, the kernel width hyperparameter is determined to be "close" to an optimal value. The second approach formulates *semi-
alignment criteria to measure between-class and within-class scatter on the similarity matrix. This automated method requires the formation of a kernel matrix, or class pair-wise similarity matrix, of the data, but does not require SVM training, making it a faster approach than the first method.

The investigation of the distributions of the class-conditional margins and number of support vectors provides significant information concerning the appropriateness of the Gaussian kernel hyperparameter for SVM optimization. Common behavior of the margin distributions has been identified and characterized. The approach entails several iterations of training the SVM in order to minimize the number of support vectors. The efficacy of a particular kernel width can be visually determined via one-dimensional density estimate plots of the training data margin values. Projecting the data onto the SVM hyperplane allows one-dimensional analysis of the data from the viewpoint of the “trained” SVM. The effect of kernel hyperparameter selection on class-conditional margin distributions is thus demonstrated in the one-dimensional projection subspace and shown in Figure 7.10. This is a simple visualization approach for quickly identifying whether a particular value of $\sigma$ is “too large” or “too small” by looking at the class-conditional margin distributions associated with the training set and mapped onto the vector defined by the SVM. This results in a method to determine the distribution for a particular value of $\sigma$ after training occurs.
For both classes of data, we see a correlation between classification performance and minimization of the number of support vectors required by the respective SVM. This is an indication of the generalization ability of the trained SVM as noted by the corresponding PCC. Selecting $\sigma$ too small or too large resulted in degradation of performance. When $\sigma$ was too large, the performance of the SVM degraded to the point where it was essentially equivalent to guessing. Generally speaking, over-fit and under-fit states manifest themselves as SVMs with close to the maximum number of support vectors. Additionally, the margin distributions are explicitly different for these conditions, and therefore the margin distributions can provide the information necessary to differentiate the between these states [12][71]. For the two problems investigated, the direct correlation between performance and minimizing the number of support vectors used to define the hyperplane is near optimal. This criterion provides a simple and effective approach for SVM hyperparameter optimization, and the models produced are desirable using model parsimony and data reduction criteria. Given the nature of Gaussian kernels, this approach is equivalent to minimizing the number of components required for covering the appropriate samples.

Although a near optimal value for $\sigma$ was determined by training over a reasonable range of values and selecting $\sigma$ corresponding to the minimum number of support vectors, we note that equivalently good values ($\hat{\sigma}_{opt}$) were found by our
second approach, kernel *semi-alignment*. This approach results in identification of \( \hat{\sigma}_{opt} \) without requiring SVM training, which can be prohibitive in many cases due to lengthy training time. This approach proved to be preferable to *alignment* for multi-class data resulting in kernel width selection that gave better classification efficacy.

While minimizing the number of support vectors used to define the hyperplane provides a method of kernel hyperparameter selection, our second approach, *semi-alignment*, based on a variation of the work of Cristianini *et al.*, provides a strategy for kernel optimization that eliminates iterative SVM training. This approach allows unsupervised evaluation of the hyperparameter resulting in an automated procedure that provides a simple and efficient approach for SVM hyperparameter optimization that does not require training of the SVM for each value of \( \sigma \) of interest, a time consuming proposition. The automated approach of the \( \hat{\sigma}_{opt} \) search algorithm, *semi-alignment*, is the scheme we now employ.
CHAPTER 8

WAVELET BASIS RESULTS

In this chapter, we present the classification results for the wavelet best-basis approaches. Classification throughout this chapter is accomplished with the SVM except as noted. In the next section, we consider the results with 360 training samples per class. We resection the data into new training and test sets for the following section and investigate a more realistic scenario, training with reduced sample support. In this section, we also add the KL method of basis selection for comparison purposes. Next we present the results with semi-alignment and alignment utilized as the methods to select the SVM $\sigma$ hyperparameter. We conclude with a comparison of our semi-alignment SVM and the Bayesian classifier.

