9-1-2008

A Confidence Paradigm for Classification Systems

Nathan J. Leap

A CONFIDENCE PARADIGM FOR CLASSIFICATION SYSTEMS

DISSERTATION

Nathan J. Leap, Captain, USAF

AFIT/DS/ENS/08-02

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense or the United States Government.

AFIT/DS/ENS/08-02

A CONFIDENCE PARADIGM FOR CLASSIFICATION SYSTEMS

DISSERTATION

Presented to the Faculty
Department of Operational Sciences
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy

Nathan J. Leap, B.S., M.S.
Captain, USAF

September 2008

AFIT/DS/ENS/08-02

A CONFIDENCE PARADIGM FOR CLASSIFICATION SYSTEMS

Nathan J. Leap, B.S., M.S.

Captain, USAF

Approved:

Dr. Kenneth W. Bauer Jr. Committee Chair	Date
Dr. Mark E. Oxley Committee Member	Date
Dr. John O. Miller Committee Member	Date

Accepted:

M.U. Thomas Dean, Graduate School of Engineering and Management	Date

Table of Contents

List of Figures

List of Tables

Abstract

There is no universally accepted methodology to determine how much confidence one should have in a classifier output. This research proposes a framework to determine the level of confidence in an indication from a classifier system where the output is or can be transformed into a posterior probability estimate. This is a theoretical framework that attempts to unite the viewpoints of the classification system developer (or engineer) and the classification system user (or warfighter). The developer designs and tests the classification system at a macro-level. The user fields the system in an environment often quite different that those used to develop the system. The user operates at a micro-level and is interested in the indications as they are made by the system. The paradigm is based on the assumptions that the system confidence acts like, or can be modelled as a value and that indication confidence can be modelled as a function of the posterior probability estimates. The viewpoints of the developer and the user are unified through the proposition that the expected value of the user's confidence should be approximately equal to the developer's confidence.

We choose a quadratic function to represent the ascent from zero confidence to absolute confidence. The tactical issues involved with fitting such a curve are addressed in this research. Once we have fit the appropriate quadratic function to the distribution of the posterior probability estimates, we have equated the engineering confidence to the user confidence. Two methods of applying the new classifier (the forced decision classifier with the confidence function overlaid) are suggested. One could await a posterior probability estimate output from the classifier and then make a declaration decision based on a random number draw as compared to $P(DEC|p)$. Most users would find the idea of "rolling the dice" on the declaration undesirable and so a second alternative is suggested. The user could consider a sequence of classifiers

devised simply from the application of a threshold on the confidence function. This creates a rational portfolio of classification systems based on the user's confidence function. Each member of the portfolio has a non-declaration rate and associated recalculated engineering confidence.

The introduction of the non-declaration possibility induces the production of a higher-level value model that weighs the contribution of engineering confidence and associated non-declaration rate. Now, the task becomes to choose the appropriate threshold to maximize this overarching value function. This paradigm is developed in a setting considering only in-library problems, but it is applied to out-of-library problems as well. Introduction of out-of-library problems requires expansion of the overarching value model. This confidence measure is a direct link between traditional decision analysis techniques and traditional pattern recognition techniques. This methodology is applied to multiple data sets, and experimental results show the sort of behavior that would be expected from a rational confidence paradigm.

Acknowledgements

There is absolutely no way I would have finished this program alone. I am thankful to everyone who have made it possible for me to finish this program.

To my advisor, Dr. Kenneth Bauer: Thanks for taking me on as a student (for the second time) and guiding me through the process. Your experience and expertise was priceless in getting me through AFIT, and you made this program an enjoyable one.

To my committee, Dr. J.O. Miller and Dr. Mark Oxley: Thanks for your guidance on my dissertation.

To Capt Mike Turnbaugh, Mr Scott Percival, and Maj June Rodriguez: You made AFIT a fun place to be everyday. You kept me sane and were there to help me through the course work as well as the research. Thanks for everything.

To my wife: You truly are my best friend, and I am so blessed to get to walk through life with you. I can't thank you enough for all the love and support you give me everyday. I would never have made it through this without you. Thank you and I love you.

To my children: Do you know what I love about you? Everything. You two make me smile and laugh every single day. Thank you for making each day a new adventure.

Finally, and most importantly, I need to thank my Lord and Savior, Jesus Christ, who has blessed me in so many ways. "But blessed is the man who trusts in the LORD, whose *confidence* is in him" (NIV, Jeremiah 17:7) [1]. You guide and protect me everyday. I would never have completed this program without Your daily intervention in my life. "This is the *confidence* we have in approaching God: that if we ask anything according to his will, he hears us" (NIV, 1 John 5:14) [1].

Nathan J. Leap

A CONFIDENCE PARADIGM FOR CLASSIFICATION SYSTEMS

1. Introduction

1.1 Classification Problem and Confidence Definitions

In general, a classification problem is a problem where it is of interest to assign an object to one of a predetermined number of classes based upon features from measurements of the object. In addition to knowing which class the object belongs, it is also of interest to know how confident one should be in that assignment.

While "confidence" is a widely used term, very few people can actually define it. According to Webster's Dictionary [2], "Confidence is a feeling of trust (in someone or something)." Wikipedia [3] defines it by saying "Confidence is trust or faith that a person or thing is capable." Confidence is an age-old concept; all the great philosophers (Socrates, Plato, Aristotle, Plotinus, St Augustine, St Aquinas, Machiavelli, Descartes, Hobbes, Locke, Rousseau, Kant, Marx, Mill, Confucius) discuss having confidence in many different ideas, but none of them actually define confidence. Aristotle comes the closest to defining confidence when he discusses courage being the mean between fear and confidence [3]. Still, he does not specifically define confidence. The common thread throughout these definitions is that confidence is a measure of how much one can trust something or somebody.

In statistics, confidence is discussed among both frequentists and Bayesians. Frequentists typically measure the probability of the data given a model, and Bayesians typically measure the probability of a model given the data [23]. From a frequentist point of view, a 95 % confidence interval means that if the experiment was run 100 times, the true parameter of interest would fall inside the interval approximately 95 times [64]. From a Bayesian point of view, the posterior probability is often used as a measure of confidence. This probability has an intuitive meaning. A high value indicates that there is a high probability that the model is correct (and thus, a high level of confidence should follow) [23]. The problem with simply using the posterior probability is that, in practice, these are hard to estimate accurately [51]. When evaluating a classifier or classifier ensembles, most of the commonly used measures are based upon the performance of the classifier on the entire data set. The main objective of this research is to develop a confidence paradigm for a classification system; one useful product of this confidence paradigm is a measure which can be applied to the individual declarations of a classification system.

1.2 Combat Identification and Automatic Target Recognition

Following the argument laid out in Laine [40], as the United States Air Force continues to become more and more technologically advanced, it is able to effectively kill any object identified as a target. Air Force Doctrine Document (AFDD) 2-1: Air Warfare [63] states that "if the enemy's key targets, target sets, or centers of gravity

(COGs) can be found and identified, they are usually within airpower's reach." Of course, identification is simply one link in the kill chain; this kill chain may include processes such as search, detect, track, classify, identify, assign, weapon launch, target acquisition, target damage, and kill assessment [26]. "Combat identification (CID) is often viewed as the weakest link in the military's kill chain [26]." As this is the considered the weakest link, CID needs improvement the most.

Sadowski [57] broadly defines CID as "the process of attaining an accurate characterization of detected objects in the joint battlespace to the extent that high confidence, timely application of tactical military options and weapons resources can occur." At the heart of CID is making high confidence declarations.

Laine [40] lays out two types of CID: cooperative and non-cooperative. Identification, friend or foe (IFF) systems are an example of cooperative CID. IFF systems involve the communication between two friendly electronic systems. When feedback from one of the systems is not obtained, non-cooperative CID must be implemented. Non-cooperative CID can be further broken down into two categories: man-in-the-loop and autonomous. A non-cooperative man-in-the-loop system is one where a human makes the final decision of whether a target is a friendly or hostile. A non-cooperative autonomous system is one where this decision is made automatically without human intervention; such a system is an automatic target recognition (ATR) system [40]. An ATR system seeks to automatically recognize targets as friendly or hostile (or a more specific type of target). There are two additional labels that may

be used by ATR systems. If a system is not confident enough in the information available, a non-declaration status can be used. This delays the system from making a declaration until more evidence is obtained (maybe via classifier fusion) [40]. In addition to a non-declaration status, an out-of-library status may also be used. Any target type that an ATR system has observed before (or has been used to train the classifier system) is considered an in-library target type; any target type that an ATR system has not observed before (or has not been used to train the classifier system) is considered an out-of-library target type [4]. Since it is quite possible that an ATR system may encounter a target type that has not been seen before, out-of-library status allows an ATR system to account for this occurrence in the combat environment [56].

1.3 Research Goal and Application Areas

Ho [27] and Melnik *et al.* [46] discuss three types of classifier outputs: abstract (single class declaration), rank (ordered list of class membership), and measurement (magnitude of class selection). The goal of this research is to develop a confidence paradigm for classification systems; one product of this paradigm is the confidence in the indication of a classification system. Specifically, this research is concerned with classification systems that produce measurement level outputs that are or can be transformed into posterior probability estimates.

The paradigm is demonstrated on a synthetic problem and two real-world classification problems. Further, the paradigm is applied to an ATR problem. The main goal of an ATR system is to identify hostile targets that can be exploited by friendly forces. In such an ATR system, it would be useful to not only identify targets of interest but also to specify how much confidence the system has that what it has just identified is a target. The warfighter is interested in a measure that expresses the confidence that a system has in any given indication [53].

1.4 Contributions of this Research

This research makes several contributions as it addresses the overarching research goal. First, this research develops a confidence paradigm that encompasses and generalizes current practices. One result of this paradigm is a new confidence quantifier that unites traditional decision analysis techniques and current pattern recognition techniques. The main axiom of this contribution is as follows: the class specific confidence we have in the output of a classifier, on average, should be approximately equal to the confidence we have in the classifier operating on that class. This research develops a methodology to determine the parameters of a generalized confidence function within the confidence paradigm. One novel application of the generalized confidence function is a new non-declaration methodology using a stochastic implementation.

A new use of the popular Kullback-Liebler distance to determine how well a classifier generalizes to an independent data set is developed. This research demonstrates the effectiveness of a new out-of-library detector and demonstrates the new concept of an out-of-library non-declaration. The new paradigm unites multiple performance measures under an overarching value model that leads to choosing the optimal operating point.

An algorithm is developed that minimizes a "confidence" measure called Binned Error in the Posterior (BEP). Then, we prove that training a classification system using back-propogation to minimize sum of squared error also minimizes BEP. This allows us to show that minimizing this confidence measure leads to outputs of the classification system which are, in the limit, posterior probabilities.

1.5 Organization of the Dissertation

The following is the organization of the dissertation. Chapter 2 provides a summary of the current literature as it relates to confidence in a classifier indication. In the literature, we have seen that posterior probability estimates or constructs known as confidence scores are employed as confidence measures [49, 53]. There is a need to measure how well these confidence measures perform across a number of exemplars; one example of this, as put forth by Parker *et al.* [49], is the confidence error. Ross [53] makes the statement that under the discrete form of confidence error, a confidence score is ideally a posterior probability. We present an algorithm that

minimizes the confidence error. We show that training a multiple layered perceptron (MLP) neural network using back-propagation to minimize sum of squared error also minimizes the confidence error in Chapter 3. This allows us to show that, when training a classification system using back-propogation to minimize confidence error, the outputs of the classification system are posterior probability estimates.

We develop a confidence paradigm in Chapter 4. This is a theoretical framework that attempts to unite the viewpoints of the classification system developer (or engineer) and the classification system user (or warfighter). The developer designs and tests the classification system at a macro-level. The user fields the system in an environment often quite different that those used to develop the system. The user operates at a micro-level and is interested in the indications as they are made by the system. The paradigm is based on the assumptions that the system confidence acts like, or can be modelled as a value and that indication confidence can be modelled as a function of the posterior probability estimates. The viewpoints of the developer and the user are unified through the proposition that the expected value of the user's confidence should be approximately equal to the developer's confidence.

This all occurs in a setting where the classifier is developed as a forced decision tool. The developer has attached confidence, which we model as value, to this classifier. The developer typically characterizes system performance based on statistics such as averages. The user does not live in a world of average performance. The user must make decisions based on the indications of the classifier and wants

to make these decisions with high confidence. Implicit in all this is the fact that a forced decision classifier that bases its decisions on the values of estimated posterior probabilities attaches (even if unconsciously) equal confidence to all its decisions. A notional histogram along with the implied forced decision confidence function is shown in Figure 1.1.

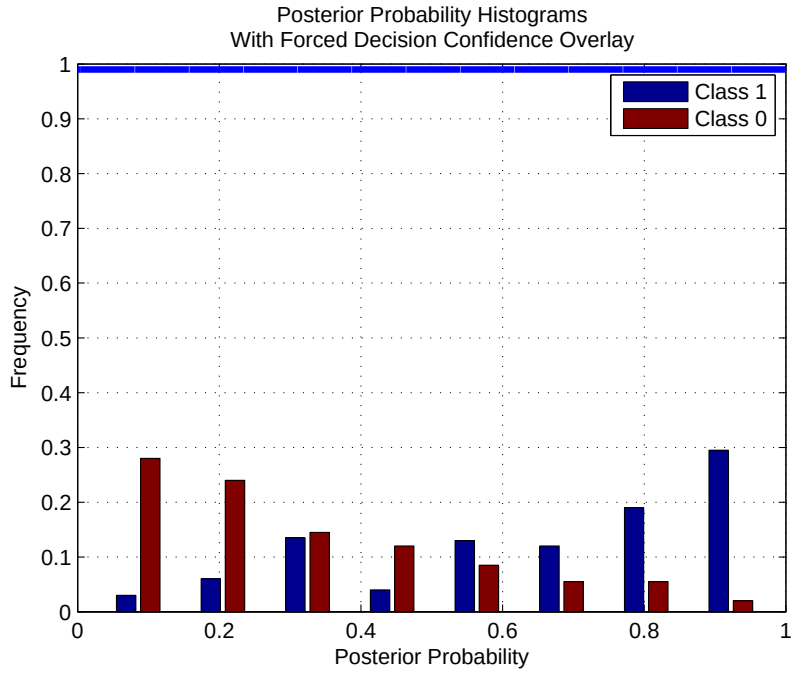

Figure 1.1 Notional Histogram with Forced Decision Confidence Function.

Standard practice is to elevate the performance of the classifier by not allowing decisions based on some range of lower magnitude posterior probability estimates. This philosophy leads to the rejection region concept. Essentially, decisions based on posterior probability estimates below some pre-determined threshold are deferred. Here, two levels of confidence are apparent: none and absolute. In this scenario, the confidence function can be thought of as simply the probability of a declaration

given a posterior probability estimate, $P(DEC|p)$. A notional histogram along with an implied rejection region confidence function is shown in Figure 1.2.

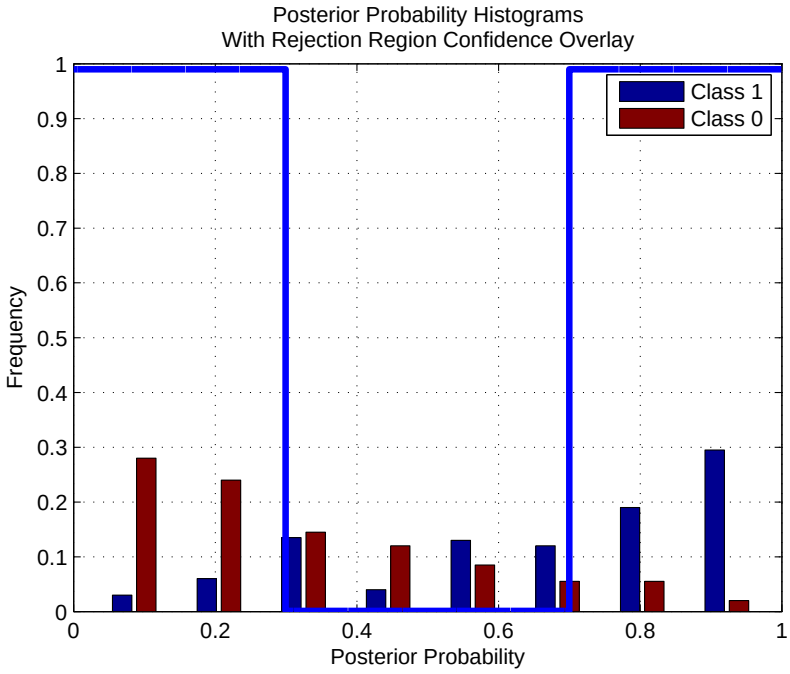

Figure 1.2 Notional Histogram with Rejection Region Confidence Function.

Believing that a middle ground exists between these extremes, we choose a quadratic function to represent the ascent from zero confidence to absolute confidence. A notional histogram along with a quadratic confidence function is shown in Figure 1.3.

The tactical issues involved with fitting such a curve are addressed in Chapter 4. Once we have fit the appropriate quadratic function to the distribution of the posterior probability estimates, we have equated the engineering confidence to the user confidence. Two methods of applying the new classifier (the forced decision classifier with the confidence function overlaid) are suggested. One could await a posterior

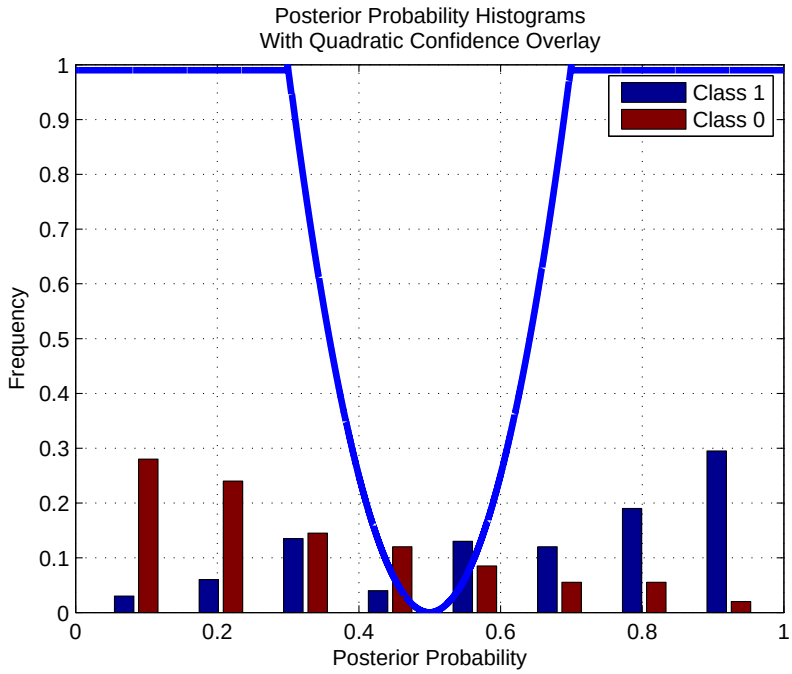

Figure 1.3 Notional Histogram with Quadratic Confidence Function.

probability estimate output from the classifier and then make a declaration decision based on a random number draw as compared to $P(DEC|p)$. Most users would find the idea of "rolling the dice" on the declaration undesirable and so a second alternative is suggested. The user could consider a sequence of classifiers devised simply from the application of a threshold on the confidence function. This creates a rational portfolio of classification systems based on the user's confidence function. Each member of the portfolio has a non-declaration rate and associated recalculated engineering confidence. The introduction of the non-declaration possibility induces the production of a higher-level value model that weighs the contribution of engineering confidence and associated non-declaration rate. Now, the task becomes to choose the appropriate threshold to maximize this overarching value function. This

overarching value model is addressed in Chapter 6. This paradigm is developed in a setting considering only in-library problems, but it can be applied to out-of-library problems as well. The addition of out-of-library problems is presented in Chapter 5 along with a demonstration of out-of-library non-declarations. Once out-of-library problems are added, the higher-level model is expanded; this expanded overarching value model is presented in Chapter 6. Experimental results are documented in Chapter 7. Contributions and conclusion of this research as well as items for future work are presented in Chapter 8.

2. Literature Review

2.1 Literature Overview

There is an extensive literature regarding classification confidence, but there seems to be no methodology that is universally accepted. Many articles argue that confidence acts like a probability and thus, conforms to the axioms of probability while others argue that confidence follows some other paradigm. Many of these articles treat confidence simply as some real number. Others restrict this idea of confidence to a number in the closed interval $[0, 1]$. Still others treat confidence as a binary value. There is also literature available that may aid in testing potential confidence measures. Fusion is a popular technique that is thought to increase confidence, and some methodologies place restrictions on the inputs to the fusion (e.g., only perform fusion on classifiers that produce a label or a rank). Finally, there are some miscellaneous articles that are on the periphery of this research effort. The following section summarizes the current literature as it relates to this research effort.

Richards *et al.* [51] describe the basic need for a confidence measure. Often, classifier systems are judged as a whole by using measures such as true positive rate, false positive rate, and Receiver Operating Characteristic (ROC) curves. However, it is also useful to have a way to measure how confident the system is on any given indication. Much of the current literature, [6, 9, 12, 41, 65, 66], uses either a posterior probability or something similar to a posterior probability to measure confidence

in a given declaration. However, there are problems with simply using posterior probabilities. Roberts [52] states that the problem in using posterior probabilities as a measure of confidence is that they sum to 1; this is a problem because if a "rogue" data point is evaluated, it is treated as if it is a "genuine" data point and thus, is classified with "apparent confidence" into one of the output classes. In the ATR application, this is the problem with out-of-library targets. Richards *et al.* [51] states that using a posterior probability is troublesome because this requires specification of the prior probabilities which are often unknown. Ross and Minardi [53] state that posterior probabilities are particulary difficult for classification systems to estimate. Because of these problems, researchers continue to search for a better confidence measure for classifier declarations.

The remainder of the chapter is organized as follows. Section 2.2 discusses confidence paradigms in the literature. Section 2.3 reviews those confidence measures where the measure is some real number. Section 2.4 discusses those confidence measures where the measure is restricted to $[0, 1]$. Section 2.5 covers those confidence measures where the measure is a binary value. Section 2.6 discusses some of the literature as it relates to testing a confidence measure. Section 2.7 discusses classifier fusion as it relates to confidence. Section 2.8 covers some other literature related to this research.

2.2 Confidence Paradigms

There is no universally accepted confidence paradigm in the literature. The most popular paradigm to evaluate confidence is as a probability, [6, 41, 46, 65, 49, 66, 51, 9, 53, 12, 30, 61]. Thus, each of these papers models confidence in a fashion such that confidence to conform to the axioms of probability. Goh *et al.* [37] state that the confidence in an exemplar is proportional to the maximum posterior probability across the possible classes; however, more information than just the posterior probability is needed for an accurate confidence estimate. Other authors treat confidence as something other than a probability. Jaeger [31] treats confidence and information as identical and discusses confidence as the trust one should have in a classifier. Therefore, confidence does not necessarily conform to the axioms of probability. Also, Tubbs and Alltop [62] discuss confidence as "probabilities, beliefs, or any suitable weight" and define a measure which does not sum to one across the possible classes (one of the axioms of probability). Each of these paradigms is discussed in further detail in subsequent sections.

2.3 Confidence as a Real Number

One approach to measuring confidence is to devise a measure whose range set is the set of all real numbers. These values have no absolute meaning since there is no lower or upper bound on the measure. Atukorale and Suganthan [7] use the idea of overlapping networks and a confidence measure from each of these networks

to classify handwriting samples. The confidence measure for classifier j is given by $c_j = (1 - \frac{d_j}{d_{acc}})$ where d_j is the minimum distance between the data point and all classes for the classifier j, and d_{acc} is the sum of d_j over all j. If $C_i = [c_j]$, then a confidence vector is calculated for each of the i overlapping networks. The overall confidence of the classifier system is found by adding the confidences from each of the i confidence vectors. The individual confidence vectors will sum to one when summing across classes, but the overall confidence vector will sum to i.

Kimball and Rothkopf [36] derive a measure of confidence in the field of speech recognition that can reliably predict when an utterance classifier has made a correct decision. The authors state that the main purpose of this measure is to aid in deciding when to use a more expensive, but also more powerful classifier. In order to represent an utterance, it is coded into a string of characters in some feature space. Given a set of k known strings and an unknown string U, it is desirable to find which known string is most similar to U. Let D_1 denote the minimum hamming distance between the unknown string and all the known strings. Let D_2 denote the second smallest hamming distance between the unknown string and all the known strings. Then, the measure $R = \frac{D_2}{D_1}$ is their measure of confidence. If R is small enough, then there is little confidence in the classification. If R is large enough, there is more confidence in the classification. This ratio will always be positive, but there is no upper bound on this measure. If an exact match occurs, $D_1 = 0$, and R is undefined. However, the authors do not address this case.

Roberts [52] looks into some of the issues that go along with estimating confidence limits in a feed-forward neural network. For a committee of classifiers, the author uses the confidence measure $c = exp(-\varepsilon[E])$ where $E = (y_m - t)^2$, $\varepsilon[E] = var[y_m] + (\varepsilon[y_m] - t)^2$, y_m is the largest output of the committee of classifiers, and t is the target value for y_m. This is essentially the negative exponentiation of the expected squared error of the committee of classifiers.

Jaeger [31] does not combine the confidence value from the individual classifiers, but instead, the author uses the information contained in this confidence value as measured by Shannon's notion of information. This method implies that confidence and information are identical. The author describes confidence as the trust that a classifier engenders in its recognition results. As the probability that the class label is correct increases, so does the confidence in that class label. Confidence values contain information that aid in classification. The estimate of the confidence for classifier i is given by: $\hat{K}_i = -E(R) * \ln(1 - \hat{p}(K_i))$ where $E(R) = R^{\frac{-1}{\ln(1-R)}}$, $\hat{p}(K_i) = E(\frac{\ln(1-R_i)}{\ln(1-R)})$, R_i is a partial recognition rate found by taking the recognition rate on all exemplars with confidence value $K \leq K_i$, and R is the overall recognition rate. The author uses the sum rule as the fusion rule to use in overall classification.

Ho *et al.* [28] discuss three methods for classifier fusion where the output of each classifier is a ranking of the classes from most likely to least likely. For each method, a confidence score is calculated as a function of the ranks assigned to each class. Only the relative magnitude of the confidence scores is considered here, and the

value of the score in absolute terms has no meaning. The three methods discussed are the highest rank, the Borda count, and the generalized Borda count. As an example, consider the fusion of three classifiers on a three class problem. Let C_{ij} be the rank that classifier i assigns to class j. Let $C_1 = [1\ 2\ 3]$, $C_2 = [2\ 1\ 3]$, and $C_3 = [1\ 2\ 3]$. The highest rank method is easy to implement, but there could be many ties that need to be arbitrarily broken. The only output of the highest rank method, HR, is the actual rank. In our simple example, the highest rank outputs $HR = [1\ 1\ 3]$. This illustrates the point of a tie that needs to be broken. The Borda count for class c, counts the number of classes ranked below c for each classifier by summing the ranks across the classifiers. The magnitude of the Borda count, BC, indicates the strength of agreement by the individual classifiers. In our simple example, the $BC = [3\ 3\ 9]$ The generalized Borda count, BC_G, weights the individual classifiers Borda counts, and these weights are determined through logistics regression. For our example, the generalized Borda count will need three weights, $w_1 = 0.5$, $w_2 = 0.25$, and $w_3 = 0.25$. Then, $BC_G = [1.25\ 1.25\ 3]$ It seems that the Borda counts could be used to measure the confidence in a declaration easier than the highest rank. While none of these measures directly measure confidence, they could be used indirectly to measure confidence.

2.4 Confidence Restricted to [0,1]

Another approach to developing a confidence measure is to limit the range of the confidence measure to the closed interval $[0, 1]$. This limitation is nice because it allows for absolute interpretation of the measure. As stated in [51], this means that an analyst need not know the inner workings of the measure since it has absolute meaning. By definition, any measure that treats confidence as a probability falls into this category. Arribas and Cid-Sueiro [6] state right up front that a posterior probability can be used as a measure of confidence in a decision. Weintraub *et al.* [66] use the posterior probability as the measure of the confidence in a word hypothesis. Goh *et al.* [37] state that confidence is not equal to the posterior probability; instead, it is proportional to the maximum posterior probability across the possible classes. This research was done with support vector machines as the classifier. In this paradigm, more information (from the support vector machine in this research) than just the posterior probability is needed for an accurate confidence estimate.

Wan [65] states the posterior probability can be thought of as a measure of confidence in any indication. In addition, Wan suggests alternate measures of confidence. One is the Kullback-Leibler distance which is a measure of relative entropy, but this requires the true posterior probability distributions. The author says that a more practical measure of confidence is the entropy of the posterior probability.

Lane *et al.* [41] examine the effects on utterance detection when an utterance is not within the limited domain of the system. This is similar to an out-of-library indication in ATR. A set of SVMs is used to classify the utterances into topics so the output is a set of confidence scores, $C(t_j|W)$, for each topic j. Then the topic confidence scores are combined to get a speech recognition result using a weighted linear combination of the topic confidence scores. Essentially, this paper uses something like a posterior probability as a measure of confidence.

Baraghimian [9] discusses the fusion of confidence values for multiple characters in a word in order to get a confidence in the entire word. While the confidence measure is a value between 0 and 1, calculating the confidence value is not discussed, but they look something like a posterior probability. One idea is to take an average confidence across all characters to get a confidence value for the word. Then, one could compare confidence scores across the possible words.

Callari and Ferrie [12] describe a paradigm common to active object recognition: gather data; build a model of the environment; attempt recognition; when recognition is ambiguous, use context information to take further measurements. In particular, the authors are interested in the problem where there is uncertainty/error in the inputs to binary classification. The authors state that the classifier produces a set of conditional probabilities. Based upon the ambiguity set forth from these posterior probabilities, the authors seek to find an efficient way to collect more data in order to reduce this ambiguity.