8.1 Procedure

In order to determine the effectiveness of the wavelet based methods, the LDB best-basis algorithm and our KS best-basis techniques were tested on both data sets according to the process shown in the block diagram of Figure 8.1 (enumerated in Figure 8.2).
1. Identify training and test signals for $m$ classes.
2. Transform training data set to new basis via best-basis algorithm.
3. Select $k \leq n$ coefficients to use as features ($n =$ signal length).
4. Train classifier on features from training set.
5. Extract features from test data.
6. Apply classifier to $k$ coefficients of the test data.

Figure 8.2 Steps for analysis of classification efficacy.

Two cases were considered, LDB and the KS basis selection techniques. A Daubechies 8 coefficient wavelet family was used for the wavelet packet decomposition and subsets of features with $k \leq n$ were investigated in order to consider the
effects of reducing the dimensionality of the classification problem. SVM input hyperparameters were tested over a range of values, allowing the best kernel width across classes to be selected.

We then reduced the training set sample support and compared results of LDB, KS and Saito's KL basis selection approaches. Since our SVM hyperparameter investigation involved simulated data, we examine the efficacy of semi-alignment on kernel hyperparameter selection for the measured data set. Finally, we compare the SVM performance to that of the Bayesian classifier.

8.2 Comparison of LDB and KS

We first examine the classification efficacy of the ECDF-based KS test as compared to the energy map approach of LDB. For this analysis, we utilize all available training and test samples. The norms used for the KS best basis are the minimax and sup norms; recall that LDB uses an L1 norm.

The classifier results for the simulated data (shown in Figure 8.3) demonstrate that the KS technique outperforms LDB when the desired number of features is small. As the number of features becomes greater than six, LDB selected features slightly outperform those selected via KS best basis, with the performance becoming essentially equivalent when all features are used. The results of the second data set are in Figure 8.4. For this data, the KS technique outperforms the LDB method until the results converge to identical classification results at 64 fea-
tures. With the measured data set, we see a denoising effect from the wavelet basis for the KS approach with the sup norm. This approach has separated the informative subspace into the first $k$ features, leaving noise, or the signal components, which do not add class separation information to the problem, in the remaining features. The PCC decreases slightly as we add more than $k$ features.

It is interesting to note that the $p$-norm technique used with the KS best basis shows some data dependence. For the first data set, the minimax norm performs best at low numbers of features, while the sup norm is slightly better for a complete basis. The sup norm results are superior for the measured data set. This allows some flexibility in exploiting the distributional differences between data sets via the $p$-norm selection. For instance, a data set where a single class is significantly different than all others may perform best for overall classification when using a minimax norm in order to minimize the overall risk of misclassification. However, if the class that demonstrates the most disparity from all others is the class of interest, a sup norm might prove to be preferable.

For both data sets, the KS technique outperforms LDB with reduced dimensionality. When retaining all features resulting in a complete basis, the results illustrate little or no difference between the two techniques.
Figure 8.3 Four-class simulated data set results.

Figure 8.4 Three-class measured data set results.
8.3 Comparison of LDB, KS and KL with Reduced Sample Support

We now redefine training and test subsets of the data in order to test the performance with lower sample support. We also add the KL technique for comparison with our KS approach and the LDB method. For the first data set, the formerly defined training set is divided into test and training sets. A sample at every third aspect is assigned to the training set resulting in 120 training samples. All remaining samples are assigned to the test set leaving 240 samples for testing per class. For the measured data set, we extract 120 samples for training and leave the remaining samples for test. This results in approximately 350 samples for each test class.

We investigate the results of the LDB, KS and KL basis selection techniques. The L1 scoring norm of the KL approach is extended to include the scoring matrix of our KS technique (see (6.8)). The results for all normalization scoring methods are shown in Figures 8.5 - 8.12.

Figure 8.5 shows the results for the L1 norm scoring method for the simulated data set. For this technique, the pair-wise scores in the score matrix are summed and used as the discriminate for pruning the over-complete basis. In Figure 8.6, the L2 or Euclidean norm is used for the KS and KL basis approaches. Figure 8.7 discloses results with the minimax norm while Figure 8.8 presents the sup norm results. The scales on the four plots are the equivalent. In all cases, we see significant improvement in the classification performance with both KS and KL
approaches over LDB at low dimensionality. The preference for KS or KL depends on the normalization scheme. For instance, with the $L_1$ and $L_2$ norms, the KS performance at one and two features is better than for the other normalizations and outperforms both KL and LDB approaches, while KL is better for one and two features with the minimax and sup norm implementations. The extension of Saito’s KL technique to include the alternative normalizations results in better performance with the minimax norm as compared to his original $L_1$ norm scheme for this data set. LDB is superior as the dimensionality increases and the results converge for LDB and KL at full dimensionality in all cases. The KS results converge at full dimensionality for the minimax scheme, but are slightly lower for all others. Overall, we see an increase in classification efficacy with KS and KL approaches at reduced dimensionality.

Figures 8.9 - 8.12 show the results for the measured data set. The scale for these four plots is identical. Here we see more dependence on the normalization scheme. In Figures 8.9 and 8.10, LDB outperforms both the KS and KL methods. Figure 8.11 shows results with the minimax scoring. The improvement in performance at low dimensionality with the KS and KL methods is evident. Figure 8.12 (sup norm) shows the best performance for this data set with the KS basis selection demonstrating a significant increase in classification efficacy at low dimensionality.

The benefit of the scoring matrix scheme is evident from the results of these two data sets. Clearly, the specific task at hand would drive, not only the basis se-
lection technique, but also the scoring matrix norm for that technique. For instance, if full dimensionality is being used, the LDB approach, which averages the wavelet packets by class, would be preferable for both data sets since it requires less data storage by using the class energy maps. However, for our data sets of interest that require classification in real-time settings, reduction of dimensionality is a driving force, which leads to the selection of our alternate approaches. The selection of the specific approach is dependent on a trade-off in the desired dimensionality and required classification performance. Retention of the score matrix allows exploitation of the norm technique that best suits the data.
Figure 8.5 Classification efficacy for basis scoring method L1 for data set 1 (four-class simulated).

Figure 8.6 Classification efficacy for basis scoring method L2 for data set 1 (four-class simulated).
Figure 8.7 Classification efficacy for basis scoring method minimax for data set 1 (four-class simulated).

Figure 8.8 Classification efficacy for basis scoring method sup for data set 1 (four-class simulated).
Figure 8.9 Classification efficacy for basis scoring method L1 for data set 2 (three-class measured).

Figure 8.10 Classification efficacy for basis scoring method L2 for data set 2 (three-class measured).
Figure 8.11 Classification efficacy for basis scoring method minimax for data set 2 (three-class measured).

Figure 8.12 Classification efficacy for basis scoring method sup for data set 2 (three-class measured).
8.4 Selection of SVM Kernel Hyperparameters on Measured Data

Our investigation into an automated approach for selection of an optimal value for the Gaussian kernel width resulted in the development of a technique, which we call *semi-alignment*. This was examined with simulated data sets: the random checkerboard and quadboard data sets. A natural question arises; what is the benefit of these techniques with a measured data set? To answer that question, we use the three-class measured data set. We continue working with the reduced training set as was described in the last section and investigate selection of the Gaussian kernel width hyperparameter with *semi-alignment* and *alignment*. Our baseline value of $\sigma$ for comparison purposes is 0.5. The results are presented in Figures 8.13 - 8.20. The figures shows a comparison of the classification efficacy of the data set using the KS or KL basis selection techniques with the indicated scoring metrics for the baseline and optimized values of $\sigma$. All scales for KS are identical as are the scales for the KL technique.

Figures 8.13 and 8.14 show the results for the KS L1 and L2 approaches. Although these approaches were not the top performers for overall classification efficacy with KS, there is a marked improvement in the results with the $\sigma$ optimization approach. The features that added confusion to the classification process without the optimization continue to exhibit a similar confusion with the optimized values for $\sigma$. In fact, the reduction in performance is greater with the optimized values. With optimized SVM hyperparameters, we continue to see the inherent data de-
pendence that must be considered when selecting the scoring approach. The KS L1 and L2 approaches show improvement with optimization, but do not exhibit performance that would elicit their selection for classification with this data set. Figures 8.15 - 8.16 show the results for the KS minimax and sup scoring approaches. Recall that these approaches resulted in improved performance over LDB with \( \sigma = 0.5 \). We see significant improvement in the performance by optimizing \( \sigma \) with both semi-alignment and alignment for both scoring methods. The results for the two optimization techniques diverge slightly with the semi-alignment exhibiting superior performance at low dimensionality and alignment at higher dimensionality.