Melnik *et al.* [46] consider confidence and probability as one and the same. Thus, they assume confidence to be a probability. The authors are interested in classifier systems where the output of each classifier is a class ranking. This paper looks at highest rank, Borda count, generalized Borda count. In addition to these methodologies, the authors devise a new methodology which is a generalization of these three methogologies. Confidence in the lower ranks is a principle that says that there is a more significant difference between ranks 1 and 2 than there is between ranks 100 and 101.

Tubbs and Alltop [62] attempt to derive a measure of confidence for fused classifiers where the confidence measure can be easily interpreted. The input to the fusion scheme is a ranked list of class membership from each classifier, and the authors develop a nonparametric fusion scheme and a quantitative confidence measure from this data. "In this paper, a measure of confidence is any real-valued function whose values indicate the level of confidence associated with a decision. The measures could be represented as probabilities, beliefs, or any suitable weight." The fusion method is based on Friedman's procedure [21]. Then, the confidence measure is given by the probability of observing a ratio of successive ordered values. This measure of confidence is calculated for all but one class (it is necessary to eliminate one class to allow for a non-singular covariance matrix of order statistics). It is interesting that the measures of confidence for all classes do not sum to one; instead,

in the example, there were cases where one would be greater than 90% confident in two separate class labels.

Zhang *et al.* [68] apply confidence to computer science by trying to prune the size of a section of computer code with confidence that some faulty code has not been included in the reduction. The idea of pruning is to reduce the section of code that could contain the erroneous code. Thus, the authors want to reduce this code to as small a section as possible while not eliminating the actual erroneous code. In doing this, the authors identify their confidence measure by modelling their computer code as a directed graph with nodes and edges. If a section of the code is classified as correct, it has a confidence score of 1. If a section of the code is classified as incorrect, it has a confidence score of 0. If a section of the code is unknown, it has a confidence score of $1 - log_{|Range(v@n)|}|Alt(v@n)|$ where $v@n$ represents the value, v, for node n. Thus, $Range(v@n)$ is the allowable range of v, and $Alt(v@n)$ is the alternate values of v for which the same result would have been computed.

Richards *et al.* [51] states that a confidence measure should act like a probability in that it should be between 0 and 1 and sum to 1. Since using posterior probabilities requires specification of the prior probabilities which are often unknown, they base their confidence measure on likelihoods, rather than probabilities. One issue with this idea is that while the authors try to eliminate prior probabilities, they replace the prior probability of hypothesis i with λ_i. This looks suspiciously like a prior probability.

Huang and Suen [30] develop a method to combine classifiers when the output of the individual classifiers is a measurement value. The method is called Linear Confidence Accumulation (LCA). The authors describe a subjective confidence value and an objective confidence value. The authors describe a confidence value based on a distance or heuristic transformation function that may be very different from the real confidence value as a subjective confidence value. If the confidence value is derived from a probability function, the authors state this is an objective confidence value. Thus, the authors are assuming that objective confidence acts like a probability. Their method takes the individual classifier measurements, transforms them into an objective confidence measure (presumably through some probability density function) and then sums them linearly by class. Thus, the class corresponding to the largest accumulated sum is the class declared by the classifier ensemble. This method assumes three things: the measurements only contribute to confidence for a given class, classifier independence, and aggregation of confidence is a linear summation of individual confidence values.

Thomas and Allcock [61] develop a statistical confidence measure for use in remote sensing applications. In this application, every pixel in a scene needs classified with an associated confidence level. For classification purposes, a maximum likelihood classifier is used. Each pixel can only be classified correctly or incorrectly. For notation, p is the probability of getting one pixel correct, and q is the probability of getting one pixel incorrect ($q = 1 - p$). The confidence level is the

probability that at least i pixels are correctly classified from a random sample of n pixels. Using the Binomial Distribution, the probability of any pixel combination can be calculated. Then, using the fact that the Binomial Distribution can be approximated with the Normal Distribution in the limit, the confidence measure is developed using the standard normal tables (because the Binomial Distribution calculations "rapidly becomes tedious"). To make sure this approximation is used appropriately, only sample sizes greater than 50 are used. The measure takes into account the mean, standard deviation, standard error of the estimate of the mean, the standard error of the estimate of the standard deviation, and the experimental error. In the end, the authors can make statements like they are 99.9% confident that at least 94.5% of the pixels were correctly classified. To make this statement, the authors calculate a 99.9% confidence interval on the average correct number of pixels across a number of samples from the population where each sample contains at least 50 pixels. The lower confidence interval value in this case would be 94.5%.

Huang *et al.* [29] uses SVMs to find a confidence measure for word confidence. Words can be broken down into sub-words. Typically, word confidence is found by combining confidence levels from the sub-words. Phone-level confidence measures are a type of sub-word confidence measure. Larger words have a larger number of sub-words; hence, they will have a larger number of features than smaller words. Since words will have different numbers of phone-level confidence measures (or sub-words), different SVMs will be used based upon the number of phone-level confidence

measures. Then, a threshold can be applied to the output of the SVM. In general, this method performed better than more conventional methods in that it had a lower error rate.

2.5 Confidence as a Binary Value

Yet another approach to developing a confidence measure is to further limit it to be a binary value. Grunwald *et al.* [25] use high confidence (HC) and low confidence (LC) as the two labels that a given indication is correct (C) or incorrect (I). Black and Franklin [11] state that confidence indications are necessary in order to allow for more efficient parallel processing in computer science applications. When two calculations need to be performed and one calculation depends on a previous calculation, it is beneficial to estimate the value of the first calculation so this estimate can be used to perform the second calculation. Hence, these confidence measures are designed to predict or not predict.

Krzanowski *et al.* [39] and Bailey *et al.* [8] build on the concept of a rejection region implementing uncertainty envelopes (UE) that are associated with unsure classifications. Over 10,000 classifiers, the percentage of classifiers that correctly classified the same exemplar is the level of confidence for that exemplar. A user defined threshold is applied to form the UE. Any data falling into the uncertainty envelope is unsure, and any data falling outside the uncertainty envelope is sure. Thus, the confidence measure is essentially a binary indicator, either sure or unsure.

Delany *et al.* [15] state that while some classifiers produce numeric scores, those scores do not necessarily correlate well with the confidence one should have in that score. It is stated that Support Vector Machine outputs and Logistics Regression outputs do correlate well with confidence, but Naive Bayes, k-Nearest Neighbor, and Neural Networks do not correlate well. In this research, the authors are trying to classify an email as either spam or not spam. Each email will either be declared as confident or not confident. Thus, this is another binary confidence measure.

2.6 Confidence Testing

Some other researchers have provided some paradigms that may prove useful when testing a confidence measure. Parker *et al.*, [49] discuss area under the ROC curve (AURC) and confidence error (CE) measures for assessing ATR performance. AURC is fairly straight-forward, and it measures how well the classifier predicts the actual class of the exemplars. The authors state that the output of some classifier is a score between 0 and 1, and they assume that these scores are distributed according to a Beta distribution for both classes. CE is a measurement of how good the $[0, 1]$ score is. To compute CE, the outputs of the classifier are ranked, sorted, and placed into bins. Then, the average score is calculated in each bin. The authors state that if the average score in a bin is .20, then you would expect that 20% of the actual exemplars in that bin are targets and 80% are non-targets. To calculate CE, the difference between the mean score and the percentage of targets is calculated by bin. This value

is squared for all bins, summed across all bins, divided by the number of bins, and the square root is taken. While the equation for CE is not provided in [49], it is provided in [53] as binned error in the posterior (BEP). $BEP = \{\sum_{i=1}^{n}((\bar{s_{B_i}} - P_{T|B_i})^2 P_{B_i})\}^{1/2}$ where n is the number of bins, $\bar{s_{B_i}}$ is the average score in bin i, $P_{T|B_i}$ is the probability of targets in bin i, and P_{B_i} is the probability of bin i. Here P_{B_i} is assumed to be equal for all bins. According to the authors, the only problem with this measure is that discrete methods are not as effective as continuous methods.

Ross and Minardi, [53] expands upon Wise *et al.* [67] in terms of describing performance measures for both discrimination as well as confidence error. Ross proposes two methods for estimating confidence. One is based on information theory, the normalized cross-entropy (NCE) measure, and one is based on Bayesian thinking, Error in the Posterior (EP) (or BEP in the discrete case). A lower BEP value indicates more confidence than a higher BEP value. The authors discuss the difference between discrete EP and continuous EP. Discrete EP uses the binning discussed in [49]. The authors state that they do not know how to directly calculate the continuous EP without specifying a distribution. They recommend changing from the discrete EP to the continuous EP where the probability density function is a series of constant segments, and the cumulative distribution function is estimated by a piecewise linear function. Parker [49] provides an algorithm to do this using Beta distributions.

Weintraub *et al.* [66] define confidence measures to evaluate the confidence performance of the recognizer. They use the mean square error (MSE), cross-entropy (CREP), and classification error rate (CER). They also use a normalized MSE as well as a measure that combines the word error rate and word confidence, the net recognition performance (NERP).

2.7 Confidence and Fusion

Fusion of individual classifiers to increase confidence is a common theme in the literature (e.g., see [28], [62], [52], [9], [30], [31], [8], [39], [35], and [46]). Taking outputs from multiple classifiers and combining them into a single output has been long thought to improve the credibility, reliability, and hence confidence in any single indication.

There are a couple subsets of these articles pertaining to fusion. Ho *et al.* [28], Melnik *et al.* [46], and Tubbs and Alltop [62] only consider fusion when the output of the individual classifiers is a ranked list of classes. Bailey *et al.* [8] and Krzanowski *et al.* [39] examine fusion along with the idea of a rejection region. Over 10,000 classifiers, the percentage of classifiers that correctly classified the same exemplar is the level of confidence for that exemplar. A user defined threshold is applied to form the uncertainty envelope or rejection region.

2.8 Related Work

There are also a few other articles that do not necessarily pertain directly to indication confidence but still are on the periphery of the current research.

Daugman, [14] and [13], defines confidence in a completely different fashion. He uses high classification accuracy and a low false positive rate as a way of defining high-confidence classification. Daugman [13] discusses classification of people based upon features of their iris. Using a Hamming Distance measure, the authors get very accurate results. For certain hamming distances, the odds of a false match are extremely low. This is what the authors are calling high confidence recognition. There is not a confidence score for every iris being a match or not (i.e. every exemplar does not have an associated confidence score, just an associated hamming distance). Daugman [13] discusses visual recognition of people; this research is similar to that of his previous work [14].

Li and Sethi [44] introduce a new methodology called confidence based classifier design and implement the method using SVMs on a two-class problem. This is basically a way to set up a classifier with non-declarations. The authors describe this method as basically a post-processing of the individual classifier. The three-step process is to train the classifier, estimate the probabilities or error rates from the classifier, and calculate the two thresholds for the rejection region (or non-declaration region). By creating this rejection region, this methodology does not make a declaration unless there is enough "confidence" in the decision.

Bassham *et al.* [10] develop a methodology to evaluate automatic target recognition (ATR) systems through the use of a decision analysis paradigm. The ATR system evaluation framework is divided into two parts: the evaluator framework and the warfighter framework. The evaluator framework encompasses the engineering aspects of an ATR system, and the warfighter framework encompasses the operational aspects of an ATR system. This paper combines the two into a single framework used for ATR system evaluation. The evaluator framework may have some applicability to the idea of classifier confidence. In the same way that one would chose an alternative system with a higher value in this paradigm, one would also have more confidence in a system with a higher value. Thus, some of the same measurements and value functions may apply. The evaluator framework can be divided into seven major categories (and their associated weights): classification ability (0.11), cost (0.10), declaration ability (0.13), employment concept (0.15), detection performance (0.17), robustness (0.20), and self-assessment accuracy (0.14). This paradigm is meant to evaluate a set of alternative ATR systems before fielding, but the confidence problem is meant to evaluate how much confidence that we have in an indication of an already-fielded system. For this reason, cost should not be considered in confidence. Employment concept is either intelligence, surveillance, and reconnaissance (ISR) or ATR. Obviously, this has nothing to do with confidence in an ATR indication so it should not be considered in confidence. The other five categories seem to be applicable to confidence in an ATR system.

Ross *et al.* [54] provide performance measures for summarizing confusion matrices. The performance measures are given both in equation form as well as tabular form so they are displayed in a nice visual format. This paper also contains some of the overall vocabulary of ATR as well as an object-taxonomy.

Fenton and Wang [20] state that classical multi-criteria decision making (MCDM) methods cannot accommodate imprecise information. Probabilistic methods such as fuzzy set theory have been used to study this problem in the past. This paper introduces general fuzzy MCDM problem. The authors state that they believe risk and confidence are the two dimensions in which the DM's attitude is very subjective.

Schmeiser and Yeh [58] begin with a discussion of classical statistical confidence interval estimation for mean and variance as well as some strengths and weaknesses of these techniques. Then, some usual confidence interval criteria are discussed including expected half-width, half-width variance, and actual coverage probability. Essentially, this becomes a multiple criteria problem. The authors state that the ideal value for actual coverage probability is the nominal coverage probability, not one. Also, the ideal value for expected half-width is the width that results from accurately estimating the sampling error in the point estimate, not zero. Therefore, the shortest interval widths and the highest actual coverage probabilities are not necessarily the best. Also, a confidence interval needs to be good for all values of α, not just a particular value. The single criterion that they suggest for evaluation is the sum of squared error between the coverage function for the ideal confidence

interval and the coverage function for some specified confidence interval. One obvious problem with this approach is finding the ideal confidence interval.

Schruben [59] develops a coverage function to measure confidence interval robustness where η is the probability that a confidence region contains some unknown parameter (and equals $1-\alpha$) and η^* is defined to be the smallest confidence coefficient such that a known parameter is within the confidence region. It is the smallest because the regions get smaller as the confidence increases. The coverage function is given by $F_{\eta^*}(\eta) = Prob(\eta^* \leq \eta)$ and represents the probability that the interval actually contains the unknown parameter. Next, the author describes the two main problems in simulation output analysis: initialization bias and autocorrelated data. The empirical distribution function of η^* can be found by $G_{\eta^*}(\eta) = \frac{1}{n}\sum_{j=1}^{n} I_{[\eta^* \leq \eta]}$ where η_j^* are observed values of the confidence coefficient for a sample with n interval estimates and I is an indicator function.

Do *et al.* [16] use associative classification where there is a rule that is associated with some class. This rule maps features of an exemplar to a class label. This rule is determined through a data-mining process from a database of exemplars. The confidence value is conventionally used to describe how a rule performs on the training set. The predictive confidence value describes how the rule performs on the test set. The difference between the two is the confidence decrease. This is generalized for using multiple rules where the rules are weighted. Empirical results for confidence decrease are shown for a single rule and multiple weighed rules.

Ghosh *et al.* [24] use a likelihood L-statistic to measure confidence in audio-visual speech recognition. Specifically, they use the separation of likelihoods as the measure of confidence. They discuss feature-level fusion and decision-level fusion and state that decision-level fusion outperforms feature-level fusion in their application. They state that in a noise-free environment, audio usually should be used on its own; in fact, video can degrade classification performance. However, when noise is present, video can improve audio-only classification performance. The authors seek to weight audio and visual according to their confidence measure. In essence, they find a combined log-likelihood as a weighted combination of the audio and video log-likelihoods. These weights are determined as functions of the sample confidence value and the overall confidence value. The sample confidence value is the L-statistic of the log-likelihood for a single frame and is used to determine acoustic confusability. The overall confidence is the cumulative mean of the sample confidence values and is used as a measure of noise level.

Arenas-Garcia *et al.* [5] develop a methodology to increase the operational speed of classifying patterns. Since many neural structures are complex, the computational time to evaluate a pattern can be large. Thus, the authors develop a way to speed up this operational phase. They do this by examining the signs of partial sums for only certain important features. They examine the partial sum of the trained weights times their features (or features through a kernel function). If this partial sum is large enough (or small enough), then the hope is that the current sign will

correlate well with the class. Thus, the entire calculation does not need to be performed; only the partial calculation needs to be performed. Conservative thresholds are developed for two cases: if the variables are coded ± 1, and if the variables are coded 0,1. Since these are conservative, the authors reduce them by a factor of β. This is their confidence measure. As this number increases, the threshold becomes more conservative and there are fewer errors. That is, the faster classification acts more like the classification where all calculations were performed. Thus, this is a confidence measure between 0 and 1, but it is not a confidence measure for every exemplar.

Kanungo and Haralick [33] develop the methodology to choose the Bayes optimal operating point from a receiver operating characteristic (ROC) curve. The idea here is to find the point of the ROC curve where the slope of the tangeant is equal to the appropriate ratio of the costs of misclassification along with the appropriate prior probabilities. Of course, this means that the ROC curve must not be discrete; the derivative of the ROC curve must be found. Thus, the authors estimate the ROC curve with a spline representation.

Everson and Fieldsend [19] develop a methodology to find the Pareto front for multi-class classification problems where the costs of misclassification are unknown by posing it as a multi-objective optimization problem. When zero-one costs are used, the conditional risk is equal to the posterior probability. It is also stated that choosing the class with the maximum posterior probability minimizes the overall

error rate. If costs are known, it is easy to chose the class assignments that minimizes the Bayes risk. When costs are unknown, it is common to vary these costs and examine classification rates. The underlying classifiers used in this research were a k-nearest neighbor and a multilayer perceptron. Multi-class ROC surfaces are discussed as well as the volume under ROC curve measure. If $C(\Theta)_{jk}$ is the cost of misclassification of declaring class j as class k, then the multi-objective optimization problem is to minimize $C(\Theta)_{jk}$ for all $j \neq k$. Then, a classifier parameterized by Θ strictly dominates a classifier parameterized by Φ if and only if $C(\Theta)_{jk} \geq C(\Phi)_{jk}$ for all $j \neq k$ and $C(\Theta)_{jk} > C(\Phi)_{jk}$ for some $j \neq k$. Then, they present an algorithm that perturbs both the weights from the underlying classifier as well as the cost vector and calculates the misclassification rates. If the classifier isn't dominated, it is added to the Pareto front. This perturbation is repeated to generate the Pareto front.

Gandrabur *et al.* [60] discuss confidence estimation for nonlinear programming (NLP) application. First, they outline confidence estimation as "a generic machine learning rescoring approach for estimating the probability of correctness of the outputs of an arbitrary NLP application." In a rescoring framework, a new score is generated that is different than the baseline that can be used as a confidence measure for rejection or reranking. Outputs with a low enough confidence score are not reliable and thus are rejected. They state that probabilistic scores can be used, but the use of separate confidence measures provide more flexibility as it allows for recalibration of the baseline system. In defining confidence mathematically, the authors

obviously are treating confidence as a probability of correctness. This method also describes something popular in speech recognition. That is, a set of confidence features are generated and fed into another classifier that outputs an overall confidence measure. The authors discuss neural networks and their estimates of probability of correctness (or posterior probability) so they seem to be leaning toward the posterior probability being the ideal confidence measure. Then, the authors discuss evaluation of confidence measures. They mention two measures: ROC Curves and Normalized Cross-entropy. Finally, there is a large section of the paper dedicated to confidence estimation in the automatic speech recognition (ASR) application area.

3. Confidence Scores and Posterior Probabilities

This chapter provides an algorithm that minimizes a classification system measure of performance in the literature, shows the algorithm's equivalence to a popular algorithm used to train neural networks, and equates the output of the new algorithm to the posterior probability estimate.

Parker *et al.* [49] describe one output of a classification system as a confidence score. These scores are also employed by the Air Force Research Laboratory (AFRL) as shown in [53] and [67]. One confidence measure for an entire classification system is called either the confidence error (CE) [49] or binned error in the posterior (BEP) [53]. Ross and Minardi [53] state that under this paradigm, "the score ideally would be the Bayesian posterior probability."

In [49], a score is defined to be a number between 0 and 1. Based upon this score, a system declares whether or not a target is present in an image. Note that this structure is applicable to all classification problems, not just target declaration in an image. This score is described as the predicted probability that a target is in the image. In [53], the authors state that "we associate the pre-decision rule quantity with this detector reported confidence. We call the quantity or confidence a score." Thus, the score is taken as a measure of confidence in the indication of the classification system. According to [49], if 100 scores between 0.19 and 0.21 were sampled, then one would expect to see 20% of these to actually be targets. One metric used to measure the accuracy of the scores is the BEP or CE. Ross and

Minardi call this BEP and define it as

$$CE_T = BEP_T = \left(\sum_{j=1}^{N} ((\bar{s}_{B_j} - P_{T|B_j})^2 P_{B_j}) \right)^{1/2}. \tag{3.1}$$

Here, we denote BEP, as defined by Ross and Minardi, as BEP_T since it is calculated with respect to only the target class, T.

Using their notation, N is the number of bins, B_j denotes the j^{th} bin, $\bar{s}_{B_j}$ is the average score in bin B_j, $P_{T|B_j}$ is the estimated probability of targets in bin B_j, and P_{B_j} is the probability associated with bin B_j. Here P_{B_j} is assumed to be equal for all bins; that is, $P_{B_j} = \frac{1}{N}$ for all j. Thus,

$$CE_T = BEP_T = \frac{1}{\sqrt{N}} \left(\sum_{j=1}^{N} ((\bar{s}^T_{B_j} - P_{T|B_j})^2) \right)^{1/2}. \tag{3.2}$$

Ross only characterizes the BEP for target class. Now, define BEP for the non-target class, $BEP_{\bar{T}}$.

$$CE_{\bar{T}} = BEP_{\bar{T}} = \frac{1}{\sqrt{N}} \left(\sum_{j=1}^{N} ((\bar{s}^{\bar{T}}_{B_j} - P_{\bar{T}|B_j})^2) \right)^{1/2} \tag{3.3}$$

Then, to calculate the total BEP, we can sum across the target and non-target classes to get

$$BEP_{TOT} = BEP_T + BEP_{\bar{T}}. \tag{3.4}$$

Ross and Minardi [53] describe BEP as a measure of performance where a smaller value is better. This suggests training a classifier to minimize BEP.

Theorem 3.1. *The outputs of a classification system trained using back-propogation to minimize SSE are, in the limit of infinite data, posterior probabilities; hence, the outputs of a classification system trained to minimize BEP_{TOT} using back-propogation are, in the limit of infinite data, also posterior probabilities.*

In the remainder of the chapter, we present and prove two lemmas. Using these lemmas, we prove Theorem 3.1.

Lemma 3.1. *BEP can be minimized using a gradient descent algorithm.*

To calculate the BEP, all the scores are sorted in ascending order and binned into a predetermined number of bins. As shown in Equation 3.1, the BEP is computed as the square root of the average (across bins) of the squared differences between the average score in a bin and the actual percentage of targets in that bin.

Let $Z = [Z_k^c]$ where we define Z_k^c as the score for the k^{th} exemplar and the c^{th} class in a data set where $k = 1, ..., K$ and $c \in \{T, \bar{T}\}$. Let $d = [d_k^c]$ and $d_k = [d_k^T \; d_k^{\bar{T}}]$ where we define d_k^c as the desired output for the k^{th} exemplar and the c^{th} class in the data set where $d_k = [1\ 0]$ if the k^{th} exemplar is a target and $d_k = [0\ 1]$ if the k^{th} exemplar is a non-target. Let Z'^T be the vector where Z^T is sorted in ascending order, and let d'^T be the vector d^T sorted in the same order as Z'^T. In addition, let $Z'^{\bar{T}}$ be the vector where $Z^{\bar{T}}$ is sorted in the same order as Z^T and let $d'^{\bar{T}}$ be the vector $d^{\bar{T}}$ sorted in the same order as Z'^T. Binning must be accomplished in order

to calculate the BEP_{TOT}. Let M be the number of exemplars in a bin, and let N be the number of bins. Thus, $K = M \cdot N$. In what follows, assume that the bins are chosen such that each bin contains exactly M exemplars. Let $Z_{ij}^{\prime T}$ be the i^{th} sorted score for the target class in the j^{th} bin and let $Z_{ij}^{\prime \bar{T}}$ be the i^{th} sorted score for the non-target class in the j^{th} bin where $i = 1, ..., M$ and $j = 1, ..., N$. Let $d_{ij}^{\prime T}$ be the i^{th} sorted desired output for the target class in the j^{th} bin and Let $d_{ij}^{\prime \bar{T}}$ be the i^{th} desired output for the non-target class in the j^{th} bin. Now, we can estimate $\bar{s}_{B_j}$ as $\frac{1}{M}\sum_{i=1}^{M} Z_{ij}^{\prime T}$ and estimate $P_{T|B_j}$ as $\frac{1}{M}\sum_{i=1}^{M} d_{ij}^{\prime T}$. Then, the BEP_T is redefined as

$$BEP_T = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} \left(\frac{1}{M} \sum_{i=1}^{M} Z_{ij}^{\prime T} - \frac{1}{M} \sum_{i=1}^{M} d_{ij}^{\prime T} \right)^2, \tag{3.5}$$

and $BEP_{\bar{T}}$ is redefined as

$$BEP_{\bar{T}} = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} \left(\frac{1}{M} \sum_{i=1}^{M} Z_{ij}^{\prime \bar{T}} - \frac{1}{M} \sum_{i=1}^{M} d_{ij}^{\prime \bar{T}} \right)^2. \tag{3.6}$$

We manipulate BEP_T which yields

$$BEP_T = \frac{1}{\sqrt{N} \cdot M^2} \sum_{j=1}^{N} \left(\sum_{i=1}^{M} Z_{ij}^{\prime T} - \sum_{i=1}^{M} d_{ij}^{\prime T} \right)^2 \tag{3.7}$$

and

$$BEP_T = \frac{1}{\sqrt{N} \cdot M^2} \sum_{j=1}^{N} \left(\sum_{i=1}^{M} Z_{ij}^{\prime T} - d_{ij}^{\prime T} \right)^2. \tag{3.8}$$

The same manipulations can be applied to $BEP_{\bar{T}}$ which result in

$$BEP_{\bar{T}} = \frac{1}{\sqrt{N} \cdot M^2} \sum_{j=1}^{N} \left(\sum_{i=1}^{M} Z_{ij}^{\prime \bar{T}} - d_{ij}^{\prime \bar{T}} \right)^2 . \tag{3.9}$$

The expression for BEP_{TOT} is

$$BEP_{TOT} = \frac{1}{\sqrt{N} \cdot M^2} \sum_{j=1}^{N} \left(\sum_{i=1}^{M} Z_{ij}^{\prime T} - d_{ij}^{\prime T} \right)^2 + \frac{1}{\sqrt{N} \cdot M^2} \sum_{j=1}^{N} \left(\sum_{i=1}^{M} Z_{ij}^{\prime \bar{T}} - d_{ij}^{\prime \bar{T}} \right)^2 . \tag{3.10}$$

At this point, we assume the standard feed-forward neural network as shown in Figure 3.1. This figure is taken from Looney [45].

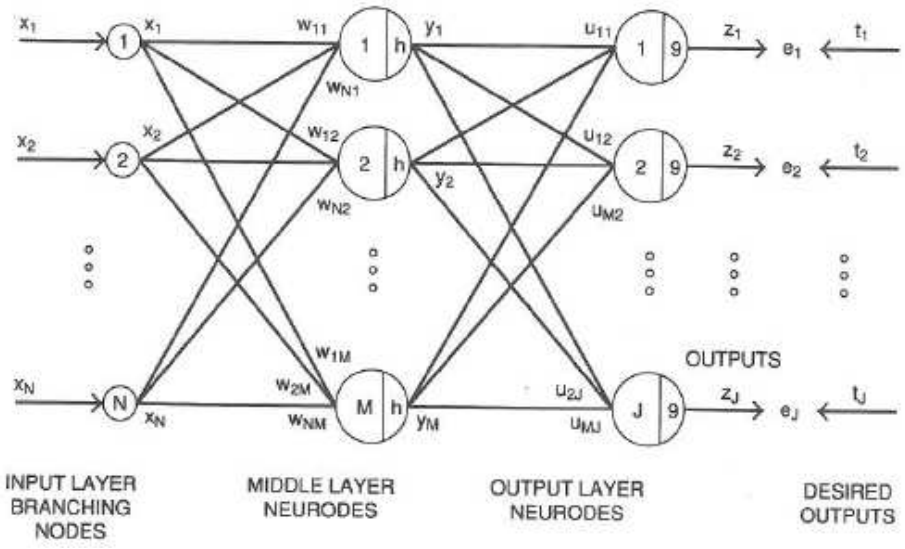

Figure 3.1 Standard Feed-Forward Neural Network [45].

This structure can be trained using a gradient descent algorithm designed to minimize the sum of squared error (SSE). The SSE is expressed as

$$SSE = \sum_{c \in \{T, \bar{T}\}} \sum_{k=1}^{K} [Z_k^c - d_k^c]^2. \tag{3.11}$$

Here we will leave the structure intact and change only the error function. Instead of minimizing SSE, we train the network to minimize BEP_{TOT} as shown in Equation 3.10. We must find the derivative of BEP_{TOT} with respect to both sets of learned weights, $\frac{\partial BEP_{TOT}}{\partial u_{mc}}$ and $\frac{\partial BEP_{TOT}}{\partial w_{mn}}$ where u_{mc} and w_{mn} are the learned weights. These are the standard back-propogation formulae which we summarize below. By substituting the new error function into equations found in Looney [45], we find

$$\frac{\partial BEP_{TOT}}{\partial u_{mc}} = \frac{\partial BEP_{TOT}}{\partial Z^c} \cdot \frac{\partial Z^c}{\partial s^c} \cdot \frac{\partial s^c}{\partial u_{mc}} \tag{3.12}$$

and

$$\frac{\partial BEP_{TOT}}{\partial w_{mn}} = \frac{\partial BEP_{TOT}}{\partial y_m} \cdot \frac{\partial y_m}{\partial r_m} \cdot \frac{\partial r_m}{\partial w_{mn}} = \frac{\partial BEP_{TOT}}{\partial Z^c} \cdot \frac{\partial Z^c}{\partial s^c} \cdot \frac{\partial s^c}{\partial y_m} \cdot \frac{\partial y_m}{\partial r_m} \cdot \frac{\partial r_m}{\partial w_{mn}}. \tag{3.13}$$

Relative to the derivative found in [45], only the error function changes and hence, the only partial derivative that changes is $\frac{\partial BEP_{TOT}}{\partial Z^c}$. Once the expression for $\frac{\partial BEP_{TOT}}{\partial Z^c}$ has been found, we can use the update equations for iteration r as given in Looney [45] where η is the step size.

$$w_{mn}^{r+1} = w_{mn}^{r} + \eta \frac{\partial BEP_{TOT}}{\partial w_{mn}} \tag{3.14}$$

$$u_{mc}^{r+1} = u_{mc}^{r} + \eta \frac{\partial BEP_{TOT}}{\partial u_{mc}} \tag{3.15}$$

Now, $\frac{\partial BEP_{TOT}}{\partial Z^{c}} = \frac{\partial BEP_{TOT}}{\partial Z_{i'j'}^{\prime c}}$ where $\frac{\partial BEP_{TOT}}{\partial Z_{i'j'}^{\prime c}}$ is the partial derivative of BEP_{TOT} with respect to $Z_{ij}^{\prime c}$ for a specific $i = i'$ and a specific $j = j'$. In order to accomplish batch learning, we seek

$$S = \sum_{j'=1}^{N} \sum_{i'=1}^{M} \frac{\partial BEP_{TOT}}{\partial Z_{i'j'}^{\prime c}}. \tag{3.16}$$

Now, we find $\frac{\partial BEP_{TOT}}{\partial Z_{i'j'}^{\prime T}}$ to be

$$\frac{\partial BEP_{TOT}}{\partial Z_{i'j'}^{\prime T}} = \frac{\partial}{\partial Z_{i'j'}^{\prime T}} \frac{1}{\sqrt{N} \cdot M^2} \sum_{j=1}^{N} \left(\sum_{i=1}^{M} (Z_{ij}^{\prime T} - d_{ij}^{\prime T}) \right)^2. \tag{3.17}$$

The following are simply the steps needed to determine the derivative with the final derivative given below.

$$\frac{\partial BEP_{TOT}}{\partial Z_{i'j'}^{\prime T}} = \frac{1}{\sqrt{N} \cdot M^2} \frac{\partial}{\partial Z_{i'j'}^{\prime T}} \left(\sum_{i=1}^{M} (Z_{ij'}^{\prime T} - d_{ij'}^{\prime T}) \right)^2 \tag{3.18}$$

$$= \frac{2}{\sqrt{N} \cdot M^2} \cdot [\sum_{i=1}^{M} (Z_{ij'}^{\prime T} - d_{ij'}^{\prime T})] \cdot \frac{\partial}{\partial Z_{i'j'}^{\prime T}} \sum_{i=1}^{M} (Z_{ij'}^{\prime T} - d_{ij'}^{\prime T}) \tag{3.19}$$

$$\frac{\partial BEP_{TOT}}{\partial Z'^{T}_{i'j'}} = \frac{2}{\sqrt{N} \cdot M^2} \sum_{i=1}^{M} (Z'^{T}_{ij'} - d'^{T}_{ij'}) \tag{3.20}$$

We can derive the final expression for S as

$$S = \frac{2}{\sqrt{N} \cdot M^2} \cdot \sum_{j'=1}^{N} \sum_{i'=1}^{M} \sum_{i=1}^{M} (Z'^{T}_{ij'} - d'^{T}_{ij'}) \tag{3.21}$$

$$= \frac{2M}{\sqrt{N} \cdot M^2} \cdot \sum_{j'=1}^{N} \sum_{i=1}^{M} (Z'^{T}_{ij'} - d'^{T}_{ij'}) \tag{3.22}$$

$$S = \frac{2}{\sqrt{N} \cdot M} \cdot \sum_{j=1}^{N} \sum_{i=1}^{M} (Z'^{T}_{ij} - d'^{T}_{ij}). \tag{3.23}$$

Now, recall how we defined Z'_{ij} as the sorted and binned values of Z_k. This implies that the following holds true,

$$\sum_{j=1}^{N} \sum_{i=1}^{M} (Z'^{T}_{ij} - d'^{T}_{ij}) = \sum_{k=1}^{K} (Z^{T}_{k} - d^{T}_{k}). \tag{3.24}$$

This means that the final expression for S is shown as

$$S = \frac{2}{\sqrt{N} \cdot M} \sum_{k=1}^{K} (Z^{T}_{k} - d^{T}_{k}). \tag{3.25}$$

Applying the same algebra to the non-target class, we get

$$S = \frac{2}{\sqrt{N} \cdot M} \sum_{k=1}^{K} (Z_k^{\bar{T}} - d_k^{\bar{T}}). \tag{3.26}$$

Proof. In order to prove Lemma 3.1, we use the results from Equations 3.25 and 3.26 and results from Looney to arrive at

$$\frac{\partial BEP_{TOT}}{\partial u_{mc}} = \frac{2}{\sqrt{N} \cdot M} \sum_{k=1}^{K} (Z_k^c - d_k^c) Z_k^c (1 - Z_k^c) y_m \tag{3.27}$$

and

$$\frac{\partial BEP_{TOT}}{\partial w_{mn}} = [\frac{2}{\sqrt{N} \cdot M} \sum_{c \in \{T, \bar{T}\}} \sum_{k=1}^{K} (Z_k^c - d_k^c)][y_m(1 - y_m)][x_n]. \tag{3.28}$$

Note that Looney uses sigmoids for activation functions, and recall that $Z = [Z_k^c]$ where we define Z_k^c (d_k^c) as the score (desired output) for the k^{th} exemplar and the c^{th} class in a data set where $k = 1, ..., K$ and $c \in \{T, \bar{T}\}$.

These can be substituted into the update Equations 3.14 and 3.15, and we have a gradient descent algorithm that minimizes BEP_{TOT}. □

Lemma 3.2. *The gradient descent algorithm that minimizes BEP_{TOT} is equivalent to the back-propogation gradient descent algorithm used to train neural networks based on sum of squared error.*

Proof. The gradient used in back-propogation as presented in Looney [45] and the gradient presented in the proof for Lemma 3.1 are equivalent to within a constant. Equations 3.29 and 3.31 show the gradient for the back-propogation weights as shown in Looney. Equations 3.30 and 3.32 show the gradient for the weights for the BEP algorithm shown here.

$$\frac{\partial SSE}{\partial u_{mc}} = -2\sum_{k=1}^{K}(Z_k^c - d_k^c)Z_k^c(1 - Z_k^c)y_m \tag{3.29}$$

$$\frac{\partial BEP_{TOT}}{\partial u_{mc}} = \frac{2}{\sqrt{N}\cdot M}\sum_{k=1}^{K}(Z_k^c - d_k^c)Z_k^c(1 - Z_k^c)y_m \tag{3.30}$$

$$\frac{\partial SSE}{\partial w_{mn}} = [-2\sum_{c\in\{T,\bar{T}\}}\sum_{k=1}^{K}(Z_k^c - d_k^c)][y_m(1 - y_m)][x_n] \tag{3.31}$$

$$\frac{\partial BEP_{TOT}}{\partial w_{mn}} = [\frac{2}{\sqrt{N}\cdot M}\sum_{c\in\{T,\bar{T}\}}\sum_{k=1}^{K}(Z_k^c - d_k^c)][y_m(1 - y_m)][x_n] \tag{3.32}$$

It is easy to see that the two sets of equations are equal to within a constant. This constant can be subsumed into the stepsize, ν, and the update equations for the two algorithms are equivalent. □

Now that we have proved Lemmas 3.1 and 3.2, we will prove Theorem 3.1.

Proof. In [55, 65, 17], it is shown that when training a classifier on the sum of squared error, the outputs theoretically estimate the probability of class membership conditioned on the data (i.e. the posterior probability). This includes any classification system (not only a neural network) that uses back-propogation to minimize sum of squared error. Thus, we have already shown in Lemma 3.2 that if we train a network to minimize the BEP_{TOT}, this is identical to training a network to minimize SSE. This, in turn, means the confidence score that minimizes BEP_{TOT} is nothing more than the posterior probability estimate. The theorem and corollaries presented here justify the statement made in [53] that the best confidence score, with respect to confidence error, is the posterior probability. □

While Ross and Minardi [53] state that BEP is a measure of system confidence, we contend this statement. While the value of BEP depends on the actual distribution of posterior probabilities, the value of BEP is independent of the *desired* distribution of the posterior probabilities. A classifier system may be able to perfectly estimate posterior probability estimates when compared to truth; however, clearly some distributions are more preferred than others. This suggests that confidence may better be represented as a function of the posterior probabilities instead of as the posterior probabilities themselves. That is the avenue taken throughout the remainder of this dissertation.

4. The Confidence Function for Non-Declarations

In the literature, we have seen that posterior probability estimates or constructs known as confidence scores are employed as confidence measures [49, 53]. There is a need to measure how well these confidence measures perform across a number of exemplars; one example of this, as put forth by Parker *et al.* [49], is the confidence error. Ross [53] makes the statement that under the discrete form of confidence error, a confidence score is ideally a posterior probability. In Chapter 3, we showed that training a multiple layered perceptron (MLP) neural network using back-propagation to minimize sum of square error also minimizes the confidence error. Additionally, we show that since, when training an MLP to minimize confidence error, the outputs of the MLP are posterior probability estimates, this implies that the best confidence score is simply a posterior probability.

Confidence error only measures how close the posterior probability estimates are to the true posterior probabilities. Confidence error does not measure how useful the posterior probability distributions are in separating the classes. In other words, there is no value in being able to perfectly measure posterior probabilities if there is no inherent separability in their true distributions. This suggests that confidence in an indication is not the posterior probability directly; instead, it should be modelled as a function of the posterior probabilities. In fact, this is exactly how current practices operate, albeit unconsciously. Without stating such, two current pattern

recognition techniques impose an implied confidence function on the posterior probability estimates.

In this chapter, a confidence framework is developed that encompasses two common pattern recognition scenarios: forced decisions and the allowance for non-declarations through rejection regions. In the forced decision problem, a classification system must choose between one of the output classes. When a rejection region is allowed, some decisions may be deferred if there is not enough confidence in the indication of the classification system. Both methodologies impose an implied confidence on the posterior probability estimates. In a forced decision problem, one has supreme confidence across the spectrum of posterior probability estimates; thus, all posterior probability estimates are used for classification purposes. The implied confidence function for this problem is a constant. The rejection region implies supreme confidence on all posterior probabilities outside the rejection region and a complete absence of confidence inside the rejection region; thus, only posterior probability estimates outside the rejection region are used for classification purposes. The implied confidence function for this problem is a step function. Believing that a middle ground exists between these extremes, we choose a quadratic function to represent the ascent from zero confidence to absolute confidence. As such, our methodology is a generalization of these current practices. In addition, our methodology produces a confidence score that can be used in conjunction with posterior probability estimates for classification decisions.

We present the notation for our confidence framework. Let a posterior probability estimate be denoted p and the set of all posterior probability estimates be denoted Ω. The posterior probability for class ω_d is given as

$$p_d = P(\omega_d|x) = \frac{P(\omega_d \cap s^{-1}\{x\})}{P(s^{-1}\{x\})} \tag{4.1}$$

where $s\{x\}$ represents the sensor/processor that produces the feature vector x and $s^{-1}\{x\}$ is the pre-image of x [17]. For the remaining discussion, we remove the d subscript from p.

Hence, $p \in \Omega \equiv [0, 1]$. Define the rejection region (an interval) as $\beta \subset \Omega$. In the forced decision scenario, $\beta = \emptyset$. Let $P(p)$ denote the distribution (probability density function) of the posterior probability estimates. We denote the confidence function as $C(p)$.

In a forced decision case, this confidence function is a constant function. Specifically,

$$C(p) = 1 \text{ for all } p \in \Omega \tag{4.2}$$

The graph of the confidence function for a forced decision is shown in Figure 4.1.

In the classical rejection region case, the confidence function is the complement of an indicator function as shown in Equation 4.3. An indicator function typically

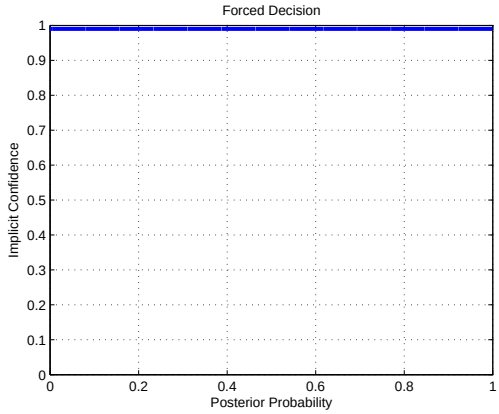

Figure 4.1 Implied Confidence Function For the Forced Decision Case.

assumes a value of 1 in a set and 0 out of a set; thus, the confidence function is the complement of an indicator function.

$$C(p) = \begin{cases} 0, & \text{if } p \in \beta \\ 1, & \text{if } p \notin \beta \end{cases} \tag{4.3}$$

The implied confidence function associated with a rejection region is shown in Figure 4.2.

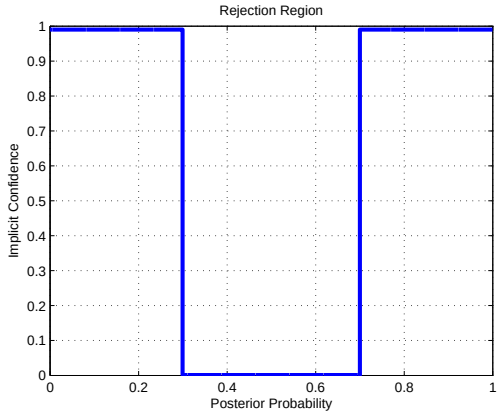

Figure 4.2 Implied Confidence Function Associated with a Rejection Region.

Figure 4.2 is somewhat deceiving as the rejection region, as it is typically displayed is actually a combination of two implied confidence functions, one for each class. Figure 4.3 shows the two confidence functions that comprise the rejection region.

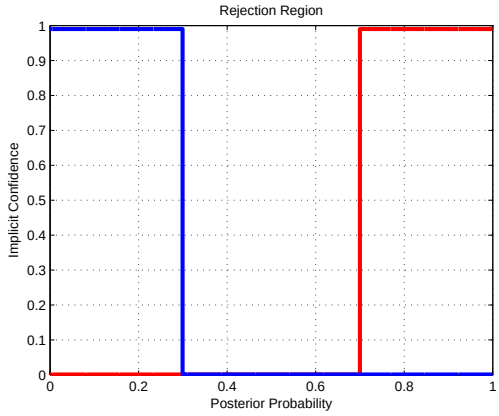

Figure 4.3 Rejection Region as Two Implied Confidence Functions.

The rejection region is implemented in the following fashion. If the posterior probability for an exemplar falls to the left of the rejection region, the exemplar is classified into one class. If the posterior probability for an exemplar falls to the right of the rejection region, the exemplar is classified into the other class. If the posterior probability for an exemplar falls in the rejection region, the exemplar is given non-declaration status.

The two confidence functions shown in Figure 4.3 are shown separately in Figure 4.4. This motivates the use for class specific confidence functions.

Thus far, two specific examples of $C(p)$ have been shown in Equations 4.3 and 4.2. From this, we can see that the rejection region is simply a generalization of

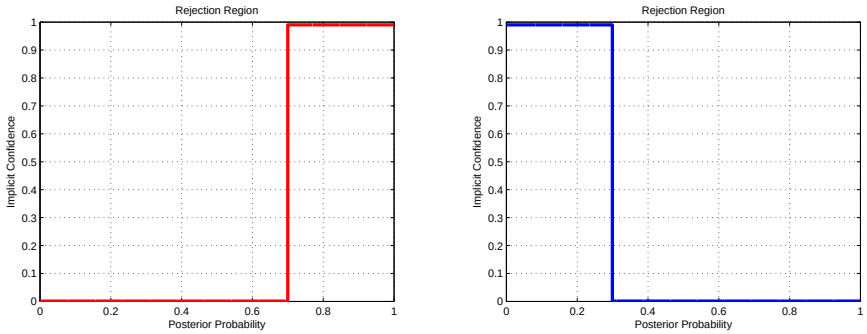

Figure 4.4 Two Separate Implied Confidence Functions for a Rejection Region.

the forced decision scenario. We propose to further generalize this methodology. Specifically, we posit a general form for the confidence function over the indications or outputs of the classifier. Specifically, $C(p)$, the confidence function, need not be a constant or a step function. In our paradigm, $C(p)$ takes on a general form and is used to impute confidence in a less severe fashion; thus, we are able to produce confidence values in the interval $[0, 1]$, as opposed to simply members of the set $\{0, 1\}$. The confidence function assigns a value in $[0, 1]$ to each possible posterior probability estimate. Two possible forms of the confidence function are modified quadratic functions and sigmoids; here, we examine the modified quadratic confidence function. Once a form is decided upon, parameters need to be estimated; the question that arises next is how to find the parameters associated with a specific confidence function.

Axiom 4.1. *The average confidence in the output of a classifier, by class, is approximately equal to the confidence in the classifier as it operates on that class (i.e., the class specific classifier confidence).*

In this research, we assume that Axiom 4.1 holds true. Thus, each class will have an associated confidence function. Class specific classifier confidence is estimated using multiattribute preference theory. The appropriate parameters for the class specific confidence function are "learned" through the assumed equality of class specific classifier confidence to the expected value of the confidence indications for each class, $E[C_d(p)]$ where $d = 1, ..., D$ and D represents the number of classes. Additionally, $C_d(p)$ produces a confidence score that can be used in conjunction with posterior probability estimates in classification decisions. The confidence function allows us to act on classifier indications in a manner that is not arbitrary and consistent with our inherent confidence in such indications.

4.1 Confidence Function Overview

There are two types of confidence developed in this research. The first is confidence in a classification system (or, for brevity, a classifier) and is denoted classifier confidence. This confidence is developed for each class. The second is the confidence in the output (indication) of a classification system or classifier. The amount of class specific confidence in a given classifier is estimated using multiattribute preference theory and forms the foundation for a quadratic confidence function that is applied to posterior probability estimates. Classifier confidence is based upon individual measurable value functions for class specific classification accuracy, average entropy, and class specific sample size (of the training set). The form of the overall measurable

value function is multilinear based upon the assumption of weak difference independence [18]; the use of this form is developed later in the chapter. This multilinear function is used to determine class specific classifier confidence; hence, we assume that value and confidence are analogous. We posit a general form for a confidence function over the indications or outputs of the classifier. The confidence function assigns a value in $[0, 1]$ to each possible estimate. The appropriate parameters for the confidence function are "learned" through the assumption that the average confidence one has in the output of a classifier, by class, should be approximately equal to the confidence one has in the classifier as it operates on that class (i.e., the class specific classifier confidence).

In this research, a modified quadratic function is used as the confidence function. In our paradigm, we have no confidence in exemplars with low posterior probability estimates. We have growing confidence in exemplars with moderate posterior probability estimates. After a point, we have supreme confidence in exemplars with high posterior probability estimates. In this paradigm, confidence in an indication is related to the posterior probability estimate but is not equal to it. This confidence methodology is a direct link between traditional decision analysis techniques and traditional pattern recognition techniques.

4.2 Multiattribute Preference Theory Application

Multiattribute preference theory is a branch of decision analysis that allows a decision maker to determine preferences for alternatives when each of the alternatives has more than one attribute. Within multiattribute preference theory, there are two classes of problems. One class of problem is determining preferences with uncertainty, and the other class of problem is determining preferences with certainty. In either case, let the set of alternatives be denoted by $A = \{a_1,\ a_2,\ ...,\ a_m\}$. Each of the m alternatives has n attributes and are denoted by $X = (X_1, X_2, ..., X_n)$ where X denotes the entire set of attributes. Let x_i represent a specific level of X_i. Each of these n attributes are evaluated over a region $x_i^L \leq x_i \leq x_i^H$ where x_i^L is the lowest level of interest for X_i and x_i^H is the highest level of interest for X_i. Also, the least preferred level of X_i is denoted by x_i^0, and the most preferred level of X_i is denoted by x_i^*. Let $\bar{x}_i$ denote all the levels of all attributes except attribute i. Let $(s_i; \bar{x}_i)$ denote that attribute i is at level s_i while all other attributes are at level x_i. This notation represents levels for all attributes but calls specific attention to the particular level of X_i. Finally, let $(s_i; \bar{x}_i)(x_i^o; \bar{x}_i)$ denote that attribute i has changed from level x_i^o to level s_i while all other attributes were held constant at levels x_i.

When the actual consequence of choosing alternative a_i is uncertain, the scenario is known as determining preferences under uncertainty. In this case, choosing alternative a_i has a number of potential consequences; each of these consequences has an associated probability. Here, an expected utility can be found for each alter-

native. In this class of problem, utility theory is used which takes into account the decision maker's risk attitudes.

The other class of problem is determining preferences under certainty. This implies that the actual consequence of choosing alternative a_i is certain. In this case, choosing alternative a_i has only one consequence; this consequence has a probability of one. In this class of problem, value functions are used. Value functions take into account the decision maker's values but not his risk preferences [38]. Since we are interested in evaluating classifiers, the equivalent decision problem would be to choose the best classifier. The alternatives are the classifiers themselves so there is no uncertainty in the consequence; thus, we are interested in using value functions.

4.2.1 Measurable and Non-Measurable Value Functions

There are two types of value functions: non-measurable value functions and measurable value functions. Non-measurable value functions allow a decision maker to rank order alternatives but do not allow any analysis of the difference between values for two alternatives (i.e., ordinal scale measurement). Measurable value functions do allow for analysis of the difference between two values (i.e., interval scale measurement). Hence, using measurable value functions, an alternative with a value of 0.7 is more preferred to an alternative with a value of 0.4 by a difference of 0.3. Here, we require measurable value functions.

4.2.2 Functional Form of the Overall Measurable Value Function

Since we know measurable value functions will be used, we must determine the functional form of the overall measurable value function. The functional form of the overall measurable value function is based on whether or not certain conditions hold. The following notation and definitions explain what each of these conditions means.

Using the notation presented by Dyer and Sarin [18], let X_I and $\bar{X}_I$ be a partition of the attributes $X = \{X_1, X_2, ..., X_n\}$. Let $s \succ t$ denote that alternative s is preferred to alternative t. Let $wx \succeq^* yz$ mean that the strength of preference for w over x is greater than or equal to the strength of preference for y over z [18].

Definition 4.1. *"Weak Difference Independence: X_I is weak difference independent of the $\bar{X}_I$ if given any w_I, x_I, y_I, $z_I \in X_I$ and some $\bar{w}_I \in \bar{X}_I$ such that $(w_I; \bar{w}_I)(x_I; \bar{w}_I) \succeq^* (y_I; \bar{w}_I)(z_I; \bar{w}_I)$, $(w_I; \bar{x}_I)(x_I; \bar{x}_I) \succeq^* (y_I; \bar{x}_I)(z_I; \bar{x}_I)$ for any $\bar{x}_I \in \bar{X}_I$ [18]."*

$(w_I; \bar{w}_I)(x_I; \bar{w}_I) \succeq^* (y_I; \bar{w}_I)(z_I; \bar{w}_I)$ means that changing attribute I from level x_I to level w_I while holding all other attributes constant at levels $\bar{w}_I$ is more preferred (or equally preferred) to changing attribute I from level z_I to level y_I while holding all other attributes constant at levels $\bar{w}_I$.

"If X_I is weak difference independent of $\bar{X}_I$, then the ordering of preference differences depends only on the values of the attributes X_I [18]."

Dyer and Sarin [18] reference a theorem in the Keeney and Raiffa text [34], change the verbiage slightly, and define the conditions for a multilinear measurable

value function. Specifically, with a set of attributes, $X = (X_1, X_2, ..., X_n)$ where $n \geq 2$ if X_I is weak difference independent of $\bar{X}_I$, then a multilinear measurable value function is appropriate. A multilinear measurable value function with three attributes is given by $v^m(x) = \lambda_1 \cdot v_1^m(x_1) + \lambda_2 \cdot v_2^m(x_2) + \lambda_3 \cdot v_3^m(x_3) + \lambda_{1,2} \cdot v_1^m(x_1) \cdot v_2^m(x_2) + \lambda_{1,3} \cdot v_1^m(x_1) \cdot v_3^m(x_3) + \lambda_{2,3} \cdot v_2^m(x_2) \cdot v_3^m(x_3) + \lambda_{1,2,3} \cdot v_1^m(x_1) \cdot v_2^m(x_2) \cdot v_3^m(x_3)$.

Kirkwood [38] outlines a four-step procedure for developing a value model using measurable value functions which we can modify based upon results of Dyer and Sarin [18] and Keeney and Raiffa [34]. The first step is to define the attributes, $X_i, i = 1, ..., n$, as well as x_i^0 and x_i^*. Kirkwood's [38] second step is to either test a set of assumptions or assume them to be true. The third step, according to Kirkwood [38], is to define $v_i^m(x_i)$, the measurable value function for the i^{th} attribute; this can be done by directly measuring the relative value increments between different attribute levels. The fourth step is to determine the weights used in the overall value function. Kirkwood assumes that the functional form of the overall value function is additive. Kirkwood [38] also states that even though Dyer and Sarin [18] have derived other forms of measurable value functions, none have actually been put into practice (as of 1997). It should be noted that the multilinear form requires weights similar to those in step four of Kirkwood, but Kenney and Raiffa provide specific instructions on how to determine these weights [34].

While Kirkwood [38] defines a four step process, our process necessitates a fifth step. In multiattribute preference theory, the actual quantity associated with the

value of an alternative has no meaning in isolation. We want the output of our value model to have meaning; specifically, we want the meaning to be the confidence that a decision maker has in that classifier. Therefore, we must add a fifth step to achieve this objective. The fifth step of our process is to validate the output of the model with the decision maker to ensure that the model does, in fact, tell us the confidence the decision maker has in that classifier.

4.2.3 Attributes for Classifier Confidence

The following describes the logic behind the attribute choices for classifier confidence. Confidence can be described as a lack of uncertainty. In information theory, the amount of uncertainty in a sequence of values is called the entropy [17]. In our case, we are interested in the amount of uncertainty in the posterior probability distributions since posterior probability estimates are often used to make declarations.

Consider a two-class problem. We will estimate the posterior probability distribution for class 1 with a histogram of posterior probability estimates from exemplars known to be in class 1. We will estimate the posterior probability distribution for class 0 with a histogram of posterior probability estimates from exemplars known to be in class 0. Let C denote the number of bins used to form the histogram. Let the frequency of posterior probability estimates calculated from class d in bin c be represented by F_c^d. Then, the entropy for class d is given by $E^d = -\sum_{c=1}^{C} F_c^d log_2\left(F_c^d\right)$. Here $0\, log_2\, 0 \equiv 0$ by definition. Figure 4.5 shows the 8-bin histogram that has the

highest entropy; the entropy in this case has a numeric value of 3. Figure 4.6 shows the 8-bin histogram that has the lowest entropy; the entropy in these cases has a numeric value of 0.

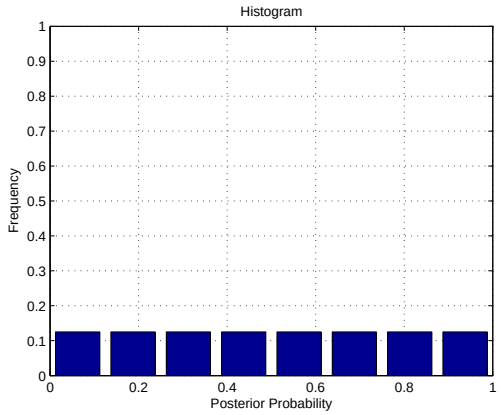

Figure 4.5 Worst Case for Entropy.

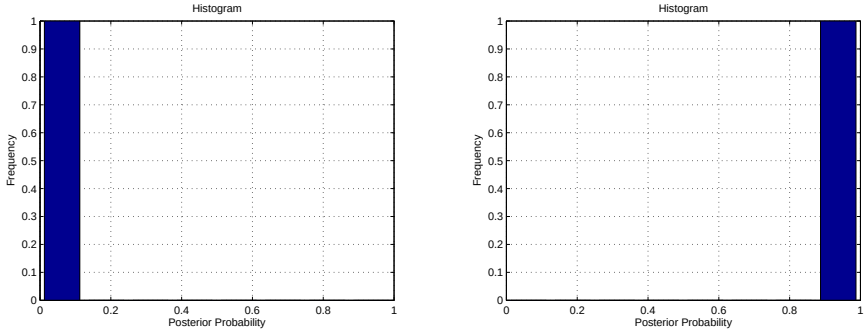

Figure 4.6 Best Cases for Entropy.

Entropy depends on the number of bins used to form the histogram. For example, the entropy of the worst case using 8 bins has a numeric value of 3, but the entropy of the worst case using 7 bins has a numeric value of 2.8. The entropy of the best case has a numeric value of 0 for any number of bins. To make entropy comparable across the number of bins, we will form a normalized entropy by dividing

by the entropy of the worst case. Let $\tilde{E}_C$ denote entropy of the worst case using C bins. Hence, the normalized entropy for class d is $E_n^d = \frac{E^d}{\tilde{E}_C}$. We also seek a single entropy value across the classes. We define the aggregated normalized entropy as $E_n = (\%\, of\ class\ 0) \cdot E_n^0 + (\%\, of\ class\ 1) \cdot E_n^1$.