The performance of this data set with the KL technique results in several interesting conclusions. Figures 8.17 - 8.20 show these results. For both the L1 and L2 approaches, there is a pronounced improvement in the classification efficacy with the optimization techniques at low dimensionality. As the dimensionality increases, there are regions with both approaches where the optimized values degrade indicating that the automated clustering breaks down. With the alignment approach, the degraded results remain better than the baseline while, for the semi-alignment technique, at 64 features the results are worse than for the baseline approach. We again note that, from our previous results, these were not the best scoring techniques for KL and would not be the approaches selected. The minimax and sup norm results are in Figures 8.19 and 8.20, respectively. Here we see the same trends that were found with the KS approach. The semi-alignment and alignment
hyperparameter selection with the addition of the option for minimax and sup norms results in marked improvement over the baseline KL technique.

For the measured data set, both the semi-alignment and alignment approaches provide better estimates for the value of $\sigma$ as compared to training over a pre-specified range of values and selecting $\sigma$ based on the best PCC. Recall that with both of the automated techniques, an optimal $\sigma$ value is determined for each class while the iterative approach selects the best overall value for $\sigma$ (all classes are restricted to a single, common, best overall value). The semi-alignment and alignment approaches performed similarly with semi-alignment classification efficacy higher at lower dimensionality and alignment better at higher dimensionality. We note that this is a three-class case; a trial with more classes most likely would begin to demonstrate differences in semi-alignment and alignment due to the inherent grouping of dissimilar classes into a world class by alignment. Indeed, we previously saw a false induction of similarity with the alignment approach for the quadboard data case as evidenced by a decrease in classification performance.
Figure 8.13 Comparison of classification results of σ selection techniques for three-class measured data set with KS L1 scoring technique.

Figure 8.14 Comparison of classification results of σ selection techniques for three-class measured data set with KS L2 scoring technique.
Figure 8.15 Comparison of classification results of $\sigma$ selection techniques for three-class measured data set with KS minimax scoring technique.

Figure 8.16 Comparison of classification results of $\sigma$ selection techniques for three-class measured data set with KS sup scoring technique.
Figure 8.17 Comparison of classification results of σ selection techniques for three-class measured data set with KL L1 scoring technique.

Figure 8.18 Comparison of classification results of σ selection techniques for three-class measured data set with KL L2 scoring technique.
Figure 8.19 Comparison of classification results of $\sigma$ selection techniques for three-class measured data set with KL minimax scoring technique.

Figure 8.20 Comparison of classification results of $\sigma$ selection techniques for three-class measured data set with KL sup scoring technique.
8.5 Comparison of Bayesian Classifier and SVM

We now consider the effects of the classifier on overall classification efficacy. Can we simply use a Bayesian classifier and omit the added complexity of the SVM? Why use the SVM at all? We address these questions by examining three cases of classification performance: the SVM with $\sigma$ selected based on classification results of trained SVMs, the semi-alignment hyperparameter selected SVM, and the Bayesian classifier. The results are presented in Figures 8.21 and 8.22. Comparing the iteratively derived $\sigma$ SVM hyperparameter results and the Bayesian classifier results, we see that the performance at low dimensionality is superior with the Bayesian classifier. As the number of features increases, the Bayesian classifier results degrade. Recall that the number of training samples is 120 per class. Generally, $5N$ training samples are required for adequate sample support. We would be on the edge of adequate sample support with 32 features where 160 samples would be adequate. At 64 features, the Bayesian results show a marked degradation. For 128 features, we no longer have adequate sample support rendering the classifier ineffective. The SVM classifier exhibits an inherent reduction in dimensionality, which eliminates the sample support issue. Using the semi-alignment version of the SVM results in the best classification performance of the three techniques at all numbers of features. With an optimally selected value for the kernel width, the SVM performance exceeds the Bayesian results. Although the Bayesian classifier gives the minimum error when the prior probabilities and
class densities are known and there is adequate sample support, we do not have this information for our data set resulting in superior performance with the *semi-alignment* SVM approach.