We have more confidence in a classification system that outputs a posterior probability estimate histogram similar to the best case than we have in a classification system that outputs a posterior probability estimate histogram similar to the worst case. Hence, entropy is a logical choice as an attribute in determining confidence in a classification system. There is one additional consideration. Both histograms shown in Figure 4.6 result in the same entropy value. However, for a given class, one will be 100 % accurate, and the other will be 0 % accurate. This suggests we combine entropy with classification accuracy to measure confidence in a classification system.

Classification accuracy (CA) is calculated across exemplars in both classes. There are four possible outcomes from a two-class, forced decision classification system.

Define a true positive (TP) as a 'target' indication from a classification system given that the exemplar is actually a target. Define a true negative (TN) as a 'non-target' indication from a classification system given that the exemplar is actually a non-target. Define a false positive (FP) as a 'target' indication from a classification system given that the exemplar is actually a non-target. Define a false negative (FN) as a 'non-target' indication from a classification system given that the exemplar is

actually a target. The equations for true positive rate, true negative rate, false positive rate, and false negative rate are defined as

$$TP_{rate} = \frac{number\ of\ TPs}{number\ of\ targets}, \tag{4.4}$$

$$TN_{rate} = \frac{number\ of\ TNs}{number\ of\ nontargets}, \tag{4.5}$$

$$FP_{rate} = \frac{number\ of\ FPs}{number\ of\ nontargets}, \tag{4.6}$$

and

$$FN_{rate} = \frac{number\ of\ FNs}{number\ of\ targets}. \tag{4.7}$$

Aggregate classification accuracy is defined as

$$CA_{agg} = (\%\ of\ targets \cdot TP_{rate}) + (\%\ of\ nontargets \cdot TN_{rate}). \tag{4.8}$$

In addition, we would have more confidence in a classification system as the number of samples used to train the system increases. Thus, the total sample size across classes is the third attribute used to determine classifier confidence.

4.2.4 Attributes for Class Specific Classifier Confidence

While the classifier confidence quantifies the confidence in the classifier as a whole, we are also interested in class specific classifier confidence. This represents how confident we are in classifier outputs for a specific class. This is especially relevant for problems where the classifier has high accuracy for one class but low accuracy for another. In this case, we would not want to let classifier performance on one class bias our confidence on the other class. For class specific classifier confidence, we calculate the class specific sample sizes (SS), entropies (Ent), accuracies. We will use the class specific sample sizes (i.e., the actual number of each class used to train the classifier). We will also use the class specific entropy values. Additionally, we will use the true positive rate as the accuracy measure for one class and true negative rate as the accuracy measure for the other class. Now, we can calculate the aggregate classifier confidence value as well as class specific classifier confidence values. Class specific confidence values are used to fit the class specific confidence functions, and the aggregate classifier confidence value is tracked as a performance measure in Chapter 7. Combination of these three attributes, including their interactions, into a classifier confidence value, C^{ENG}, is portrayed in Figure 4.7.

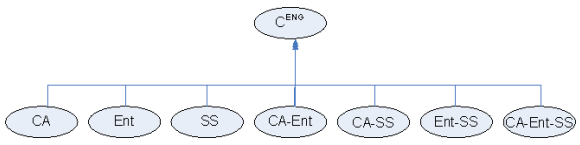

Figure 4.7 Value Hierarchy with Interactions.

4.2.5 Implementation of the Five-Step Process

For this research, the engineer is the decision maker. Current implementation is with respect to the engineer and produces an engineering confidence in a classifier. Now, we apply the four-step process detailed above. The first step is to define the attributes of the alternatives. In this case, there are three attributes: class specific sample size, class specific classification accuracy, and average entropy. Let $d = 1, ..., D$ be the possible class labels where D is the total number of class labels. Entropy will be constant across all classes because it is calculated across classes. Class specific classification accuracy and sample size will both vary across the class labels. Hence, they are subscripted with the class label, d. Let sample size be denoted X_{SS_d} where $x^0_{SS_d} = 2$ and $x^*_{SS_d} = 500$. Let classification accuracy be denoted X_{CA_d} where $x^0_{CA_d} = 0.5$ and $x^*_{CA_d} = 1$. Let average entropy be denoted $X_{\bar{E}}$ where $x^0_{\bar{E}} = 1$ and $x^*_{\bar{E}} = 0$. Because we are assuming a two-class problem, the scenario with the minimum confidence is a coin-flip scenario. Since classification accuracy could, practically speaking, be less than 0.5, any classification accuracy less than 0.5 will be assessed a value of 0. Also, since sample size, practically speaking, could be greater than 500, any sample size greater than 500 will be assigned a value of 1.

Following Kirkwood's [38] second step, we can let $X_I = \{X_{CA}, X_{\bar{E}}\}$ and $\bar{X}_I = \{X_{SS}\}$. Now, we can test the attributes for weak difference independence. The difference in value of having an increase in sample size from 50 to 100 does not depend on the levels of classification accuracy and average entropy. We examine 2 cases. Let

the levels of CA and average entropy be set at 0.6 and 0.4, respectively in case 1, and let the levels of CA and average entropy be set at 0.8 and 0.2, respectively in case 2. The value difference resulting from an increase in sample size (e.g., sample size increases from 50 to 100) creates the same increase in case 1 and case 2. Therefore, X_I and $\bar{X}_I$ are weak difference independent and the functional form of the overall measurable value function is multilinear [18, 34].

Step 3 is to define the individual measurable value functions $v_{ca}(x_{ca})$, $v_{\bar{e}}(x_{\bar{e}})$, and $v_{ss}(x_{ss})$. This is done by directly measuring the relative value increments between different attribute levels. Using this methodology, individual measurable value functions were derived. Figure 4.8 shows individual measurable value functions for classification accuracy, average entropy, and sample size.

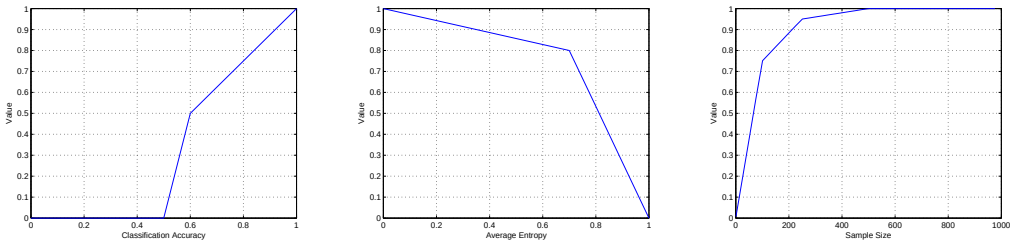

Figure 4.8 Measurable Value Functions.

Once the individual value functions have been found, the next step is to find the weights for those individual attributes in the overall value function. The same theorem from Keeney and Raiffa [34] that gives the conditions for a multilinear measurable value function also states how to determine the weights for the multilinear

measurable value function. Specifically, $k_i = v(x_i^*, \bar{x}_i^0)$, $k_{ij} = v(x_i^*, x_j^*, \bar{x}_{ij}^0) - k_i - k_j$, and $k_{ijk} = 1 - \sum_i k_i - \sum_{i,j>i} k_{ij}$.

Figure 4.9 shows two contour plots of engineering confidence; each plot displays classification accuracy vs. average entropy for a fixed level of sample size. A low level of sample size is used for the plot on the left, and a high level of sample size is used for the plot on the right.

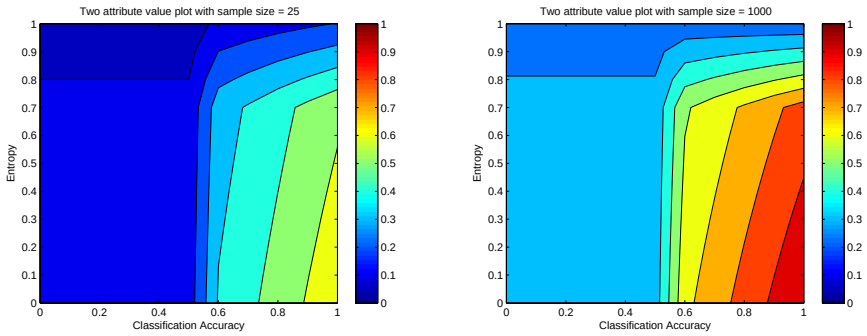

Figure 4.9 Confidence Contours for Classification Accuracy and Average Entropy.

Figure 4.10 shows two contour plots of engineering confidence; here, each plot shows classification accuracy vs. sample size for a fixed level of average entropy. A low level of average entropy (in terms of value) is used for the plot on the left, and a high level of average entropy (in terms of value) is used for the plot on the right.

Figure 4.11 shows the final two contour plots of engineering confidence; each plot depicts average entropy vs. sample size for a fixed level of classification accuracy. A low level of classification accuracy is used for the plot on the left, and a high level of classification accuracy is used for the plot on the right.

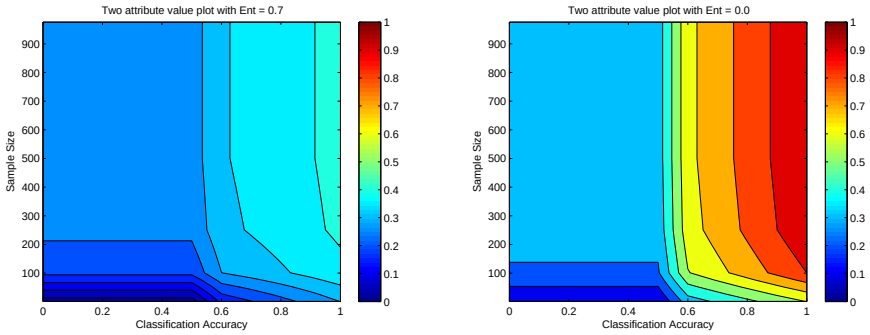

Figure 4.10 Confidence Contours for Classification Accuracy and Sample Size.

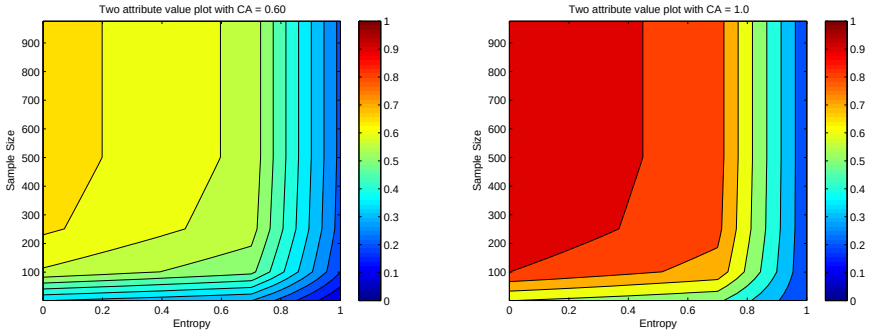

Figure 4.11 Confidence Contours for Average Entropy and Sample Size.

Figure 4.12 shows the value hierarchy with the individual value function. The interactions are removed for simplicity.

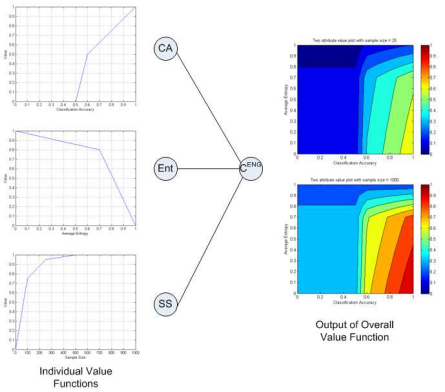

Figure 4.12 Engineering Confidence Value Hierarchy.

The fifth and final step involves validation of the values in Figures 4.9, 4.10, and 4.11 with the engineer to ensure that the output of the model can be interpreted as the confidence that the engineer has in a classifier. Using multiattribute preference theory, classification accuracy, average entropy, and sample size are used to derive a single engineering confidence value by class, C_d^{ENG}, that can be used to train a quadratic confidence function.

4.3 Quadratic Confidence Function

One way to view confidence for a given indication is through the use of a quadratic confidence function. One should have less confidence in a posterior probability estimate in a region where there is a great deal of overlap between two populations; on the other hand, one should have more confidence in posterior probability estimates in regions with less overlap between the two populations. To this end, a quadratic confidence function is fit for each of the classes. Hence, we have no confidence in exemplars with low posterior probability estimates. We have increased confidence in exemplars with moderate posterior probability estimates. After a point, we have supreme confidence in exemplars with high posterior probability estimate. The region of growing confidence is the fundamental different between the quadratic confidence function and the implied confidence function imposed by a rejection region. For illustrative purposes, Figure 4.13 shows one potential quadratic confidence function along with one implied confidence function imposed by a rejection region.

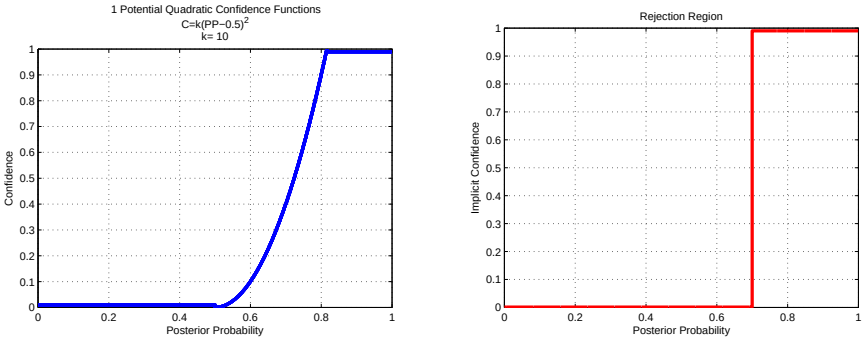

Figure 4.13 Comparison Between Rejection Region Implied Confidence Function and Quadratic Confidence Function.

We will modify the standard form of a quadratic function to use as our confidence function. Let's define the standard, unmodified quadratic function, $C_U(p;k,p_0) = k \cdot (p-p_0)^2$ where p_0 is the vertex of the function and k is the multiplicative factor. We will make two modifications to this quadratic function and denote the modified quadratic function $C(p;k,p_0)$. If $p < p_0$, $C(p;k,p_0) = 0$. Thus, our confidence increases as a posterior probability increases from p_0, but we have zero confidence in posterior probabilities less than p_0. Second, if $C_U(p;k,p_0) > 1$, $C(p;k,p_0) = 1$. These modifications bound our confidence function between 0 and 1 and form a piecewise confidence function. The definition for the quadratic confidence function is

$$C(p;k,p_0) = \begin{cases} 0, & \text{if } p < p_0 \\ 1, & \text{if } p \geq \sqrt{\frac{1}{k}} + p_0 \cdot \\ C_U(p;k,p_0), & \text{otherwise} \end{cases} \tag{4.9}$$

Figure 4.14 shows potential modified quadratic confidence functions for eight values of k where $p_0 = 0.5$. As k increases, the modified quadratic confidence function becomes steeper. Also, one can now visualize the regions of no confidence, growing confidence, and supreme confidence. In the next section, we derive methodologies to determine p_0 and k.

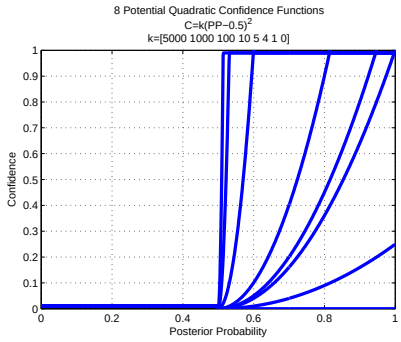

Figure 4.14 Potential Modified Quadratic Confidence Functions.

4.4 Parameterizing the Confidence Function

In our confidence paradigm, we posit that the average confidence one has in the output of a classifier, by class, should be approximately equal to the confidence one has in the classifier as it operates on that class (i.e., the class specific classifier confidence). The confidence function for class d is denoted $C_d(p; k, p_0)$. Since the same process is followed for all classes, for ease of notation, we suppress the subscript d and write C^{ENG} for C_d^{ENG} and $C(p; k, p_0)$ instead of $C_d(p; k, p_0)$. Hence, we wish to solve the following optimization problem

$$\underset{k \geq 0,\ p_0 \in [0,1]}{min} (C^{ENG} - E[C(\bullet; k, p_0)])^2. \tag{4.10}$$

Since k and p_0 are both unknown, we will fix p_0 and solve for k. This should yield a functional relationship where $k^* = F(p_0)$. We will now define methodologies for fixing p_0 and solving for k.

4.4.1 Finding the Loss Vertex

One logical methodology to find p_0 is to place it close to the Bayes optimal threshold; this is the threshold that minimizes the classification error. However, there is an advantage in our paradigm to choosing a smaller value of the vertex rather than a larger value of the vertex. As stated earlier, we are developing a confidence measure less severe than the implied confidence set forth by a rejection region. As the vertex value decreases, a smaller quantity for k can be used which will correspond to a less severe quadratic confidence function (i.e., one that rises from 0 to 1 more slowly). In turn, this will provide a less severe approach than the rejection region approach. Thus, we will move the vertex (i.e., by decreasing it) away from the Bayes optimal point to allow for a gentle increase in confidence initially and then a more rapid increase past the Bayes optimal point. For a given histogram, a smaller vertex value allows for a larger range of expected confidence values to be achieved. To illustrate this point, consider Figure 4.15 where two plots are shown. The first plot shows a notional histogram of posterior probabilities with possible quadratic

functions overlaid. The vertex for all possible quadratic functions shown is 0.5. The second plot shows the expected confidence as a function of k.

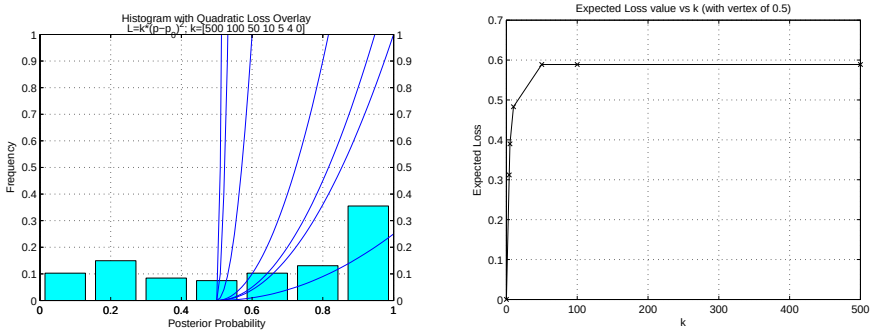

Figure 4.15 Histogram with Confidence Functions and Expected Confidence.

It is easy to see that no matter how large k is, the largest expected confidence is 0.59. If this represented a scenario where the engineer has more confidence than 0.59, we would not be able to solve the optimization problem in Equation 4.10 without decreasing the value of p_0.

Now that we have shown how our paradigm benefits from smaller p_0 values, we will now discuss our procedure for finding p_0. We start by finding the Bayes optimal threshold from the classifier when applied to the training set (i.e., the threshold that corresponds to the maximum classification accuracy in the training set). Let the Bayes optimal threshold be denoted θ_{Bayes}. Let the classification accuracy corresponding to this threshold be denoted CA_{Bayes}. Let $\epsilon_{CA} > 0$ be the allowable tolerance on classification accuracy, and let the classification accuracy corresponding to any set threshold, θ, be denoted CA_θ. Now, the optimal θ, denoted θ^*, is found by solving the constrained optimization problem

$$min\{\theta \in [0,1] : CA_\theta \geq CA_{Bayes} - \epsilon_{CA}\}. \tag{4.11}$$

Then, $p_0 = \theta^*$. Thus, the optimal threshold is the minimum threshold that meets the minimum acceptable classification accuracy constraint. Now that a methodology to determine p_0 has been set forth, a methodology to determine optimal k must be developed.

4.4.2 Finding the Optimal Multiplier

We can find the expected value of the function, g, by integrating it with respect to its probability density, f, by calculating $E[g] = \int_{-\infty}^{\infty} g(p)f(p)dp$ [64]. In the discrete case, we calculate the expected value as $E[g] = \sum_{i=1}^{n} g(p_i)f(p_i)$ where p is divided into n bins.

By integrating the confidence function, $C(p; k, p_0)$, relative to a posterior probability density, $f(p)$, we can find the expected value of the confidence function, $E[C(\bullet; k, p_0)] = \int_0^1 C(p; k, p_0)f(p)dp$. In the discrete case, we can calculate the expected value as $E[C(\bullet; k, p_0)] = \sum_{i=1}^{n} C(p_i; k, p_0)f(p_i)$. For this research, we only use the discrete case. Our paradigm dictates that we choose k^*, the optimal value of k, such that $E[C(\bullet; k^*, p_0)]$ equals the confidence the engineer has in the classifier, C^{ENG}. This is shown as

$$k^* = \underset{k}{argmin} \; (C^{ENG} - E[C(\bullet; k, p_0)]_{p_0=\theta^*})^2. \tag{4.12}$$

Here, C^{ENG} was determined using multiattribute preference theory.

In practice, the theoretical density of the posterior probabilities is most likely unknown. Thus, we must estimate this density empirically using the posterior probability estimates from the underlying classifier. This is done using a histogram of the posterior probability estimates. The first decision that must be made is determining the number of intervals, c, in the histogram. Sturges's rule is the most well-known heuristic and states that $c = \lfloor 1 + log_2\, n \rfloor$ where n is the number of exemplars in the data set [42]. Let p_i, $i = 1, ..., c$ be the center of the i^{th} interval. Then, let $f(p_i)$ be the percentage of data in the i^{th} interval. Then, empirically, $E[C(\bullet; k, p_0)] = \sum_{i=1}^{c} C(p_i; k, p_0) f(p_i)$. Now that we have an expression for $E[C(\bullet; k, p_0)]$, and a methodology to find C^{ENG} and p_0, we can find k^*; once k^* is found, the quadratic confidence function will be fully defined.

Since $C(p; k, p_0)$ is a piecewise function, we can break $E[C(\bullet; k, p_0)]$ into three components. Define r to be the smallest index of p_i such that $p_i > p_0$. Define s as the largest index of p_i such that $k(p_i - p_0)^2 < 1$. Equation 4.13 shows the expression for $E[C(\bullet; k, p_0)]$ broken down into three components.

$$E[C(\bullet; k, p_0)] = \sum_{i=1}^{r-1} 0 \cdot f(p_i) + \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=s+1}^{c} 1 \cdot f(p_i) \qquad (4.13)$$

Thus, we have broken the expected value into three parts: $1 \leq i < r$, $r \leq i \leq s$, and $s < i \leq c$. As k changes, s will change in reaction. Since changing k changes the limits of the summation when determining $E[C(\bullet; k, p_0)]$, we must solve for k^* numerically. Specifically, we will use the bisection method to solve for the optimal value of k. In order to use the bisection method, we must find one value of k that is less than k^* and one value of k that is greater than k^*.

4.4.2.1 Bounds on k^*

One way to find a lower bound on k^* is to solve a relaxation of the problem. In particular, we will solve for the optimal value of k on the unmodified confidence function, $C_U(p; k, p_0)$. Since $C_U(p; k, p_0)$ is not piecewise, the limits of the summation do not depend on k so we can directly solve for the optimal k. In order to optimize k, we must find the value of k that minimizes the expression $D = \|C^{ENG} - E[C_U(p; k, p_0)]_{p_0=\theta^*}\|_2$. We know this is a lower bound on k^* because $C_U(p_i; k, p_0) \geq C(p_i; k, p_0)$ for all i. Therefore, for the same values of k and p_0, $E[C_U(p_i; k, p_0)] \geq E[C(p_i; k, p_0)]$. In order to achieve the same expected value, the unmodified version must use a smaller k than the modified version. Hence, this will provide an initial lower bound on k^*, denoted k_0^L.

In order to solve the relaxation, classical unconstrained optimization theory tells us that we can take the derivative of this expression, set it equal to zero, and

solve for k. This value will give us the optimal value of k [47]. The minimization problem is shown in Equation 4.14.

$$k_0^L = \underset{k}{argmin}\ D = (C^{ENG} - E[C_U(p;k,p_0)]_{p_0=\theta^*})^2 \tag{4.14}$$

The derivative of D with respect to k is

$$\frac{\partial D}{\partial k} = 2 \cdot (E[C_U(p;k,p_0)] - C^{ENG}) \cdot \frac{\partial E[C_U(p;k,p_0)]}{\partial k} \tag{4.15}$$

and

$$\frac{\partial E[C_U(p;k,p_0)]}{\partial k} = \frac{\partial}{\partial k}[k \cdot \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i)]. \tag{4.16}$$

Now, the derivative is very simple since everything except k is a constant; finally, $\frac{\partial E[C_U(p;k,p_0)]}{\partial k}$ is shown as

$$\frac{\partial E[C_U(p;k,p_0)]}{\partial k} = \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i). \tag{4.17}$$

Now, substituting Equation 4.17 into Equation 4.15 and setting this expression equal to zero yields Equation 4.18.

$$2 \cdot (E[C_U(p;k_0^L,p_0)] - C^{ENG}) \cdot \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i) = 0 \tag{4.18}$$

The expression for $E[C_U(p; k_0^L, p_0)]$ is given as

$$E[C_U(p; k_0^L, p_0)] = k_0^L \cdot \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i). \tag{4.19}$$

Then, substituting the expression for $E[C_U(p; k_0^L, p_0)]$ into Equation 4.18 yields

$$2 \cdot (k_0^L \cdot \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i) - C^{ENG}) \cdot \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i) = 0. \tag{4.20}$$

Dividing both sides of the Equation 4.20 by $2 \cdot \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i)$ results in

$$k_0^L \cdot \sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i) - C^{ENG} = 0. \tag{4.21}$$

With some simple algebra, the solution for k_L^0 is completed and shown in Equation 4.22.

$$k_L^0 = \frac{C^{ENG}}{\sum_{i=1}^{c} (p_i - p_0)^2 \cdot f(p_i)} \tag{4.22}$$

Now that we have the initial lower bound on k^*, we must find an initial upper bound on k^*. This is found by numerically evaluating $E[C(p; k, p_0)]$ with large values of k until we observe a value of k where $E[C(p; k, p_0)] > C^{ENG}$. Then, this k is set to the initial upper bound for k, denoted k_0^U. Now, that we have an upper and a

lower bound on k^*, we can proceed with the bisection method. However, it is also interesting to observe how much confidence changes as k changes; this is developed next.

4.4.2.2 Change in Expected Loss

Define $C = E[C(\bullet; k, p_0)]$ and $\Delta C = E[C(\bullet; k + \Delta k, p_0)] - E[C(\bullet; k, p_0)]$. Given a step of Δk, we wish to find ΔC for use in the bisection method. Consider p_i, $i = 1, ..., c$, to be the histogram centers (i.e., discretized p into c histogram bins). Define r to be the smallest index of p_i such that $p_i > p_0$. Define s as the largest index of p_i such that $k(p_i - p_0)^2 < 1$. Define t to be the largest index of p_i such that $(k + \Delta k)(p_i - p_0)^2 < 1$. With these definitions, we can determine how much C changes as k changes by evaluating the expression for ΔC. There are three cases to evaluate: $s = t$, $s < t$, and $s > t$. Figure 4.16 shows case 1. Figure 4.17 shows case 2. Figure 4.18 shows case 3. For simplicity, no histograms are shown, but the histogram centers corresponding to r, s, and t are marked accordingly.

In the first case, $s = t$, the magnitude of the change in k is so small that the limits of the summation have not changed. Equations 4.23 and 4.25 simplify the expression $E[C(\bullet; k + \Delta k, p_0)] - E[C(\bullet; k, p_0)]$.

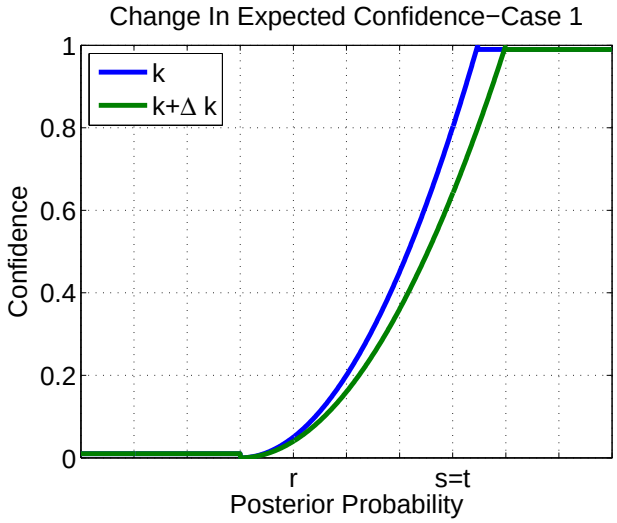

Figure 4.16 Change in Expected Loss-Case 1.

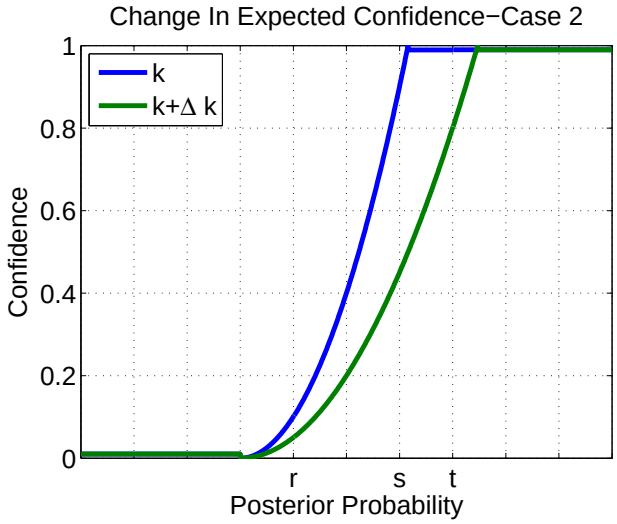

Figure 4.17 Change in Expected Loss-Case 2.

$$\Delta C = \sum_{i=r}^{t} (k+\Delta k)(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=t+1}^{c} f(p_i) - \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=s+1}^{c} f(p_i) \tag{4.23}$$

Since $s = t$ in the first case, the

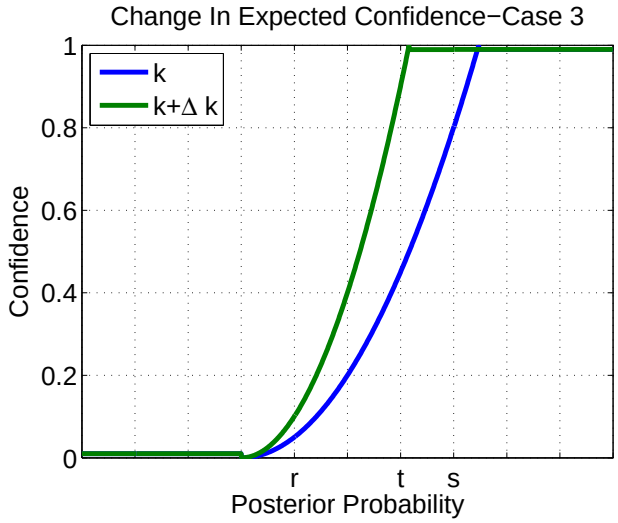

Figure 4.18 Change in Expected Loss-Case 3.

$$\Delta C = \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=r}^{s} \Delta k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=s+1}^{c} f(p_i) - \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=s+1}^{c} f(p_i) \tag{4.24}$$

which simplifies to

$$\Delta C = \sum_{i=r}^{s} \Delta k(p_i - p_0)^2 \cdot f(p_i). \tag{4.25}$$

It should be noted that this case, $s = t$, is a special instance of the other two cases where there are no restrictions on the sign of Δk.

In the second case, $\Delta k < 0$ so $s < t$. The expression for $E[C(\bullet; k + \Delta k, p_0)] - E[C(\bullet; k, p_0)]$ becomes

$$\Delta C = \sum_{i=r}^{s} (k+\Delta k)(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=s+1}^{t} (k+\Delta k)(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=t+1}^{c} f(p_i) - \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=s+1}^{c} f(p_i)$$

$$\Delta C = \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=r}^{s} \Delta k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=s+1}^{t} (k+\Delta k)(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=t+1}^{c} f(p_i) - \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=s+1}^{t} f(p_i) - \sum_{i=t+1}^{c} f(p_i)$$

$$\Delta C = \sum_{i=r}^{s} \Delta k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=s+1}^{t} (k+\Delta k)(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=s+1}^{t} f(p_i). \tag{4.26}$$

In the third case, $\Delta k > 0$ so $s > t$. The expression for $E[C(\bullet; k+\Delta k, p_0)] - E[C(\bullet; k, p_0)]$ becomes

$$\Delta C = \sum_{i=r}^{t} (k+\Delta k)(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=t+1}^{c} f(p_i) - \sum_{i=r}^{s} k(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=s+1}^{c} f(p_i)$$

$$\Delta C = \sum_{i=r}^{t} k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=r}^{t} \Delta k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=t+1}^{s} f(p_i) + \sum_{i=s+1}^{c} f(p_i) - \sum_{i=r}^{t} k(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=t+1}^{s} k(p_i - p_0)^2 \cdot f(p_i) - \sum_{i=s+1}^{c} f(p_i)$$

$$\Delta C = \sum_{i=r}^{t} \Delta k(p_i - p_0)^2 \cdot f(p_i) + \sum_{i=t+1}^{s} f(p_i) - \sum_{i=t+1}^{s} k(p_i - p_0)^2 \cdot f(p_i). \tag{4.27}$$

4.4.2.3 Bisection Method

Given an initial upper bound, k_U^0, and lower bound, k_L^0, we can start the bisection method. For iteration i, calculate Δk, calculate ΔC, the change in expected confidence at that point, and calculate $E[C(p; k_i, p_0)]$. If $E[C(p; k_i, p_0)] > C^{ENG}$, $k_{i+1}^U = k_i$ and $k_{i+1}^L = k_i^L$; otherwise, $k_{i+1}^L = k_i$ and $k_{i+1}^U = k_i^U$. Then, we can calculate $k_{i+1} = \frac{k_i^U + k_i^L}{2}$ and $\Delta k = k_i - k_{i-1}$. This process is iterated until $|E[C(p; k_i, p_0)] - C^{ENG}| < \epsilon_{conf}$ where ϵ_{conf} is the tolerance on the bisection method. When $|E[C(p; k_i, p_0)] - C^{ENG}| < \epsilon_{conf}$, $k^* = k_i$.

4.5 Final Form of the Quadratic Confidence Function

Using a quadratic confidence, we have fit the best function for the paradigm. This procedure is repeated for all classes. This confidence function, $C_d(p; k^*, p_0)$, uses a posterior probability estimate, p, as input and outputs a confidence. This confidence measure adjusts the posterior probability based upon the values of the engineer. Here, indication confidence is not equal to the posterior probability estimate but is related to it. This confidence measure directly incorporates the preference structure of the engineer and links traditional decision analysis techniques and pattern recognition techniques.

4.6 Kullback-Leibler Distance

It is vital to the application of this paradigm that the histogram of the posterior probabilities from the test set be similar to the histogram of the posterior probabilities from the training set. This is typically the case when the training and test sample sizes are relatively large. However, in small sample size cases, this is not always true since it is more likely that the training set is not completely representative of the population as a whole. To alleviate this problem, we will implement a new methodology that will help measure how well the classifier generalizes to an independent data set. Only classifiers that generalize well enough to an independent data set, as measured through Kullback-Leibler (KL) distance, will be considered further in our paradigm. In the early stages of the research, a multiple layered perceptron was being used as a classifier. Because the MLP initializes its weights randomly, there are times when the fully trained classifier does not perform well. To alleviate this affect, the following methodology was developed. KL distance has been used extensively in pattern recognition problems as it measures the distance between two probability distributions, $f(p)$ and $g(p)$ over the same variable p where $f(p)$ is the true distribution, and $g(p)$ is the estimated distribution[17]. The KL distance between distributions is shown as

$$D_{KL}(f,g) = \sum_{p} g(p) \ln \frac{g(p)}{f(p)}. \tag{4.28}$$

In our scenario, we treat the distribution of the target class posterior probability estimates from the training set as the truth data, $f(p)$, and the distribution of the target class posterior probability estimates from the test set as the observed data, $g(p)$. If the classifier generalizes well to an independent data set, then the distribution from the test set should be similar to that of the training set.

The variable p was already divided up into c intervals when the histograms were created, and c was held constant across the training set and test set. Then, D_{KL} is the distance between the two distributions. If this distance is small, then the two distributions are similar indicating that the classifier performance on the test set is similar to its performance on the training set (i.e., the classifier is generalizing well). If this distance is large, then the two distributions are not similar indicating that the classifier performance is not similar on training set and test set (i.e., the classifier is not generalizing well). The one remaining issue is how small does the KL distance have to be before it becomes significant. Define ϵ_{KL} as the user defined tolerance on KL distance. If $D_{KL} \leq \epsilon_{KL}$, we consider the classifier adequately trained. Otherwise, we consider the classifier inadequately trained and remove it from further consideration within our paradigm.

4.7 Implementation of the Confidence Function

Now that a general confidence function has been parameterized for each class, the question remains of how to use the functions. There are three options for appli-

cation of the confidence score. First, as is current practice, the posterior probability estimate for only those exemplars with supreme confidence (i.e., confidence score equal to 1) are used for classification decisions. As stated earlier, this is the approach that is used by forced decision and rejection region methodologies. The second option for application is to choose another threshold (i.e., other than 1) and only use the posterior probability estimates for those exemplars with confidence score greater than the threshold. The following summarizes the first two options. Only exemplars with confidence scores greater than a threshold will be used for classification purposes. Let L_{id} be the output of the parameterized confidence function for the i^{th} exemplar of the d^{th} class. Let θ_C be the confidence threshold. Then, exemplar i is classified according to its posterior probability estimate if the posterior probability corresponding to class d exceeds the Bayes optimal threshold and $L_{id} > \theta_C$. Otherwise, exemplar i is considered a non-declaration. While this still results in a non-declaration window, θ_C does provide some insight into the minimum confidence associated with a given window. In the first option, $\theta_C = 1$; in the second option, $0 \leq \theta_C < 1$. The third option has a random component involved. Here, the confidence function is interpreted as the probability of a declaration given a posterior probability, $P(DEC|p)$. Thus, we can compare a random number to the output of the confidence function; if the random number exceeds the output of the confidence function, the exemplar is considered a non-declaration. If the output of the confidence function exceeds the random number, a declaration is made

based upon the posterior probability estimate. For example, if many exemplars had $C(p) = 0.80$ and a random number was drawn for each exemplar, approximately 80% of these exemplars would be chosen as declarations and 20% would be chosen as non-declarations.

4.8 Summary of the Confidence Paradigm

In this chapter, we develop a theoretical confidence framework based on the assumptions that the classification system confidence acts like a value and that indication confidence can be modelled as a function of the posterior probability estimates. Classification system confidence is modelled using multiattribute preference theory, and indication confidence is modelled as a quadratic function. The tactical issues involved with fitting such a curve are addressed. A new use for the KL distance is developed. Finally, three implementations of the confidence function are discussed.

5. The Confidence Paradigm for In-Library and Out-of-library Problems

5.1 In-Library and Out-of-Library Problems

Previously, we have discussed forced decision classification and classification schemes that allow for non-declarations. Classification problems can be further broken down into two additional categories: in-library (IL) and out-of-library (OOL). For both cases, only a certain discrete number of target types are used to train a classifier. That is, a classification system has data on a certain number of target types, and that data is used to train the classification system to distinguish between those classes. If, in practice, the classification system only observes the target types used to train the classifier, then the problem is considered an "in-library" problem. However, if, in practice, the classification system observes new target types, not used to train the classification system, then the problem is considered an "out-of-library" problem [4].

When the confidence paradigm is applied to a classification system on only in-library targets, the confidence paradigm presented in Chapter 4 can be applied directly where IL non-declarations are used. When an out-of-library detector is available, this can be treated as another classifier and the paradigm developed in Chapter 4 can be applied directly to this classifier. The remainder of this chapter

discusses the OOL problem, a new OOL detector. This leads to the new concept of OOL non-declarations

5.2 The OOL Problem

As stated in [4], an out-of-library target class is one that was not used to train the classifier. This chapter introduces a new out-of-library (OOL) detector through the use of a generalized regression neural network (GRNN). Once this OOL detector is described, we develop the idea of OOL indication confidence along with the new concept of an OOL non-declaration.

For this research, an OOL detector is a classifier that discriminates between in-library (IL) targets and OOL targets based upon the features of those targets. As such, we desire that this OOL detector output a posterior probability associated with each class. We now develop the methodology to develop such an OOL detector.

What makes the OOL problem difficult is that since the OOL targets are, by definition, not in the library, we do not have exemplars to use to train a classifier in the traditional sense. One could argue that it is the idea that "we don't know what we don't know" that makes the OOL problem difficult. The OOL detector presented below is based on an amendment to the previous proposition. It is restated as "we don't know what we don't know," but "we do know that we don't know it."

Consider the two-class problem where each exemplar has two features. For each class, the two-dimensional feature space has a multivariate normal distribution.

Let the mean of the class 0 be (0, 0), and let the mean of class 1 be (1.95, 2.15). Let both classes have a covariance matrix equal to the identity matrix. Let each class have 200 observations. The observations in the feature space are shown in Figure 5.1.

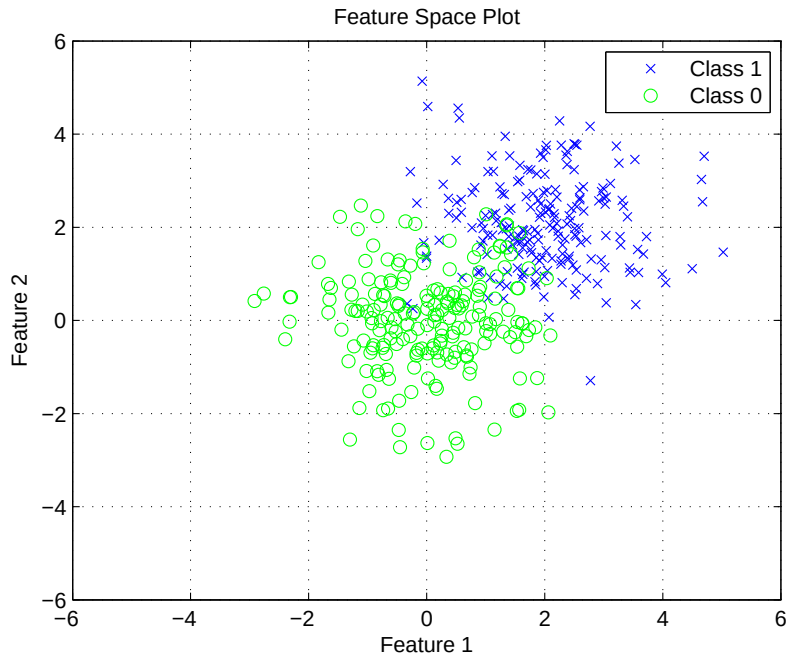

Figure 5.1 Two-Class Feature Space Plot (200 Observations Each).

Now, assume that there is another class that is unknown to our classifier (i.e., an OOL class). Let the features of this class also follow a multivariate normal distribution with mean (1.95, -2.15) and a covariance matrix equal to the identity. Figure 5.2 shows the feature space with the OOL class included.

We desire an OOL detector that can tell the difference between the in-library classes, classes 0 and 1, collectively, and the OOL class; this becomes our new two-class problem. This problem is portrayed in Figure 5.3.

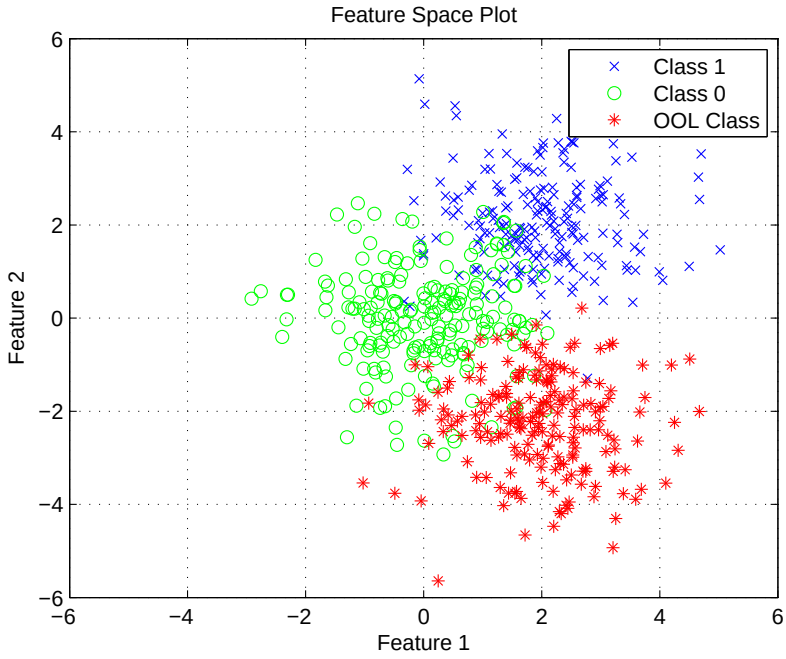

Figure 5.2 Two-Class Feature Space Plot with OOL Class (200 Observations Each).

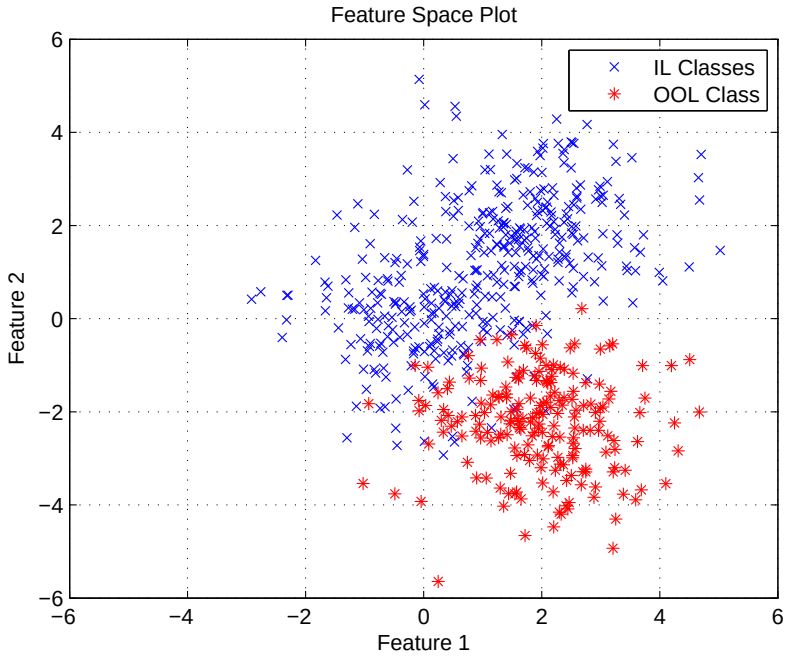

Figure 5.3 Two-Class OOL Problem (600 Observations Total).

To emphasize the point made earlier, the hard part about the OOL problem is that we do not have knowledge of the OOL class in training. Hence, the only data available to train the classifier is the features of the IL classes. This feature space which is simply the combined in-library classes is shown in Figure 5.4.

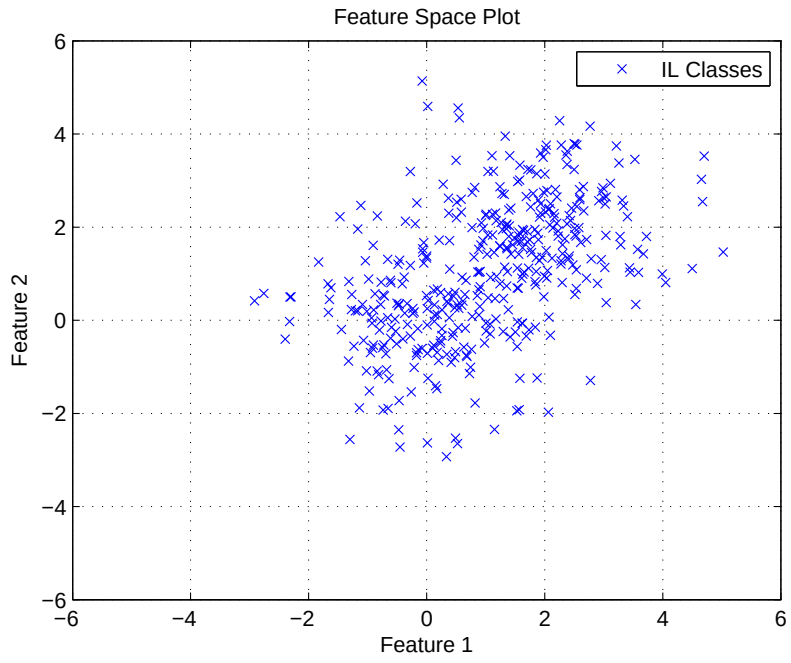

Figure 5.4 In-Library Features (400 Observations Total).

5.3 New OOL Detector

There is information that has not been used yet. That is, there are regions of the feature space where we have not observed any in-library targets. These regions yield the points to be used as features for the out-of-library class. We just need to know how to choose these points. Here, we discuss two types of data: observed data and data generated from a bounded, discretized feature space. We propose to bound

and discretize the feature space, assign each discrete point to either the IL class or the OOL class, and treat those discretized features as input to a GRNN. For each data point, we calculate the Mahalanobis Distance between the data point and each of the IL classes. For this example, we bound the feature space to be the square [-6,6]x[-6,6] and discretize the space by using integers in this range. Thus, for this simple example, we have 13 settings per feature for a total of 169 discrete points. Figure 5.5 shows the Mahalanobis Distance between each discrete point and class 0, and Figure 5.6 shows the Mahalanobis Distance between each discrete point and class 1.

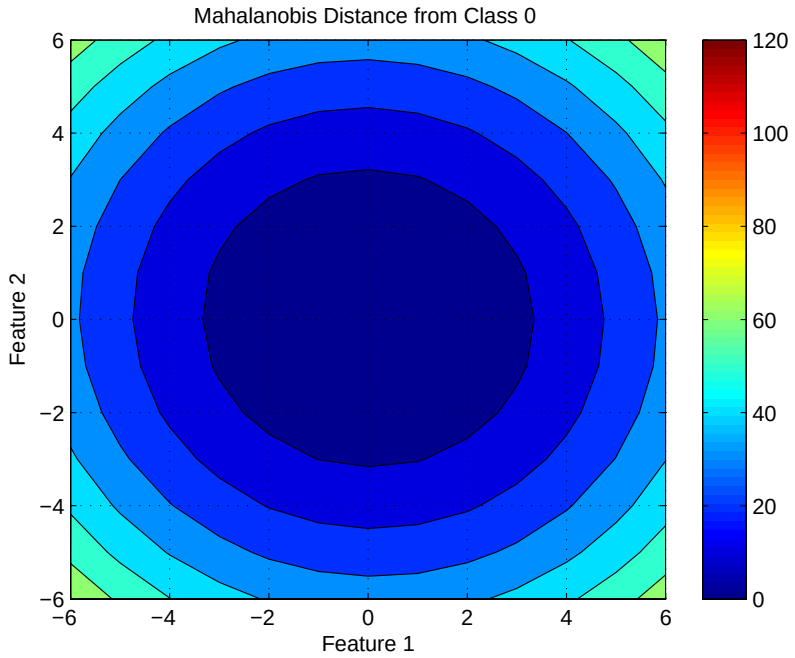

Figure 5.5 Mahalanobis Distance between Discrete Points and Class 0.

It is desirable for an out-of-library point to be far from both in-library classes so we calculate the minimum Mahalanobis Distance (across classes) for each of the

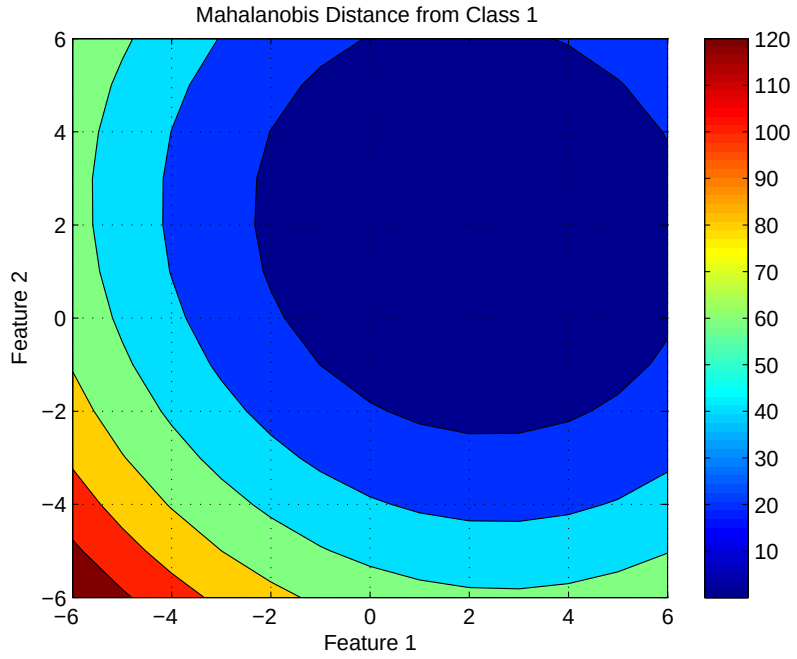

Figure 5.6 Mahalanobis Distance between Discrete Points and Class 1.

discrete points. Figure 5.7 shows the minimum Mahalanobis Distance for each of the discrete points.

For a discrete point, if the minimum Mahalanobis Distance is less than a threshold, the point is considered an IL point and is included in the training set for the IL class. However, if the minimum Mahalanobis Distance is greater than the threshold, the point is considered an OOL point and is included in the training set for the OOL class. For this low-dimensional feature space, a single threshold can be used. Using a single threshold, we can divide the entire discretized feature space into either IL or OOL. Figure 5.8 shows the discretized features space divided into IL (blue) and OOL (red) using a single threshold.

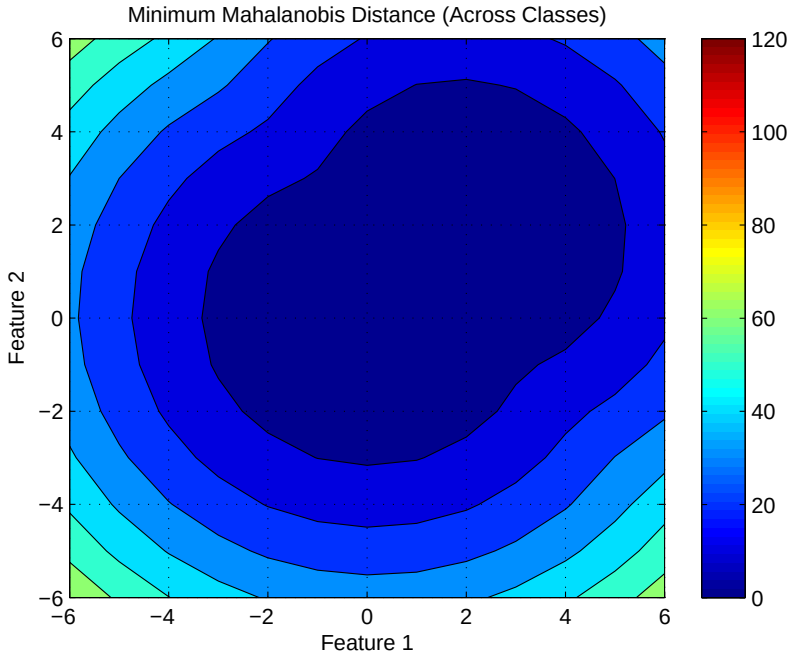

Figure 5.7 Mahalanobis Distance for Discrete Points.

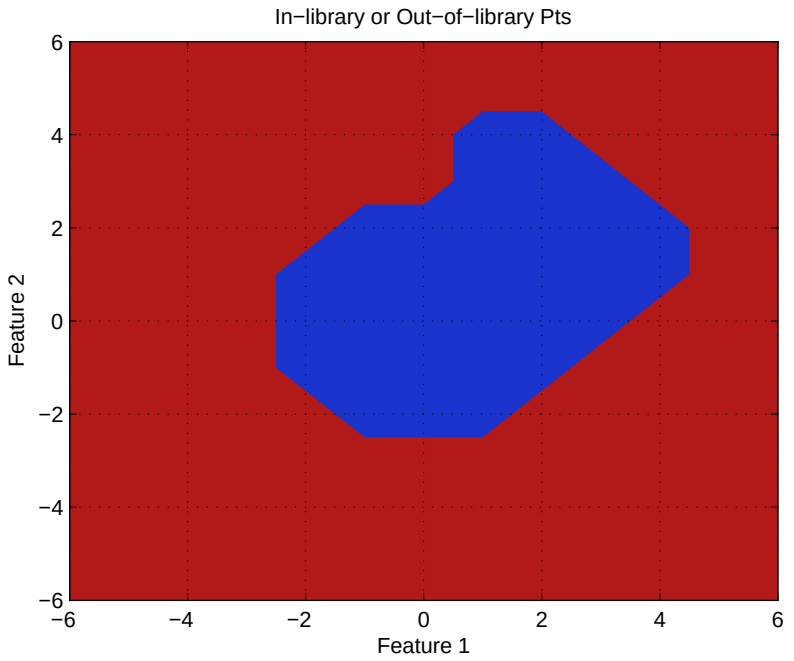

Figure 5.8 GRNN Feature Map using Single Threshold.

As the dimensionality of the feature space grows, the number of discrete points to evaluate also grows. In turn, the number of data points used to train the GRNN grows. To illustrate this point, consider two problems. In the first problem, let the feature space be two-dimensional. If we divided the feature space into 10 discrete points in each dimension, we only need to evaluate $10^2 = 100$ data points. In the second problem, let the feature space be 10-dimensional which is not unreasonable for a real data set. If we divided the feature space up into 10 discrete points in each dimension, we need to evaluate $10^{10} = 10,000,000,000$ data points. So, there is an obvious benefit to reducing the number of features to be considered in the discretization process. Another way to limit the number of training points, especially in the corners of the feature space, is to implement the use of a second threshold. This second threshold helps to create a cloud around the in-library data points to be used to train the OOL detector. Figure 5.9 shows the discretized features space divided into IL (blue) and OOL (red) using two thresholds.

The reason we feel we can implement the second threshold successfully is through understanding how a GRNN will classify such points. We suspect that the GRNN will still select exemplars from the regions beyond the second threshold as OOL even though they were not used directly in training. That is, since we expect they are closer to the OOL class than the IL class, they will be classified as OOL points.

Figure 5.9 GRNN Feature Map using Two Thresholds.

Using this methodology on 30 replications of the problem described above, our OOL detector records an average OOL true positive rate of 0.90, an average OOL true negative rate of 0.82 and an average classification accuracy or 0.85. Of course, these results will vary based upon the geometry of the OOL class with respect to the IL class. It is easy to envision a scenario where the OOL class is so far separated from the IL class that the OOL detector can easily tell the difference between the two classes. Likewise, it is easy to envision a different scenario where the OOL class is so close to the IL class that the OOL detector cannot do much better than chance.