![KS min](image)

Figure 8.21 Comparison of SVM and Bayesian classifier results for the KS basis selection technique with the minimax norm.
Figure 8.22 Comparison of SVM and Bayesian classifier results for the KS basis selection technique with the sup norm.

When class densities fall on top of each other as in the checkerboard data set (Figure 8.23), the classes are not separable with the quadratic Bayesian classifier. This can be seen by simply observing a scatter plot of the data with a Bayesian decision boundary for one of the random trials (Figure 8.24). The SVM automatically generates the decision boundary in a higher dimension, which results in class separation when projected back into the original feature space. We note that the SVM makes no assumptions about the underlying structure of the data.
Figure 8.23 Class-conditional densities for the two features for the checkerboard problem. (a) Feature 1. (b) Feature 2.

Figure 8.24 Checkerboard two-class problem is nonseparable with Bayesian quadratic classifier.
CHAPTER 9

CONCLUSIONS

In order to determine separability of the targets, a suite of features characterizing the signal is utilized. Determining the optimal set to use is critical. Using more features than is necessary can have the effect of adding noise and lead to poor results, in addition to adding undesired processing time for classification. Selecting features that do not separate the targets well obviously leads to poor results. We examined two general approaches to feature extraction. The first used classical feature extraction techniques and served to provide the motivation for the second approach, an automated wavelet packet best-basis technique. In addition, we investigated two novel schemes for optimization of SVM kernel hyperparameter selection. Our automated semi-alignment approach provided a viable method for determining the hyperparameter.

The most significant results of the research proved to be the KS best-basis selection approach due to the improved performance at reduced dimensionality as compared to the baseline LDB approach. The automated semi-alignment approach we developed resulted in improved performance over alignment for greater than two-class cases. We briefly review the results of the research.

For the heuristic feature extraction approach, a suite of potential statistics/features was extracted from the training data, and the classification efficacy
was measured for the targets of interest. Features were selected by plotting histograms of the features for all targets and comparing them across all aspect angles. Those features that appeared to have the best separation between classes for the full training set were selected as candidates. Secondly, single-feature classification efficacy was considered. Combinations of good single-feature performers were grouped and found to contribute to degraded performance as compared to the single-feature case. This effect of "two best" features not combining to the "best two" features has been well documented [22]. A brute force method was used to select the best set of features. The features that performed best were found to be derived from both single and full polarization signal returns. For the targets investigated, fully polarimetric signatures provided discriminatory information not available with single polarization data. This is significant in terms of the design of the hardware. Features selected exhibit aspect-angle independence in their exploitation of the differences between the classes of data. A different set of features might well be selected if the classes of interest change.

As noted, the selection of the feature sets for testing the performance data was done by observing target type separation via histograms of the features at all 360° aspect angles. Segmentation of the histograms into smaller aspect-angle groups could result in increased knowledge for selection of the features. Additional effort in this area could potentially result in robust classification schemes based on more advanced techniques including fusing of current approaches as well as the
development of new approaches. Additional classifier types might be considered. References [21], [31] and [56] provide some classification schemes that may be of interest. However, overall, the classification results with these classical feature extraction techniques were unsatisfactory and served to provide the motivation for the wavelet best-basis investigation.

Our KS basis selection and feature extraction approach resulted in classification performance that was improved over both the heuristic approach and over the baseline wavelet approach of LDB. Our approach modified the LDB method by utilizing a CEDF to retain the distributional information of the training data wavelet packets for determining the basis. Additionally, we maintained a matrix of class pair-wise scores that allowed for some flexibility in pruning the wavelet packets and thus the basis selected. When comparing the wavelet-based approaches to the more traditional heuristic feature extraction techniques, we found a number of potential advantages. These included the ability to handle increased threat sets and to free the feature estimation expert from the mechanics of the classification process, as well as improved classification performance. In this best-basis research, we developed a novel alternative to the LDB basis selection process, while attempting to quantify the classification efficacy of both LDB and our new technique for identification of vehicles from high-resolution radar signatures. Our general problem was extracting features from a large data set in order to discriminate between classes of objects. The raw data was mapped from a high-
dimensional data space via the wavelet packet basis functions to an orthonormal feature space.