5.4 OOL Indication Confidence

Of course, this OOL detector is simply another two-class classifier; thus, it has quantities associated with it such as classification accuracy, true positive rate, true negative rate, entropy, and sample sizes. Assuming some test data is available that contains OOL and IL targets, we can calculate the class specific OOL engineering confidence values using true positive, true negative, class specific entropy, and class specific sample sizes, just as we did in Chapter 4. Using the class specific OOL engineering confidence values, we can find the class specific quadratic confidence functions. These can be used in the same fashion as Chapter 4. As such, the methodology will produce OOL confidence values in the range $[0, 1]$. This confidence value represents the confidence that an exemplar is either an in-library exemplar or an out-of-library exemplar. Since we can apply a new threshold to these confidence values, a new concept is suggested. In Chapter 4, we discussed the non-declaration option. Below, we introduce the idea of an OOL non-declaration and show that the region of the feature space where OOL non-declarations occur is a different region in the feature space than the in-library non-declaration region.

5.5 OOL Non-Declaration

Non-declaration status is an option given to classification systems when there is not enough confidence to declare an exemplar into one of the in-library classes. In previous work by Friend [22], non-declaration is only an option once an exemplar

passes as an IL target (i.e., the exemplar is not given OOL status). Thus, a non-declaration region is typically a region of the feature space where there is overlap between multiple in-library target classes. Figure 5.10 shows a traditional in-library non-declaration region.

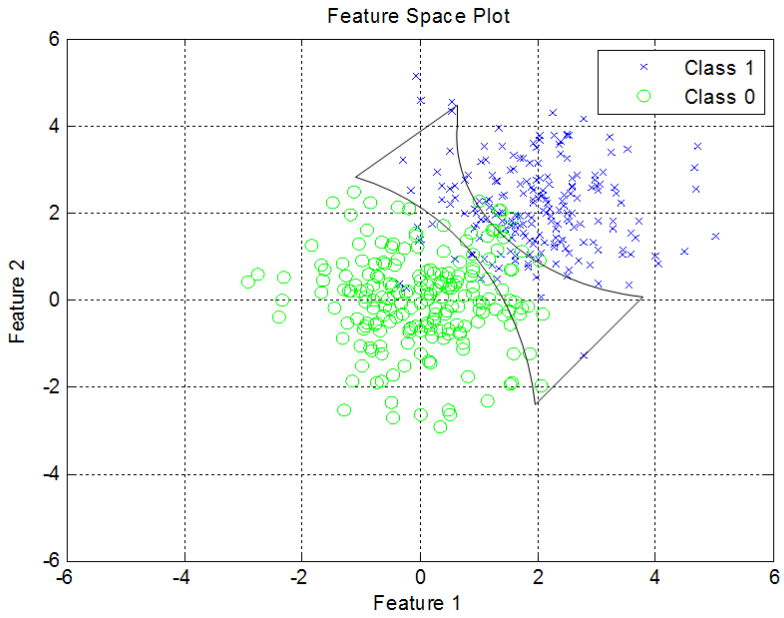

Figure 5.10 Notional IL Non-Declaration Region.

We now introduce the new concept of an OOL non-declaration region. This is the region where there is overlap between the in-library and potential out-of-library target classes. Figure 5.11 shows an OOL non-declaration region.

Now, the regions from Figures 5.10 and 5.11 can be combined to form the complete non-declaration region. Figure 5.12 shows a combined non-declaration region.

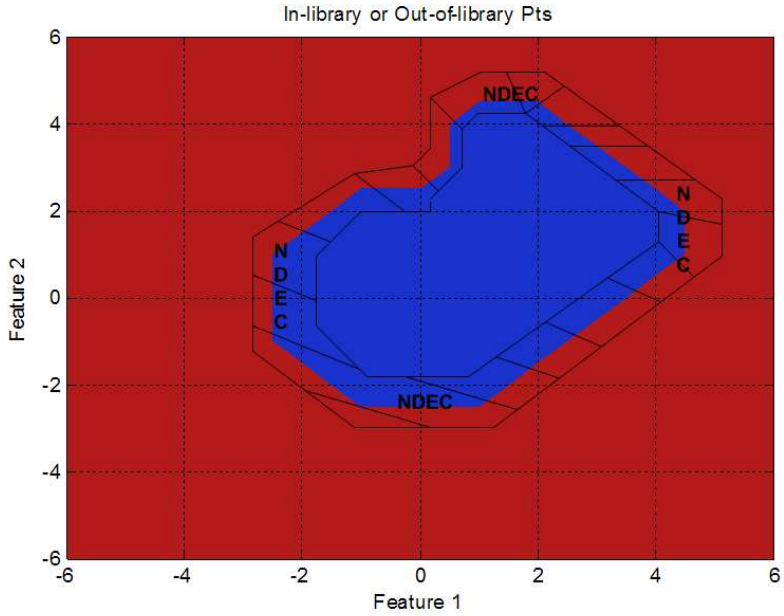

Figure 5.11 OOL Non-Declaration Region.

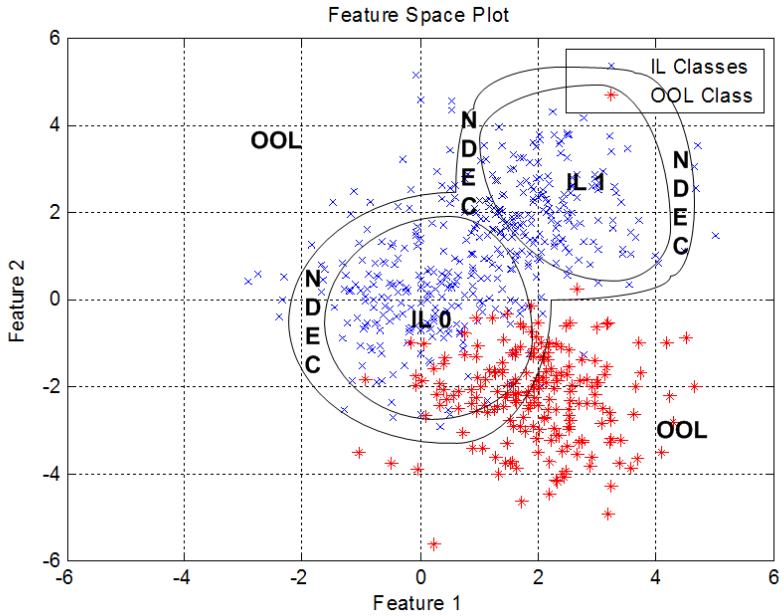

Figure 5.12 Combined Non-Declaration Region.

5.6 Summary of the Confidence Paradigm for IL and OOL Problems

This chapter started with describing the difference between an IL problem and an OOL problem in the context that our confidence paradigm can be applied to both

the IL problem as well as the OOL problem. Then, a more detailed description of the OOL problem is provided as well as a new OOL detector. This OOL detector bounds and discretizes the feature space and assigns these discrete points as either IL or OOL. These points are used to train a GRNN which acts as an OOL detector. Since this is simply another classifier, our confidence paradigm is applicable. This application produces the concept of an OOL non-declaration. The issue of combining IL confidence and OOL confidence is addressed in subsequent chapters.

6. Confidence and Non-Declarations

6.1 Overview

There are a variety of scenarios in which a classification system can be employed; three of them are discussed here: (1) in-library (IL) with non-declarations (no out-of-library (OOL) option), (2) IL with non-declarations and OOL targets with non-declarations, and (3) IL targets with non-declarations and OOL forced decision. Case (1) considers classification systems where only IL targets are present with no OOL targets. In this scenario, no OOL detector is needed, and no OOL confidence measure is needed. In this case, there is a trade-off between IL confidence and IL non-declarations. Case (2) addresses classification systems that are able to identify IL targets and OOL targets and allow for non-declarations in both realms. In this case, there is a trade-off between IL engineering confidence and IL non-declarations as well as a trade-off between OOL engineering confidence and OOL non-declarations. Finally, case (3) addresses classification systems that allow non-declarations on IL targets but force a decision on whether the target is OOL or not. In this case, there is a trade-off between IL engineering confidence and IL non-declarations, and OOL confidence is present.

6.2 IL with Non-Declarations

In a forced decision scenario, a classification system is forced to make a decision between one of the output classes. In the forced decision, there is no tradeoff between confidence and non-declarations because non-declaration status is not an option; hence, the non-declaration rate is zero when a classification system is forced to make a decision. This is the exact scenario in which the class specific, engineering confidence values are derived. This reflects how confident we are in how a classifier makes decisions on a certain class when it is forced to decide between the pre-specified output classes. This engineering confidence value is used to determine the quadratic confidence function that is applied to the posterior probability distribution. Using this confidence function, every exemplar is assigned a confidence value. A threshold can be applied to this value. If the confidence is above the threshold, the exemplar is classified according to its posterior probability estimate. If the confidence is below the threshold, the exemplar is consider a non-declaration. As we vary this threshold between 0 and 1, a rational portfolio of classification systems is created based on the user's confidence function. For each member of the portfolio, we can observe the values of certain performance measures, specifically, engineering confidence and non-declaration rate. There is an underlying trade-off between these two measures. As the threshold increases, typically, we will observe an increase in engineering confidence. That is, as questionable exemplars are removed from classification consideration, the classification accuracy tends to increase and the average

entropy tends to decrease. This causes engineering confidence to increase. Note that sample size is the number of exemplars used to train the classifier so this remains constant across the confidence threshold space. However, this increase in engineering confidence does not come without a price. The percentage of declarations continues to decrease as the confidence threshold increases. Thus, there is a tradeoff between more confident decisions and number of these decisions made.

We use multiattribute preference theory to determine the optimal alternative. Here, each threshold setting represents an alternative. Let IL engineering confidence be denoted X_{IL-EC} where where $x^0_{IL-EC} = 0$ and $x^*_{IL-EC} = 1$. Let IL non-declaration rate be denoted X_{IL-ND} where $x^0_{IL-ND} = 1$ and $x^*_{IL-ND} = 0$. Value functions will be developed for both IL engineering confidence, $v_{il-ec}(x_{il-ec})$ and IL non-declaration rate, $v_{il-nd}(x_{il-nd})$. They will be combined using a weighted linear overall value function, $v_{overall} = w_{il-ec} \cdot v_{il-ec}(x_{il-ec}) + w_{il-nd} \cdot v_{il-nd}(x_{il-nd})$ where w_{il-ec} is the weight associated with IL engineering confidence and w_{il-nd} is the weight associated with IL non-declaration rate. Figure 6.1 shows the individual value functions for IL engineering confidence and IL non-declaration rate.

Since engineering confidence is already a value, it need no further transformation. For non-declaration, there is a region between 0 and 0.2 where the value is a constant 1. After that, the value drops off in a linear fashion until it reaches a constant 0. The higher the non-declaration rate, the lower the value assigned to that rate. The form of the overall value function will be linear. In essence, the overall

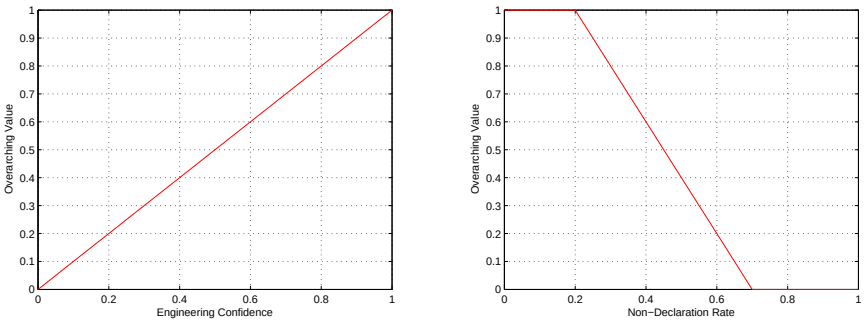

Figure 6.1 IL Individual Value Functions.

value will be a weighted average of the values assigned to engineering confidence and non-declaration rate. Figure 6.2 shows the value hierarchy for this case. Figure 6.3 shows the contour plot of overall value vs engineering confidence and non-declaration rate.

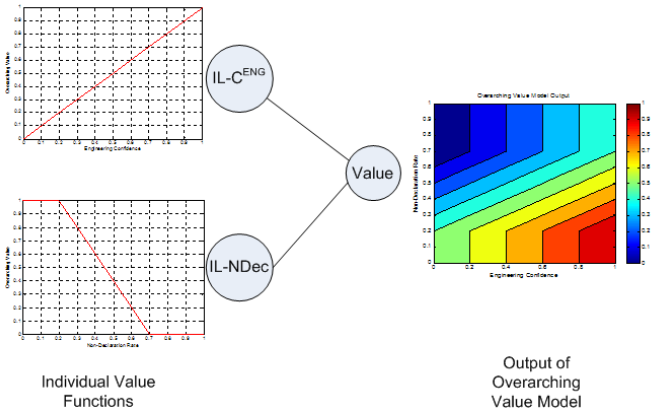

Figure 6.2 Overall Value Hierarchy-Case 1.

Each threshold setting will result in a non-declaration rate, engineering confidence pair that is input into the overall value model. The threshold setting that provides the most value is the chosen as the operating point for the paradigm.

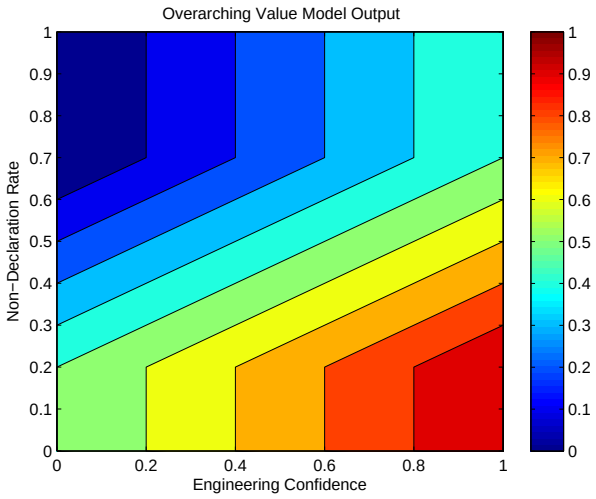

Figure 6.3 Overall Value Contour.

6.3 IL and OOL with Non-Declarations

In the case where IL and OOL targets are present and the classification system uses both IL and OOL non-declarations, we simply need to extend the model developed for the IL with non-declarations. Also, the decision space is two-dimensional as we now must find settings for both the IL threshold and the OOL threshold. Now, there are two new attributes to the decision problem. Let OOL engineering confidence be denoted X_{OOL-EC} where where $x^0_{OOL-EC} = 0$ and $x^*_{OOL-EC} = 1$. Let OOL non-declaration rate be denoted X_{OOL-ND} where $x^0_{OOL-ND} = 1$ and $x^*_{OOL-ND} = 0$. Value functions will be developed for both OOL engineering confidence, $v_{ool-ec}(x_{ool-ec})$ and OOL non-declaration rate, $v_{ool-nd}(x_{ool-nd})$.

They will be combined with the IL values using a weighted linear overall value function, $v_{overall} = w_{il-ec} \cdot v_{il-ec}(x_{il-ec}) + w_{il-nd} \cdot v_{il-nd}(x_{il-nd}) + w_{ool-ec} \cdot v_{ool-ec}(x_{ool-ec}) + w_{ool-nd} \cdot v_{ool-nd}(x_{ool-nd})$ where w_{ool-ec} is the weight associated with OOL engineer-

ing confidence and w_{ool-nd} is the weight associated with OOL non-declaration rate. Figure 6.4 shows the individual value functions for OOL engineering confidence and OOL non-declaration rate.

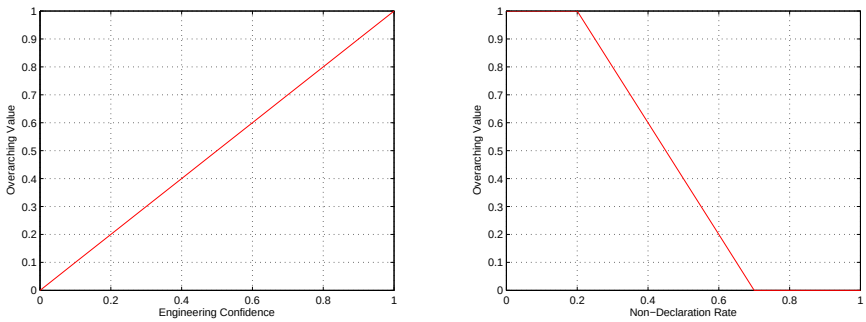

Figure 6.4 OOL Individual Value Functions.

Figure 6.5 shows the value hierarchy for this case.

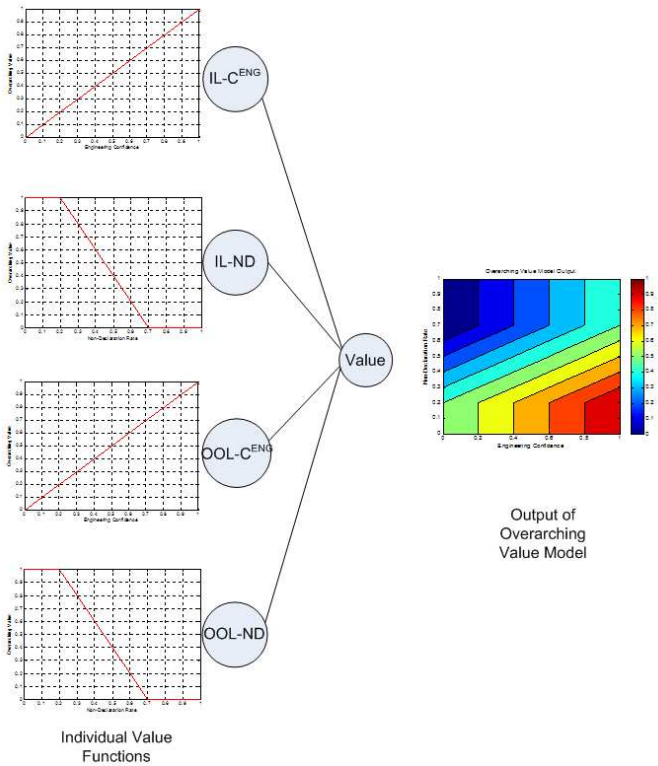

Figure 6.5 Overall Value Hierarchy-Case 2.

Each threshold setting pair will result in a four-tuple that is input into the overall value model. The threshold setting pair that provides the most value is the chosen as the operating point for the paradigm.

6.4 IL with Non-Declarations and OOL Forced Decision

In this scenario, non-declaration is not an option to the OOL detector. Thus, we only have three attributes to account for since there are no OOL non-declarations. We will further limit this scenario; this detector will only output a class label. It will not give any measurement that indicates its confidence in the class label, such as a posterior probability estimate. Note that this is the scenario in Friend's research [22] where he uses percentiles of Mahalanobis distance as his OOL detector. Without this measurement, we do not have all the information needed to calculate the full OOL engineering confidence since we cannot meaningfully calculate entropy. In this case, we will need to reduce the engineering confidence model by removing entropy and only considering classification accuracy and sample size. It is from this reduced model that OOL engineering confidence is calculated for this case. With only the three remaining attributes, the overall value model is reduced to

$$v_{overall} = w_{il-ec} \cdot v_{il-ec}(x_{il-ec}) + w_{il-nd} \cdot v_{il-nd}(x_{il-nd}) + w_{ool-ec} \cdot v_{ool-ec}(x_{ool-ec})$$

Figure 6.6 shows the value hierarchy for this case.

Even though non-declarations are not allowed in this case, there is still a threshold setting associated with the OOL detector. This threshold directly determines

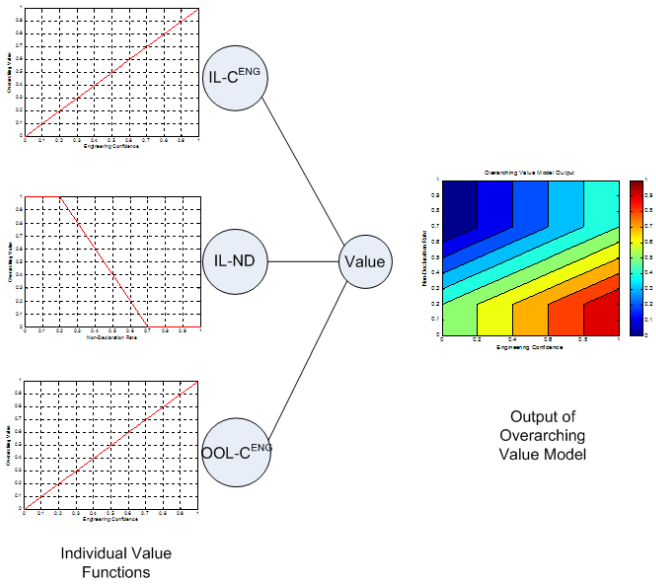

Figure 6.6 Overall Value Hierarchy-Case 3.

the classification accuracy of the OOL detector. Thus, the decision space is still two-dimensional. Each threshold setting pair will result in a three-tuple that is input into the overall value model. The threshold setting pair that provides the most value is the chosen as the operating point for the paradigm.

6.5 Summary of Confidence and Non-Declarations

For a given confidence threshold, there is a tradeoff present between engineering confidence and non-declarations. This is present for both IL problems and OOL problems. Three cases are discussed as decision problems with competing objectives. All cases are handled using value models. The optimal operating threshold for all three cases is the threshold (or threshold pair) that produces the highest value.

7. Experimentation and Results

7.1 Overview

This chapter provides results and insights from the confidence paradigm developed in this research. First, the four data sets, referred to as the Cancer, Diabetes, multivariate normal (MVN), and automatic target recognition (ATR), used in this research are described in detail. Second, results are provided on three two-class problems without out-of-library (OOL) targets: Cancer, Diabetes, and MVN. Next, results are provided on a MVN problem with OOL targets when a novel OOL detector is used. Then, OOL detector results on the ATR data set are provided. Finally, results are provided on the ATR data set with OOL targets when an OOL detector is not available.

7.2 High Level Methodology

For a given classifier family and a given data set, the following process was followed. First, the data set was split into training data, test data, and validation data, and a classifier was trained from a family of classifiers. For this classifier, the classification accuracy, average entropy, and set of posterior probability estimates were calculated for the training, test, and validation sets. The training data was used to train the classifiers, and the training sample size was used as an input into the value model to determine classifier confidence. When fusion is employed,

the training set posterior probabilities from the individual classifiers were used to train the fusion. The test set served multiple purposes. First, the classification accuracy and the average entropy from the test set were used as inputs to the value model to determine the engineering confidence. Classifier performance, specifically classification accuracy, is typically overestimated when applying the classifier to the data on which it was trained. To avoid this, an independent test set was used to estimate classification accuracy and average entropy, and these estimates were used as inputs to the value model. In addition, the threshold, p_0 is selected based upon performance in the test set, and k^* is found using the histogram of the test set. After the quadratic confidence function is fit, it is applied to both the test set and the validation set. For all the data sets except the ATR data set, this process was then repeated 30 times for different random samples of the data set and average performance across the replications is reported. As discussed in Chapter 6 and based upon results from the test set, an overarching value model can be used to determine the optimal threshold settings to be used in practice. In problem types where there are no OOL targets, only a single threshold is employed. In problem types when there are OOL targets, two thresholds are employed. Since confidence thresholds are found based upon results from the test set, it is critical that results from the test set and the validation set be similar across the range of confidence thresholds. For all of the problems studied, performance from the test set was, in fact, similar to

performance on the validation set. Hence, only performance on the validation set is shown here.

7.3 University of Wisconsin Breast Cancer and Diabetes Data Sets

The first two real world data sets used in the testing of the confidence paradigm were the University of Wisconsin breast cancer data set and the Pima Indian Diabetes data set. Since these are both two-class problems, they cannot be used directly to test any out-of-library methodology. Both data sets were taken from [48]. The University of Wisconsin breast cancer data set contains 683 exemplars, and each exemplar has 9 features. The two classes in the the data set are malignant and benign tumors. The training set contained 178 benign exemplars and 96 malignant exemplars. The test set contained 177 benign exemplars and 95 malignant exemplars. The validation set contained 89 benign exemplars and 48 malignant exemplars. Two classifiers were applied to this data set: a linear discriminant function and a quadratic discriminant function. These classifiers were applied in two ways. First, all 9 features were used to train both classifiers. Then, the data set was split into two separate feature sets: features 1-5 and features 6-9. In this case, features 1-5 were used to train the quadratic discriminant function, and features 6-9 were used to train the linear discriminant function.

The Pima Indian Diabetes data set contains 768 exemplars, and each exemplar has 8 features. The two classes in the data set are diabetics and non-diabetics. The

training set contained 200 negative exemplars and 107 positive exemplars. The test set contained 200 negative exemplars and 107 positive exemplars. The validation set contained 100 negative exemplars and 54 positive exemplars. Two classifiers were applied to this data set: a linear discriminant function and a quadratic discriminant function. These classifiers were applied in two ways. First, all 8 features were used to train both classifiers. Then, the data set was split into two separate feature sets: features 1-4 and features 5-8. In this case, features 1-4 were used to train the quadratic discriminant function, and features 5-8 were used to train the linear discriminant function.

Since real world data sets are being used here, replications cannot be arbitrarily generated as with Monte Carlo simulations. Since we do not want to draw conclusions on a single experiment, we need to perform replications of the experiment. Thus, "bootstrapping" (sampling was done without replacement) is employed. In this process, a random selection of the data is taken to train the classifier (approximately 40 %), another random selection of the data is taken to test the classifier and train the confidence paradigm (approximately 40 %), and the remaining data is left for validation (approximately 20 %).

7.4 Multivariate Normal Data Set

The MVN data set is a synthetic two-class data set where each class has two features. For each class, the two-dimensional feature space has a multivariate normal

distribution. Let the mean of the class 0 be (0, 0), and let the mean of class 1 be (0.95, 1.15). Let both classes have a covariance matrix equal to the identity matrix. Since we are using simulated data, we can generate as much data as necessary for all three data sets (training, test, and validation); thus, bootstrapping is not necessary. For all three data sets, there are 200 observations in each class. Johnson and Wichern [32] define the total probability of misclassification (TPM); this is an estimate of the error rate of the classification system. We can calculate this for the MVN problem. The TPM for this problem is 0.23. Thus, when we report classification accuracy for the forced decision on the MVN problem, we should expect classification accuracy to be approximately 0.77.

7.5 Fusion: Cancer, Diabetes, and MVN

We also want to examine fusion of multiple classifiers on the Cancer, Diabetes, and MVN problems. Research by Leap *et al.* [43] showed that probabilistic neural network (PNN) fusion outperformed Boolean fusion across a variety of synthetic problem types. To this end, PNN fusion is used as described in [43]. In PNN fusion, the posterior probabilities from the individual classifiers are used as features to a new classifier, a PNN. For each of these three problems, the quadratic and linear discriminant functions are fused and fusion results are provided. Another finding from [43] was that PNN fusion performed best when the individual classifiers had independent feature sets. We examine the effects of fusion in two scenarios. First,

we show fusion when the individual classifiers are trained on the same features sets. In this case, the classifiers feature sets have perfect correlation. In an attempt to reduce this effect, the classifiers were trained again using separate feature sets for the Cancer and Diabetes problems and independent feature sets for the MVN problem. Here, we make a distinction between separate and independent feature sets. The level of correlation can be controlled on the MVN problem, and the features given to the two classifiers are statistically independent. The level of correlation can not be controlled on the Cancer and Diabetes sets so we simply separated the features into two disjoint sets. The level of correlation between the features for the Cancer, Diabetes, and MVN data sets was calculated using a pooled correlation matrix as shown in Tables 7.1, 7.2, and 7.3.

Table 7.1 Correlation Matrix For the Features of Cancer Data Set.

	F1	F2	F3	F4	F5	F6	F7	F8	F9
F1	1	0.14	0.17	-0.03	0.06	0.01	0.03	0.04	0.08
F2	0.14	1	0.71	0.31	0.45	0.05	0.36	0.33	0.22
F3	0.17	0.71	1	0.26	0.38	0.12	0.30	0.32	0.18
F4	-0.03	0.31	0.26	1	0.21	0.22	0.29	0.19	0.19
F5	0.06	0.45	0.38	0.21	1	0.04	0.20	0.26	0.29
F6	0.01	0.05	0.12	0.22	0.04	1	0.15	-0.02	-0.02
F7	0.03	0.36	0.30	0.29	0.20	0.15	1	0.27	0.04
F8	0.04	0.33	0.32	0.19	0.26	-0.02	0.27	1	0.21
F9	0.08	0.22	0.18	0.19	0.29	-0.02	0.04	0.21	1

By examining Tables 7.1, 7.2, and 7.3, one can tell that, in general, the features from the Cancer data set are more highly correlated than the features of the Diabetes data set. Also, the features of the MVN have extremely low correlation.

Table 7.2 Correlation Matrix For the Features of Diabetes Data Set.

	F1	F2	F3	F4	F5	F6	F7	F8
F1	1	0.03	0.13	-0.10	-0.11	-0.05	-0.08	0.52
F2	0.03	1	0.14	0.03	0.31	0.10	0.06	0.18
F3	0.13	0.14	1	0.20	0.08	0.28	0.03	0.23
F4	-0.10	0.03	0.20	1	0.43	0.39	0.17	-0.14
F5	-0.11	0.31	0.08	0.43	1	0.17	0.17	-0.08
F6	-0.05	0.10	0.28	0.39	0.17	1	0.10	-0.04
F7	-0.08	0.06	0.03	0.17	0.17	0.10	1	-0.01
F8	0.52	0.18	0.23	-0.14	-0.08	-0.04	-0.01	1

Table 7.3 Correlation Matrix For the Features of MVN Data Set.

	F1	F2	F3	F4
F1	1	0.02	-0.002	0.0004
F2	0.02	1	-0.005	0.01
F3	-0.002	-0.005	1	-0.01
F4	0.0004	0.01	-0.01	1

7.6 Cancer, Diabetes, and MVN Results

We consider the results of the Cancer and Diabetes problems at the same time where in each problem, both a quadratic discriminant function and a linear discriminant function are used. Figure 7.1 plots the two inputs to the overarching value model, engineering confidence vs. non-declaration rate. On the same plot, the contours for the overarching value model are shown. Each point on the plot represents a different confidence threshold. These trajectories represent a rational portfolio of classification systems based on the user's confidence function. In addition to the trajectories, the stochastic non-declaration methodology is employed and performance is shown. The plot on the right is a zoomed-in version of the plot on the left.

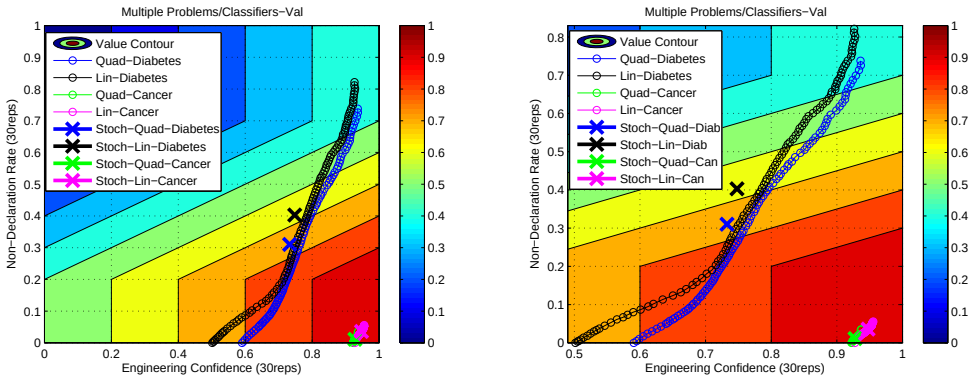

Figure 7.1 Cancer,Diabetes-Engineering Confidence vs. Non-Declaration Rate with Value Contours-Validation Set.

These plots are used to demonstrate the fact that classifier performance depends not only on the classifier employed but also the problem on which it is used. That is, typically, we examine classification systems as they are applied to a single problem. However, this confidence paradigm can be used to look across multiple problems. By examining Figure 7.1, the first observation to make is that we are more confident in either the linear or the quadratic classifiers as they are applied to Cancer problem. For any confidence threshold, the classification systems when applied to the Cancer problem yield more overarching value. This illustrates the point that the inherent confidence we have in a classification system depends upon the difficulty of problem geometry, the classifier employed, and the number of samples used to train the data. Second, within a problem, we can see which classifier we prefer. In the Diabetes problem, the quadratic classifier dominates the linear classifier. In the Cancer problem, neither classifier dominates the other across the threshold range. With these observations, we can see that this paradigm can be used

as a classifier screening tool to see which classifier you would prefer when you can choose from a variety of classification systems.

Each point on the trajectories reflects a specific confidence based rejection region. As stated in Chapter 4, there is one alternative to forming a rejection region, and that is the stochastic non-declaration procedure. For the Diabetes problem with either classifier, the stochastic non-declaration procedure is dominated by the confidence based rejection region. However, for the Cancer problem, the performance of the stochastic non-declaration procedure falls along the trajectory of the confidence based rejection region. While probably not operationally acceptable to employ this procedure, it is interesting to note that for the Cancer problem, its performance is very similar to the confidence based rejection region.

After examining four problem-classifier combinations together, we will now examine the problems separately and consider PNN fusion performance. Figure 7.2 shows three performance parameters: classification accuracy, non-declaration rate, and engineering confidence; the performance parameters are plotted vs. confidence threshold for the MVN Problem on the validation set where the features used to train both classifiers are the same. Figure 7.3 shows two performance parameters: overarching value and average entropy; the performance parameters are plotted vs. confidence threshold for the MVN Problem on the validation set where the features used to train both classifiers are the same. Figure 7.4 plots the two inputs to the overarching value model, engineering confidence vs. non-declaration rate. On the

same plot, the contours for the overarching value model are shown. Each point on the plot represents a different confidence threshold. These trajectories represent a rational portfolio of classification systems based on the user's confidence function. The plot on the right is a zoomed-in version of the plot on the left.

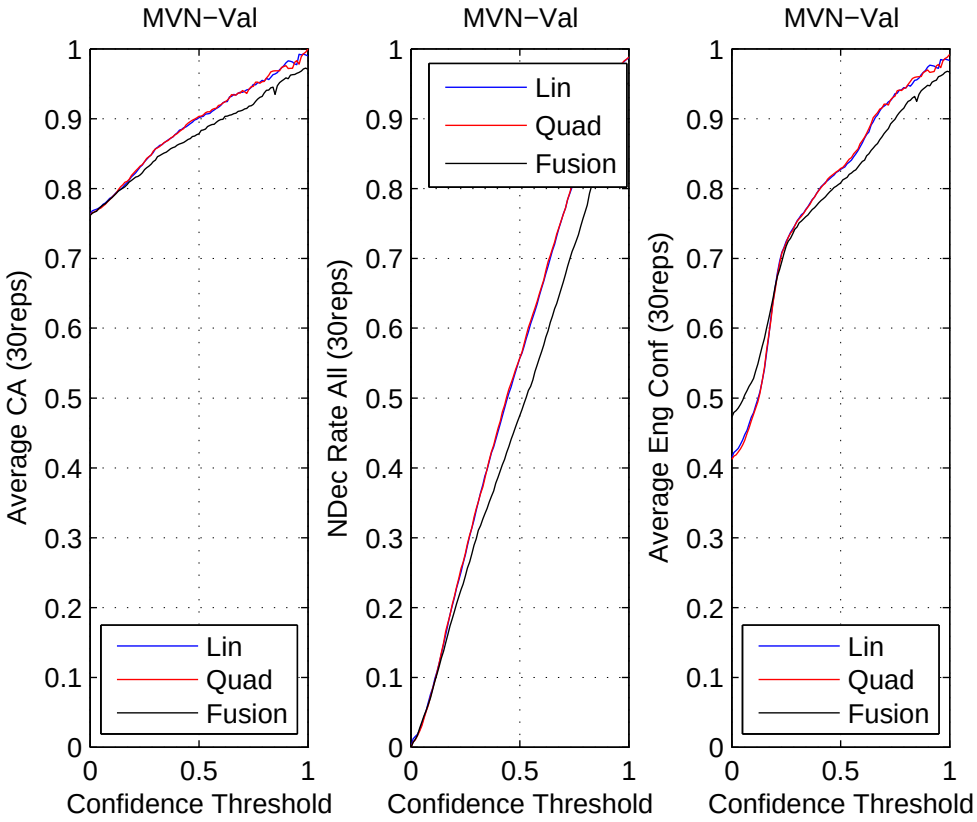

Figure 7.2 MVN-Three Performance Parameters vs. Confidence Threshold-Validation Set.

In examining the MVN problem when the feature sets observed by the individual classifiers are identical, we see that performance across all 5 performance measures is nearly identical across individual classifiers. Since the feature sets are generated to have identical covariance matrices, it is not surprising that the linear and

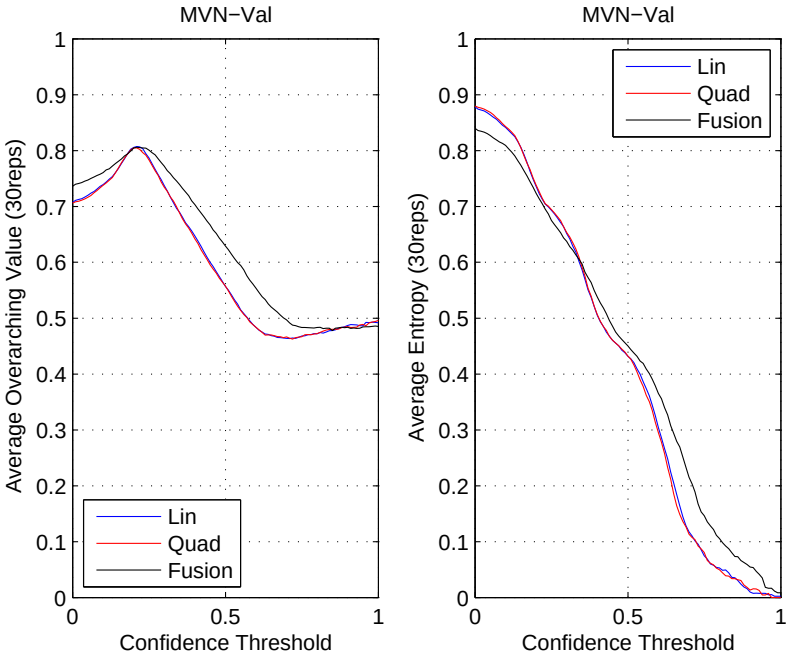

Figure 7.3 MVN-Two Performance Parameters vs. Confidence Threshold-Validation Set.

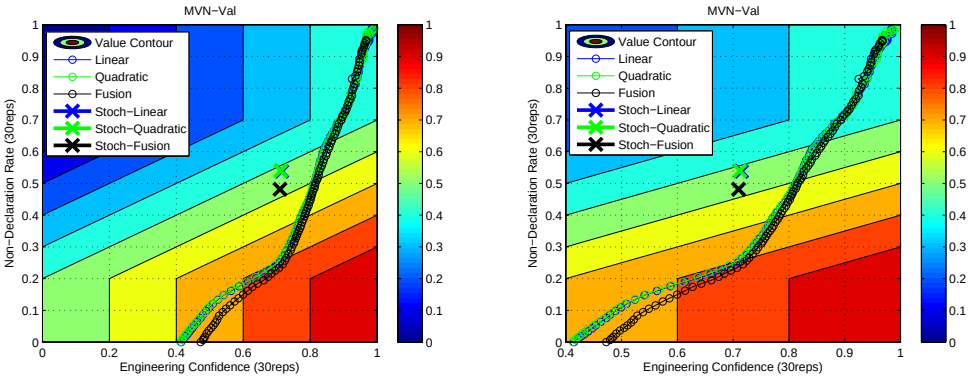

Figure 7.4 MVN-Engineering Confidence vs. Non-Declaration Rate with Value Contours-Validation Set.

quadratic classifiers perform nearly identically. With respect to overarching value, the fusion is preferred across a large range of the threshold space. This is due to a combination of lower entropy (particularly at lower confidence thresholds) and lower non-declaration rates (particularly at higher confidence thresholds). Additionally,

we see rational trends in the performance parameters for the individual classifiers and the fusion as the confidence threshold increases. That is, classification accuracy, non-declaration rate, and engineering confidence increase as the confidence threshold increases, and average entropy decreases as the confidence threshold increases. The performance of the stochastic non-declaration procedure is outperformed by the confidence based rejection region across the confidence threshold space.

Figure 7.5 shows three performance parameters, classification accuracy: non-declaration rate, and engineering confidence; the performance parameters are plotted vs confidence threshold for the Diabetes Problem on the validation set where the features used to train both classifiers are the same. Figure 7.6 shows two performance parameters: overarching value and average entropy; the performance parameters are plotted vs confidence threshold for the Diabetes Problem on the validation set where the features used to train both classifiers are the same. Figure 7.7 plots the two inputs to the overarching value model, engineering confidence vs. non-declaration rate. On the same plot, the contours for the overarching value model are shown. Each point on the plot represents a different confidence threshold. These trajectories represent a rational portfolio of classification systems based on the user's confidence function. The plot on the right is a zoomed-in version of the plot on the left.

For the Diabetes problem, when the feature sets used by the individual classifiers are identical, we observe logical trends with respect to the performance parameters. When examining overarching value, fusion performs at least as good as

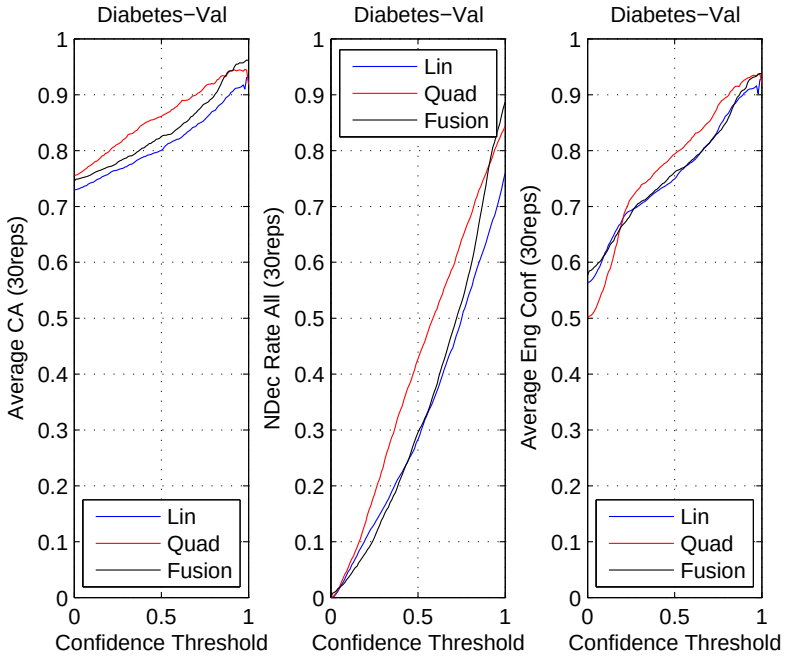

Figure 7.5 Diabetes-Three Performance Parameters vs. Confidence Threshold-Validation Set.

the worst classifier across a large range of the confidence threshold space, and fusion is preferred at extremely low confidence thresholds. Across most of the range of the confidence threshold space, the linear discriminant function and the fusion perform nearly identically with respect to overarching value. The performance of the stochastic non-declaration procedure is outperformed by the confidence based rejection region across the confidence threshold space.

Figure 7.8 shows three performance parameters: classification accuracy, non-declaration rate, and engineering confidence; the performance parameters are plotted vs confidence threshold for the Cancer Problem on the validation set where the features used to train both classifiers are the same. Figure 7.9 shows two performance

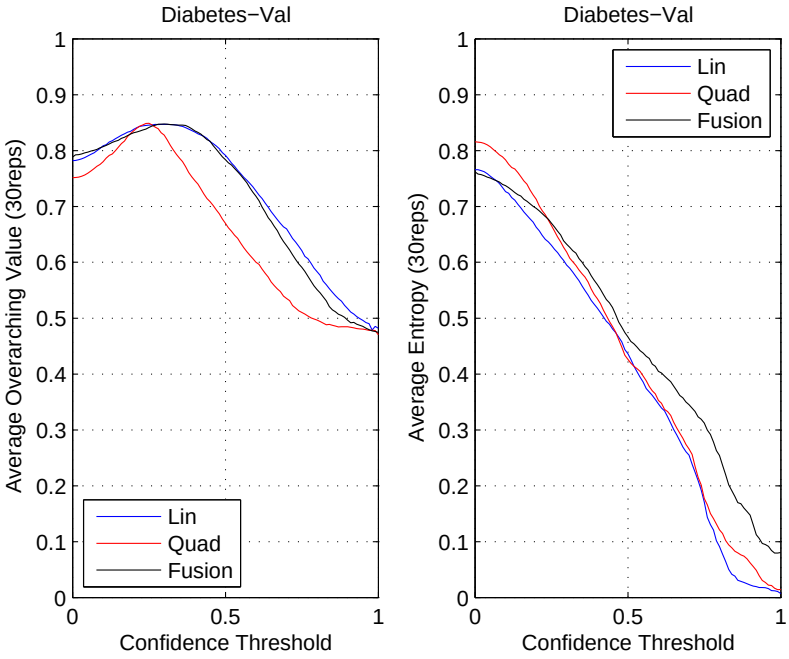

Figure 7.6 Diabetes-Two Performance Parameters vs. Confidence Threshold-Validation Set.

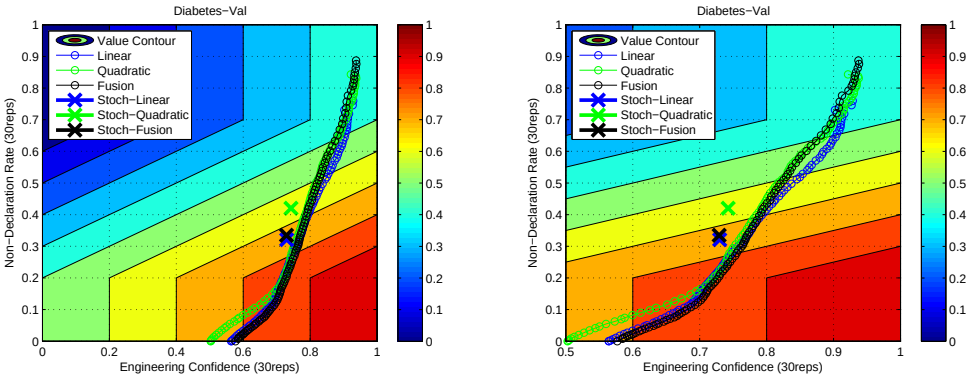

Figure 7.7 Diabetes-Engineering Confidence vs. Non-Declaration Rate with Value Contours-Validation Set.

parameters: overarching value and average entropy; the performance parameters are plotted vs confidence threshold for the Cancer Problem on the validation set where the features used to train both classifiers are the same. Figure 7.10 plots the two inputs to the overarching value model, engineering confidence vs. non-declaration

rate. On the same plot, the contours for the overarching value model are shown. Each point on the plot represents a different confidence threshold. These trajectories represent a rational portfolio of classification systems based on the user's confidence function. The plot on the right is a zoomed-in version of the plot on the left.

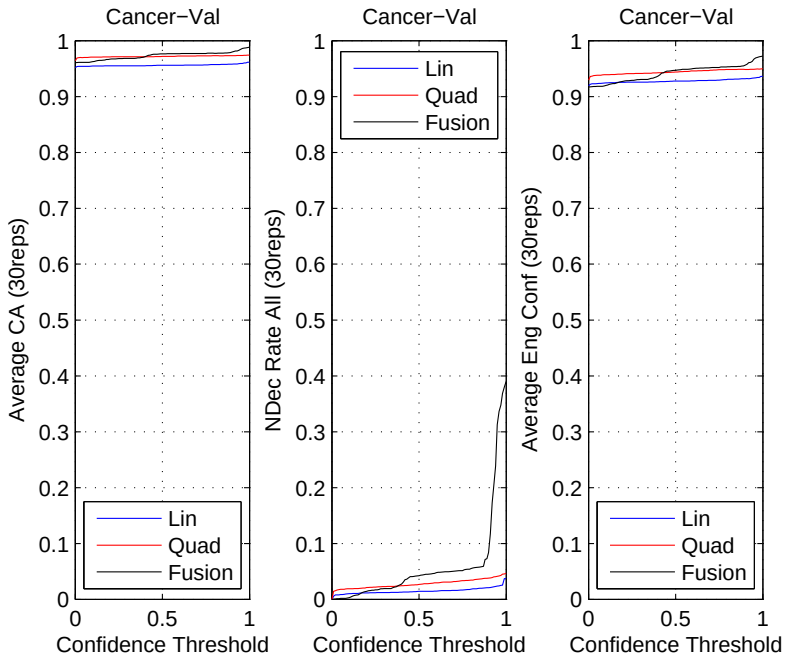

Figure 7.8 Cancer-Three Performance Parameters vs. Confidence Threshold-Validation Set.

In the Cancer problem, when the feature sets used by the individual classifiers are identical, we observe logical trends with respect to all the performance parameters. When examining overarching value, fusion is only marginally better than the quadratic classifiers across approximately half of the confidence threshold space. The performance of the stochastic non-declaration procedure is very similar to the performance of the confidence based rejection region across the confidence threshold space. For the individual classifiers, the stochastic non-declaration performance falls along

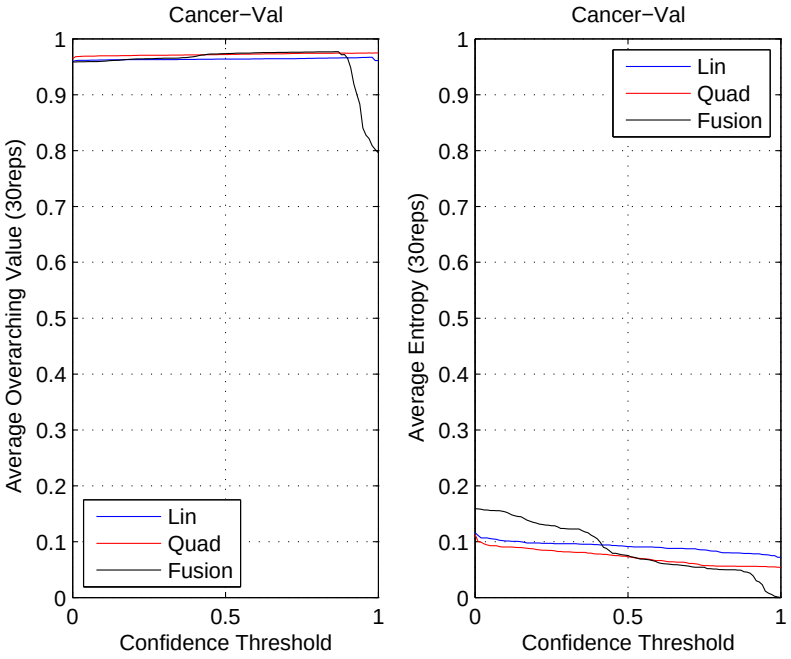

Figure 7.9 Cancer-Two Performance Parameters vs. Confidence Threshold-Validation Set.

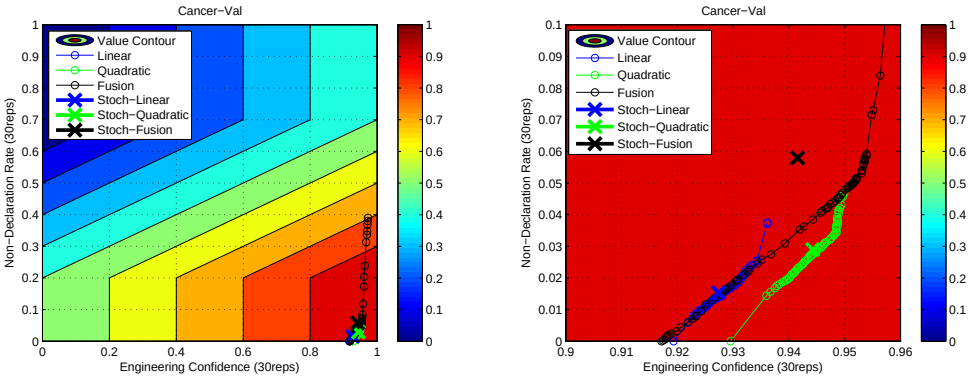

Figure 7.10 Cancer-Engineering Confidence vs. Non-Declaration Rate with Value Contours-Validation Set.

the trajectory of the confidence based rejection regions. For the fusion, the stochastic non-declaration performance is marginally outperformed by the confidence based rejection regions.

The results thus far may be surprising. While the overarching value from the fusion performs at least as good as the worst classifier for some portion of the confidence threshold space, some may expect the fusion to outperform the individual classifiers across a large region of the confidence threshold space. Leap, *et al.* [43] already found that fusion of highly correlated classifiers does not perform as well as fusion of independent classifiers so the results are less surprising. So far, the feature sets between the two individual classifiers have been identical (i.e., perfectly correlated) so there may be little to no benefit from the fusion. This is exactly what is observed. Now, let's use separate features to train the individual classifiers. This should provide the individual classifiers with less dependent information.

Figure 7.11 shows three performance parameters: classification accuracy, non-declaration rate, and engineering confidence; the performance parameters are plotted vs confidence threshold for the MVN Problem on the validation set where the features used to train both classifiers are disjoint. Figure 7.12 shows two performance parameters: overarching value and average entropy; the performance parameters are plotted vs confidence threshold for the MVN Problem on the validation set where the features used to train both classifiers are disjoint. Figure 7.13 plots the two inputs to the overarching value model, engineering confidence vs. non-declaration rate. On the same plot, the contours for the overarching value model are shown. Each point on the plot represents a different confidence threshold. These trajectories represent

a rational portfolio of classification systems based on the user's confidence function. The plot on the right is a zoomed-in version of the plot on the left.

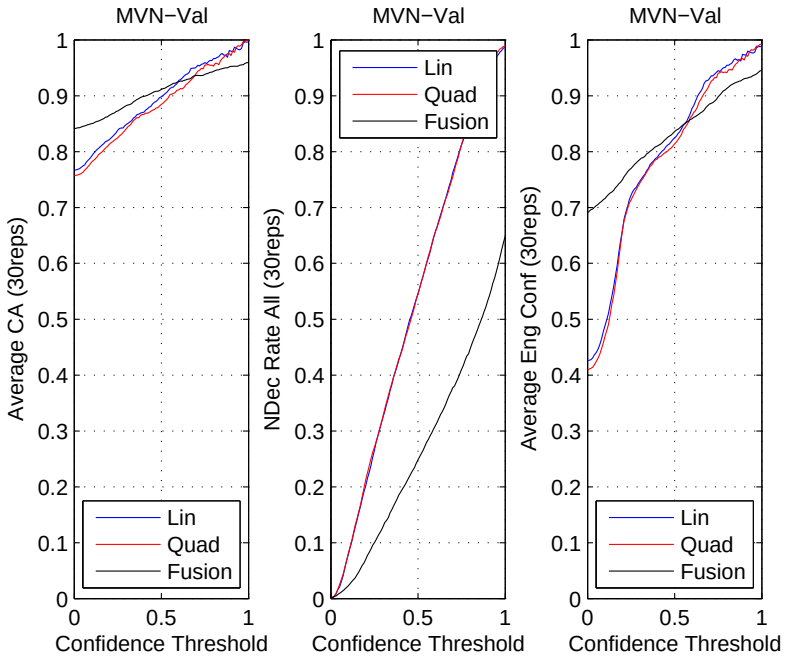

Figure 7.11 MVN-Three Performance Parameters vs. Confidence Threshold-Validation Set-Separate Features.

Figures 7.11, 7.12, and 7.13 show results for the MVN problem where the feature sets observed by the individual classifiers are disjoint. In the controlled scenario with synthetic data, we know the feature sets observed by the two individual classifiers have very low correlation. This is the scenario where we would expect the largest increase in performance between the individual classifiers and the fusion, and that is exactly what happens. In terms of overarching value, the fusion outperforms the individual classifiers across the entire confidence threshold space. The performance

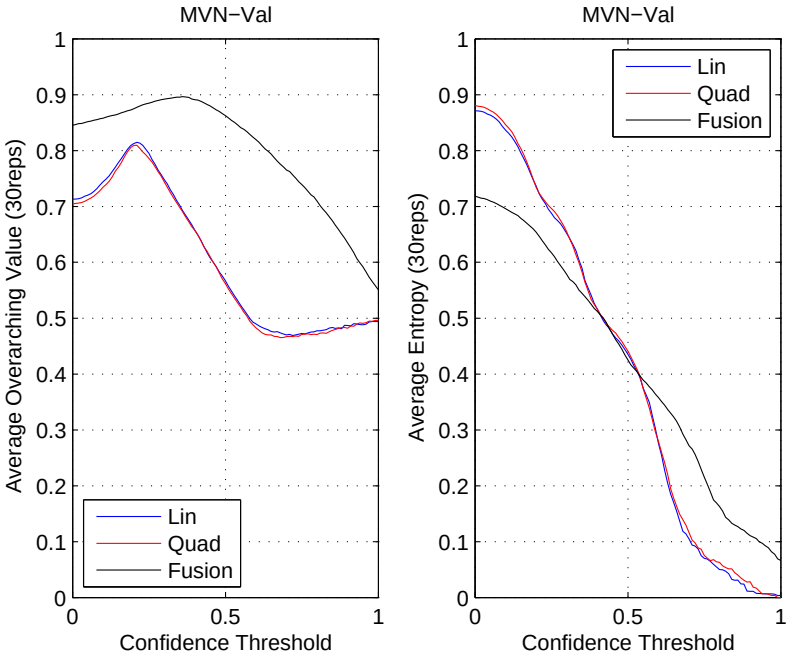

Figure 7.12 MVN-Two Performance Parameters vs. Confidence Threshold-Validation Set-Separate Features.

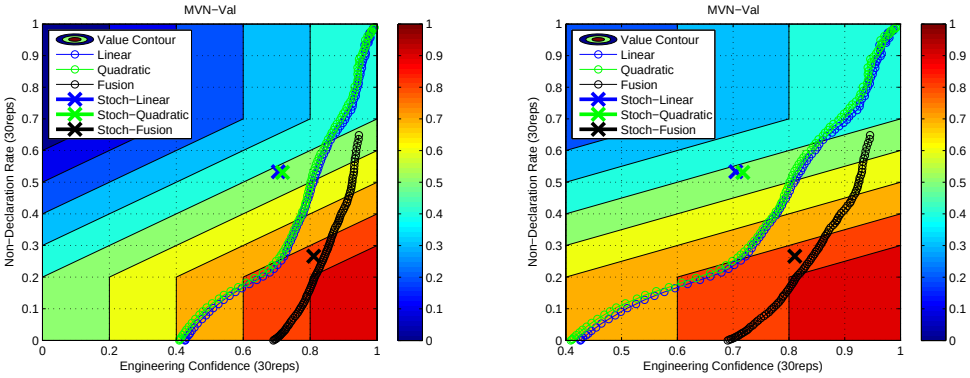

Figure 7.13 MVN-Engineering Confidence vs. Non-Declaration Rate with Value Contours-Validation Set-Separate Features.

of the stochastic non-declaration procedure is outperformed by the confidence based rejection region across the confidence threshold space.