The performance of the new KS discriminant based feature extraction technique, along with the normalization approaches for the scoring matrix, proved to be superior to LDB at low dimensionality as evidenced by the improvement in the classification performance. Our technique becomes less significant as the number of selected features increases, with both techniques converging to identical classification results at full dimensionality. Convergence of classification efficacy as the number of wavelet features increases was also reported by Saito et al. [64] when comparing LDB with the KL based discriminant function. In our research, we modified Saito’s KL approach to include the alternate scoring normalizations and found comparable results to our new KS technique. These developments are significant in application environments where the benefits of working in a reduced feature space lead to implementations for real-time operations.

The most significant result of this research, from the author’s perspective, lies in the automation of SVM hyperparameter selection. Our investigation into SVM kernel hyperparameter optimization found that, while minimizing the number of support vectors used to define the hyperplane provides a method of kernel hyperparameter selection, our second approach based on a variation of the work of Cristianini et al. provides a strategy for unsupervised kernel optimization that eliminates iterative SVM training. Our technique begins with a similarity (kernel)
matrix. We form the similarity matrix with an initial guess for $\sigma$ and, using a Frobenius norm, begin a search for the value of $\sigma$ that forms the best within-class clusters as compared to the between-class clusters. This entire process occurs prior to training with no iterating on the hyperparameters in the training stage. Unlike the work presented in [25], we work on a subset of the similarity matrix thus removing the induction of a false bias from clustering disparate classes as a single class. With our Frobenius functional minimization, we are able to identify the Gaussian kernel width hyperparameter without iterative training of the SVM, resulting in significant computational savings. This approach allows unsupervised evaluation of the hyperparameter, resulting in an automated procedure where the expert "in the loop" can be removed, thereby providing a simple and efficient approach for SVM hyperparameter optimization. Selection of the kernel hyperparameters is critical to the performance of SVMs, directly impacting the generalization and classification efficacy of the SVM. An automated procedure for hyperparameter selection is clearly desirable given the intractable problem of exhaustive search methods.
## APPENDIX

### ACRONYMS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASH</td>
<td>average shifted histograms</td>
</tr>
<tr>
<td>ATR</td>
<td>automatic target recognition</td>
</tr>
<tr>
<td>DFT</td>
<td>discrete Fourier transform</td>
</tr>
<tr>
<td>ECDF</td>
<td>empirical cumulative distribution function</td>
</tr>
<tr>
<td>FT</td>
<td>Fourier transform</td>
</tr>
<tr>
<td>FTE</td>
<td>far target extent</td>
</tr>
<tr>
<td>HPF</td>
<td>high-pass filter</td>
</tr>
<tr>
<td>IDFT</td>
<td>inverse discrete Fourier transform</td>
</tr>
<tr>
<td>ISAR</td>
<td>inverse synthetic aperture radar</td>
</tr>
<tr>
<td>KL</td>
<td>Kullback-Leibler</td>
</tr>
<tr>
<td>KLT</td>
<td>Karhunen-Loève transform</td>
</tr>
<tr>
<td>KS</td>
<td>Kolmogorov-Smirnov</td>
</tr>
<tr>
<td>LDB</td>
<td>local discriminant basis</td>
</tr>
<tr>
<td>LPF</td>
<td>low-pass filter</td>
</tr>
<tr>
<td>MMW</td>
<td>millimeter wave</td>
</tr>
<tr>
<td>NTE</td>
<td>near target extent</td>
</tr>
<tr>
<td>PCA</td>
<td>principal component analysis</td>
</tr>
<tr>
<td>PCC</td>
<td>percent correct classification</td>
</tr>
<tr>
<td>PDF</td>
<td>probability density function</td>
</tr>
<tr>
<td>QMF</td>
<td>quadrature mirror filter</td>
</tr>
<tr>
<td>SAR</td>
<td>synthetic aperture radar</td>
</tr>
<tr>
<td>SCR</td>
<td>signal-to-clutter ratio</td>
</tr>
</tbody>
</table>
REFERENCES