Figure 7.14 shows three performance parameters: classification accuracy, non-declaration rate, and engineering confidence; the performance parameters are plotted vs confidence threshold for the Diabetes Problem on the validation set where the features used to train both classifiers are disjoint. Figure 7.15 shows two performance parameters: overarching value and average entropy; the performance parameters are plotted vs confidence threshold for the Diabetes Problem on the validation set where the features used to train both classifiers are disjoint. Figure 7.16 plots the two inputs to the overarching value model, engineering confidence vs. non-declaration rate. On the same plot, the contours for the overarching value model are shown. Each point on the plot represents a different confidence threshold. These trajectories represent a rational portfolio of classification systems based on the user's confidence function. The plot on the right is a zoomed-in version of the plot on the left.

Figures 7.14, 7.15, and 7.16 show results for the Diabetes problem where the features observed by the individual classifiers are disjoint. In the Diabetes problem, the fusion now outperforms the individual classifiers for almost the entire confidence threshold space. The correlation between the disjoint feature sets for the Diabetes problem was higher than the MVN problem but lower than the Cancer problem. Therefore, we expected there to be some increase in performance resulting from the fusion but not as much increase in performance as was experienced in the MVN problem. This is exactly what we observe. The performance of the stochastic non-

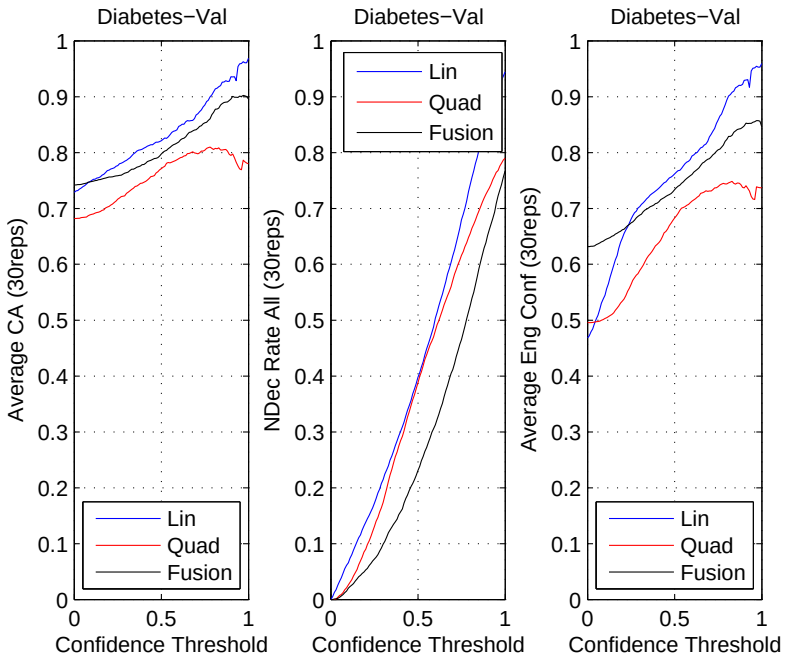

Figure 7.14 Diabetes-Three Performance Parameters vs. Confidence Threshold-Validation Set-Separate Features.

declaration procedure is outperformed by the confidence based rejection region across the confidence threshold space.

Figure 7.17 shows three performance parameters: classification accuracy, non-declaration rate, and engineering confidence; the performance parameters are plotted vs confidence threshold for the Cancer Problem on the validation set where the features used to train both classifiers are disjoint. Figure 7.18 shows two performance parameters: overarching value and average entropy; the performance parameters are plotted vs confidence threshold for the Cancer Problem on the validation set where the features used to train both classifiers are disjoint. Figure 7.19 plots the two inputs to the overarching value model, engineering confidence vs. non-declaration rate. On

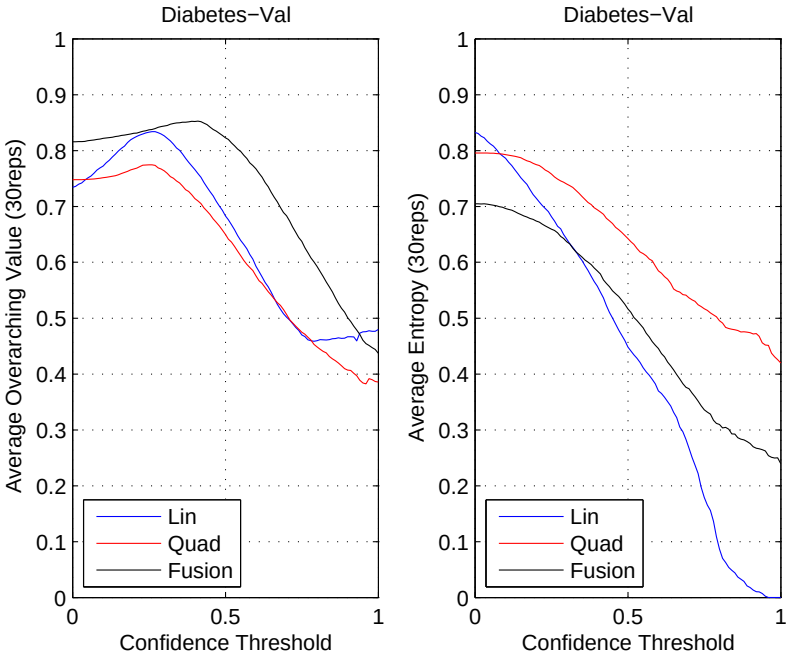

Figure 7.15 Diabetes-Two Performance Parameters vs. Confidence Threshold-Validation Set-Separate Features.

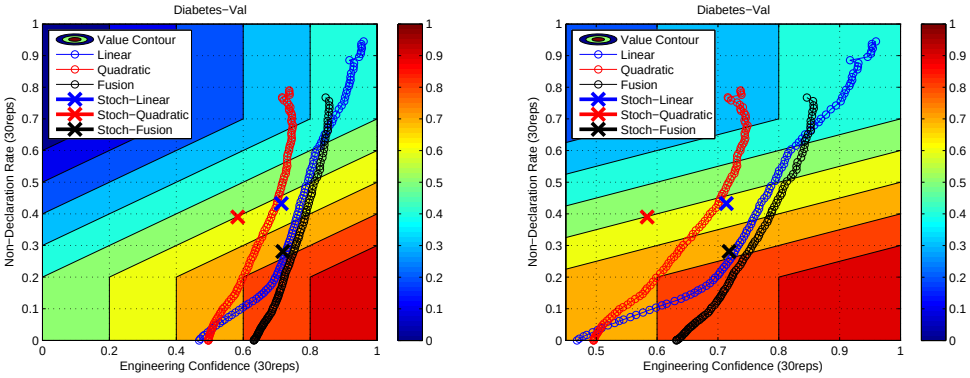

Figure 7.16 Diabetes-Engineering Confidence vs. Non-Declaration Rate with Value Contours-Validation Set-Separate Features.

the same plot, the contours for the overarching value model are shown. Each point on the plot represents a different confidence threshold. These trajectories represent a rational portfolio of classification systems based on the user's confidence function. The plot on the right is a zoomed-in version of the plot on the left.

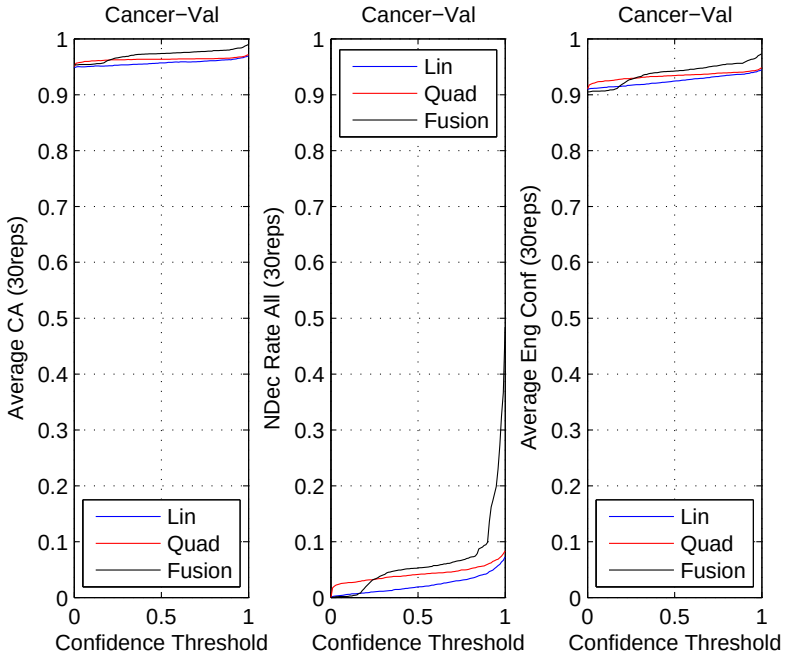

Figure 7.17 Cancer-Three Performance Parameters vs. Confidence Threshold-Validation Set-Separate Features.

Figures 7.17, 7.18, and 7.19 show results for the Cancer problem where the features observed by the individual classifiers are disjoint. In the Cancer problem, the fusion marginally outperforms the individual classifier across a large region of the confidence threshold space. Since the correlation between the disjoint feature sets was the highest for the Cancer problem, we expected the fusion in this problem to be the worst, and this is exactly what was observed. The performance of the stochastic non-declaration procedure is very similar to the performance of the confidence based rejection region across the confidence threshold space. For the individual classifiers, the stochastic non-declaration performance falls along the trajectory of the

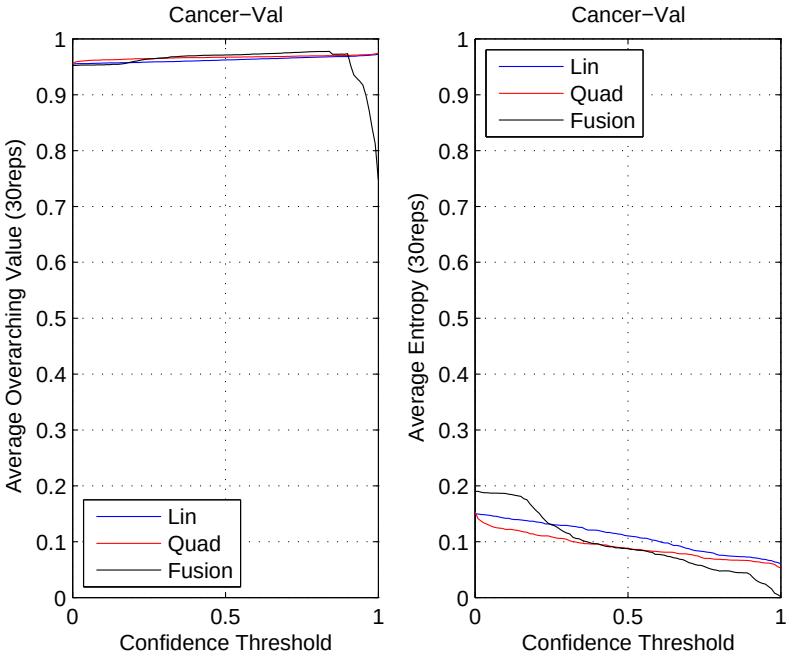

Figure 7.18 Cancer-Two Performance Parameters vs. Confidence Threshold-Validation Set-Separate Features.

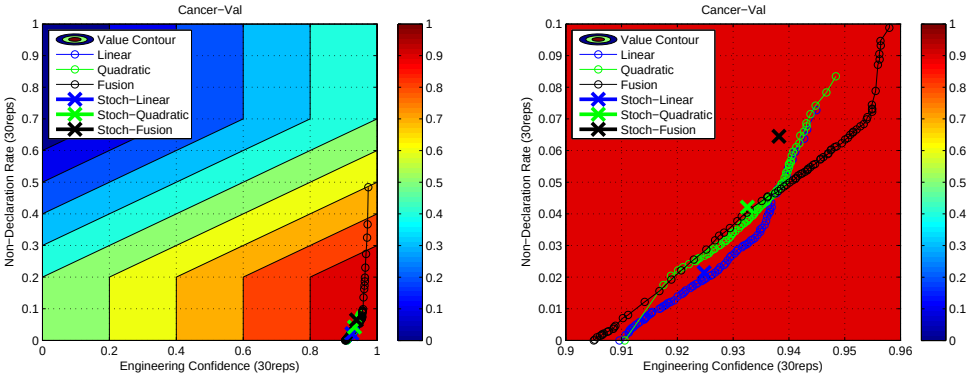

Figure 7.19 Cancer-Engineering Confidence vs. Non-Declaration Rate with Value Contours-Validation Set-Separate Features.

confidence based rejection regions. For the fusion, the stochastic non-declaration performance is marginally outperformed by the confidence based rejection regions.

We implemented the confidence paradigm on three data sets and shown rational trends in performance parameters. We've demonstrated the performance of the novel stochastic non-declaration procedure on all data sets. We have also shown how the paradigm performs across a spectrum of problem difficulties as well as different classifiers. We demonstrated performance of individual classifiers as well as PNN fusion. At this point, we have not considered data sets with OOL targets. Next, we will extend the paradigm to consider both IL and OOL target types.

7.7 Results for MVN Problem with OOL targets and OOL detector

Chapter 5 details the OOL problem and introduces the new concept of an OOL non-declaration. Chapter 6 discusses the overarching value model used on problem types where an OOL detector is available. Results in this section are reported on the multivariate problem discussed in Chapter 5 where the GRNN OOL detector is used. As discussed in Chapter 6, there are four inputs to the overarching value model: IL engineering confidence, IL non-declaration rate, OOL engineering confidence, and OOL non-declaration rate. All of the IL data is used to train the IL confidence function. A bounded and discretized feature space is used to train the OOL detector and the test set, composed of both real IL and OOL targets, is used to test the OOL detector. Quantities such as classification accuracy and entropy as well as a histogram of the posterior probability estimates from the test set are used to fit the confidence function. Once the confidence functions are fit, both IL and OOL

confidence thresholds are varied together, and an overarching value is observed. In addition, each exemplar is processed in the following serial fashion. First, the OOL detector must determine if the exemplar is IL, OOL, or OOL non-declaration. Only those exemplars that are classified as IL by the OOL detector are considered for classification by the IL classifier. The OOL entropy and OOL classification accuracy (CA) are calculated on all exemplars classified as IL or OOL by the OOL detector. The IL entropy and IL CA are calculated on those exemplars classified as IL by the OOL detector and classified as either a target or non-target by the IL classifier. Using this methodology, IL engineering confidence, IL non-declarations, OOL engineering confidence and OOL non-declarations are calculated and input into the overarching value model. The threshold pair that yields the highest overarching value is the optimal threshold pair. Figure 7.20 shows a contour plot of the overarching value model for both the test set and the validation set when varying the two thresholds together. The maximum of the overarching value model occurs when the IL threshold is set at 0.88 and the OOL threshold is set at 0.76; this point is plotted on Figure 7.20 as well. Also, we can see that performance does not change much from the test set to the validation set.

This concludes implementation of the confidence paradigm on all data sets except the ATR data set. Next, we describe the ATR data set.

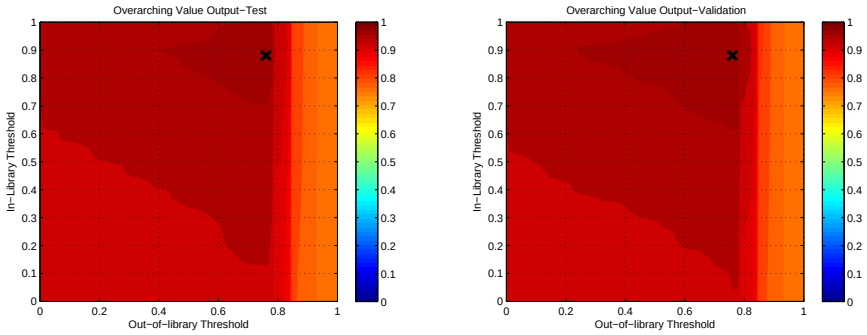

Figure 7.20 Overarching Value Contour vs IL and OOL Confidence Thresholds.

7.8 ATR Data Description

The ATR data set has been used extensively by Laine [40], Albrecht [4], and Friend [22]; as such, the process used to generate the features is given in detail in these dissertations. The ATR data set used in this research was captured from the General Dynamics Data Collection System (DCS). The ATR data set contains high-range resolution (HRR) profiles processed from synthetic aperture radar (SAR) data from two separate polarizations (HH and VV). These HRR profiles form the foundation for the feature set used in this research. The HH and VV polarizations are treated as different classifiers and are used for fusion as well.

The following describes the methodology used in evaluating the ATR data set. The data set is comprised of 724 exemplars each for 10 in-library classes (5 in-library targets and 5 in-library non-targets) and 5 out-of-library classes where each exemplar contains 10 features and each exemplar corresponds to a specific aspect angle. This data is split into 3 data sets: training set, testing set, and validation data set. Each of these data sets is comprised of 241 exemplars. At this point, each data set of 241

exemplars per class is linearly interpolated into 360 exemplars per class. Again, each exemplar corresponds to a specific aspect angle so the interpolated data set provides 1 exemplar per degree of aspect angle. The training set is used to develop the template classifier similar to that described in chapter 5. Only the data from the 10 in-library classes are used to build the template classifier. A template classifier for the ATR data set is built in the following manner. Let a denote the aspect angle where $a = 1, ..., 360$. Since aspect angle is not necessarily known with 100 % certainty, a symmetric wedge of 15 degrees is formed around the aspect angle estimate, a (i.e. $a \pm 7$ degrees). For each of the 10 in-library targets, a template is formed for every aspect angle where $\bar{x}_{ad}$ represents the average of 10 features inside the 15 degree symmetric wedge for the d^{th} class and $\hat{\Sigma}_{ad}$ is a 10 x 10 estimated diagonal covariance matrix generated from the features in a 15 degree wedge centered at aspect angle a for the d^{th} class. Once the template classifier is built using in-library targets, it is tested using two independent data sets containing both in-library and out-of-library targets.

Let x_a represent the features for an exemplar at aspect angle a. Let M_{ad} denote the Mahalanobis distance for an exemplar at aspect angle a from class d where $M_{ad} = (x_a - \bar{x}_{ad})' \hat{\Sigma}_{ad}^{-1} (x_a - \bar{x}_{ad})$.

We also leave open the option to take multiple looks similar to that performed in [40], [4], and [22]. For more than one look, the average Mahalanobis distance is found across the number of looks. Thus, if five looks are used, the Mahalanobis

distance from the current aspect angle and the next four aspect angles are averaged together. This average Mahalanobis distance is used to calculate posterior probabilities. Let M_{ad}^{e} be the Mahalanobis distance for an exemplar at aspect angle a from class d for look e where $e = 1, ..., E$ and E is the number of looks.

Let $\bar{M}_{ad}$ denote the average Mahalanobis distance across the number of looks for aspect angle a and class d. Then, $\bar{M}_{ad} = \frac{1}{E}\sum_{e=1}^{E} M_{ad}^{e}$. This $\bar{M}_{ad}$ is input into the multivariate normal distribution given by $l_{ad} = \frac{1}{(2\pi)^{f/2}|\hat{\Sigma}_{ad}|^{1/2}} e^{-\frac{1}{2}\bar{M}_{ad}}$ where f is the dimensionality of the feature space [17]. In this case, $f = 10$. In order to use Bayes rule to calculate the posterior probabilities, we need to sum across likelihoods (assuming equal prior probabilities). Let s_a represent the sum of the likelihoods across classes for aspect angle a where $s_a = \sum_{d=1}^{10} l_{ad}$. The posterior probabilities are calculated as $p_{ad} = \frac{l_{ad}}{s_a}$. For simplification, we only implement the confidence paradigm on the aggregate hostile/friendly problem. To calculate the aggregate hostile and friendly posterior probability estimates, we sum across the 5 hostile target posterior probabilities and the 5 friendly target posterior probabilities, respectively. Let p_{aT} denote the posterior probability of observing a target at aspect angle a and let p_{aN} denote the posterior probability of observing a non-target at aspect angle a. Then, $p_{aT} = \sum_{d=1}^{5} p_{ad}$, and $p_{aN} = \sum_{d=6}^{10} p_{ad}$.

We process the ATR data set slightly different from the way Friend [22] processed the data. His methodology was replicated using our processing of the data and the results are shown in Figure 7.4.

Table 7.4 Results Summary for Friend Methodology

Sensor	Looks	TPR	IL CA	OOL CA	Dec
HH	1	0.98	0.95	0.70	0.23
VV	1	0.97	0.96	0.70	0.24
HH	2	0.97	0.94	0.70	1
VV	2	0.96	0.95	0.72	1
HH	5	0.99	0.97	0.68	1
VV	5	0.97	0.96	0.72	1
HH	10	0.99	0.98	0.59	1
VV	10	0.97	0.98	0.62	1

By examining the results in Table 7.4 as well as results show in the following sections, it is evident that our paradigm produces results comparable to those produced by Friend [22].

7.9 Implementing the OOL detector on the ATR Data set

The first step in applying the GRNN OOL Detector is discretizing the bounded feature space. It is advantageous to reduce the number of features since this reduces the overall number of discrete data points that must be evaluated. Using results from Friend's research [22], the top seven features were used for discretization; Friend found features 1, 2, 3, 7, 8, 9, and 10 to be the most salient features. As a feasibility test, we chose to implement the GRNN detector at a favorable aspect angle. In past research, the template classifiers have had good performance at a 30 degree aspect angle. The feature space was discretized and Mahalanobis Distances were calculated from exemplars in a 15 degree wedge centered at 30 degrees. Figure 7.21 shows the GRNN performance as the GRNN spread value is varied. The top plot shows GRNN

performance on the validation set, and the bottom plot shows GRNN performance on the training set.

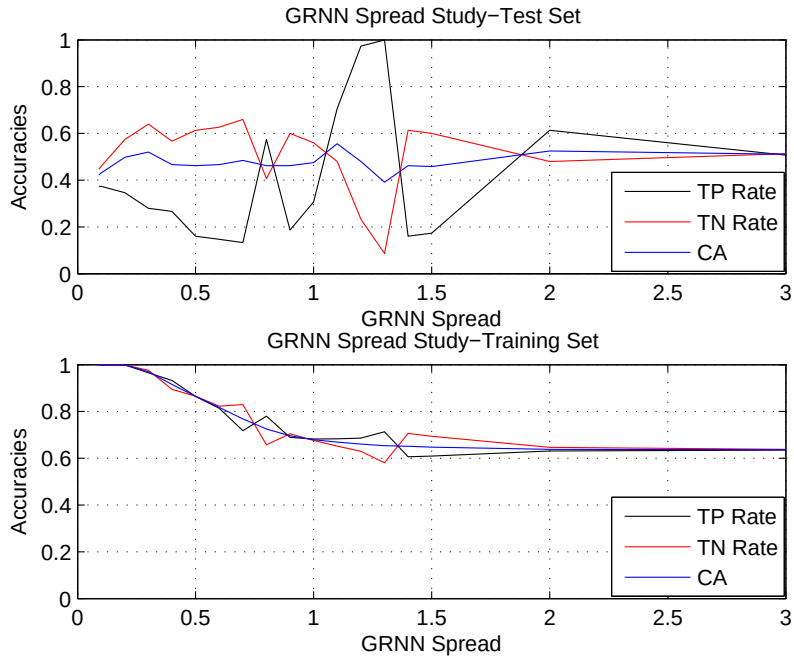

Figure 7.21 GRNN Performance vs Spread.

Figure 7.21 shows that on the training set classification accuracy, true positive rate, and true negative rate are quite high for low spread values. However, it must be noted that the training set is comprised of all discrete points, not the features from the actual ATR data set. Thus, it is predictable that the GRNN classifier performs well on the data it was trained on. Figure 7.21 shows that the classification accuracy on the validation set is approximately 50 % across all spread values. Obviously, one would like to see performance better than 50 %. This poor performance is because of the geometry of the ATR problem. There is a great deal of overlap between the IL and OOL targets; thus, it is difficult for a GRNN classifier to provide good

classification accuracy in the face of such a difficult problem. Since the GRNN OOL detector is not providing adequate performance at one of the least challenging aspect angles, we removed the GRNN OOL detector from further consideration and left it as an item of future research. Instead, we implement the confidence paradigm without an OOL detector.

7.10 ATR Data Set and Current OOL Methodology

Friend [22] develops an out-of-library procedure for a template classifier. Let the class label for exemplar a be denoted L_a where $L_a \in \{1, ..., 10\}$. For exemplar a, the class label, L_a, is determined by finding the class corresponding to the minimum average Mahalanobis distance. $L_a = \underset{d}{^{argmin}} M_{ad}$ and $M_a = \underset{d}{^{min}} M_{ad}$. By looking across all exemplars in a training set by class for only those exemplars that were correctly classified, we can determine the maximum correct average Mahalanobis distance. Friend [22] actually quantizes the correct average Mahalanobis distances, but for simplication, we will only consider the maximum. This maximum value is used as a class specific threshold, $T_d = \underset{a}{^{max}} M_{ad}$. Then, an exemplar is given out-of-library status if the minimum Mahalanobis distance corresponds to class d and $M_a > T_d$. Otherwise, it is considered an in-library exemplar and classified as one of the in-library classes or given non-declaration status.

It is apparent that according to the Friend methodology, one has supreme confidence that an exemplar is in the library if the corresponding average Mahalanobis

distance is less than the class specific threshold, and one has a complete absence of confidence that an exemplar is in the library otherwise. This is nothing more than a confidence step function. Thus, in this method, an exemplar is only observed by the IL classifier if the out-of-library system has complete confidence that the exemplar is in-library. The out-of-library confidence values as implemented in this section are binary values in the set $\{0, 1\}$.

7.11 The ATR Data Set Without an OOL Detector

Without an OOL detector, we will use the current OOL methodology and combine that with the IL confidence already developed. Just as current practices for non-declarations are imposing an implied confidence function, current out-of-library practices also impose an implied confidence function. Where current non-declaration practices typically operate on the posterior probability domain, current out-of-library practices operate on the Mahalanobis distance domain. For this case, we use the value model detailed in Chapter 6.

The posterior probabilities from all the in-library targets were used to form the IL confidence functions. The same serial processing of the data that was used with an OOL detector is used for this implementation. First, an exemplar is determined to be IL or OOL by comparing its average Mahalanobis distance to the class specific threshold. Only those exemplars that have small enough average Mahalanobis distance are considered for classification by the IL template classifier.

7.12 ATR Data Set-HH and VV Classifier Results

Figure 7.22 shows classification accuracy, non-declaration rate, and engineering confidence vs. confidence threshold for the HH and VV classifiers across 1, 2, 5, and 10 looks. Figure 7.23 shows average entropy, OOL CA, and the overarching value vs. confidence threshold for the HH and VV classifiers across 1, 2, 5, and 10 looks. The scales are different in the subplots of Figures 7.22 and 7.23 to make the subplots easier to read. It should be noted that in the research done by Friend, only OOL TP rate was recorded. OOL TP rate increased as the number of looks increased. However, further examination in this research found that this increase in OOL TP rate was at the expense of OOL TN rate. This caused an overall drop in OOL CA. To account for this trade-off, OOL CA is reported here.

These result confirm the results found by Friend [22]. There is a general increase in classification accuracy as the number of looks increase. There is also a decrease in non-declaration rate as the number of looks increase. Additionally, as expected, the engineering confidence increases as the number of looks increase. The OOL CA for 1, 2, and 5 looks are similar; however, this is a large decrease in performance for OOL CA for 10 looks. Since the features are heterogeneous across aspect angles, this suggests that there may be a point where averaging too many Mahalanobis distances (across looks) is no longer advantageous, especially with respect to OOL CA. As a result of this decrease in OOL CA for 10 looks, when examining the

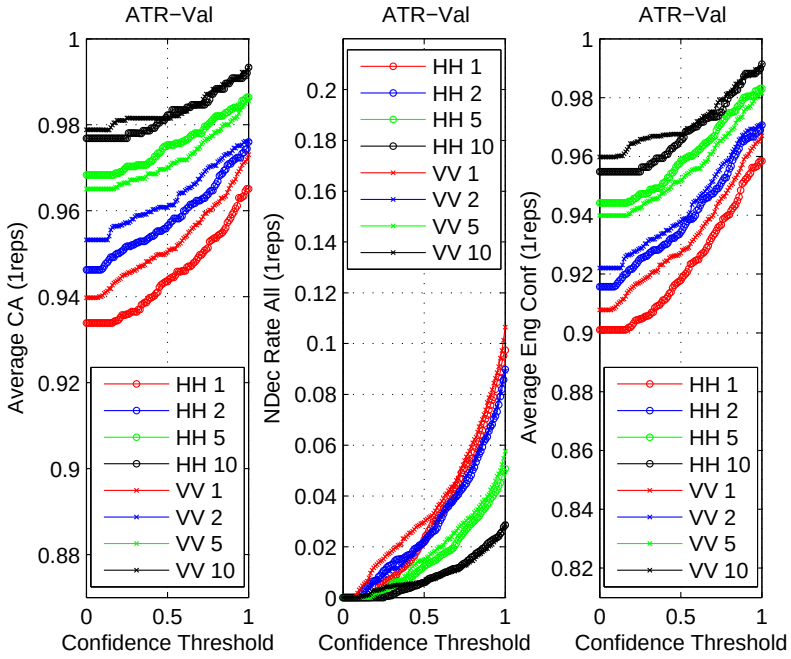

Figure 7.22 ATR-First Three Performance Parameters vs Confidence Threshold-Validation.

overarching value, 1, 2 and 5 look classifiers provide more value than the 10 look classifiers.

7.13 ATR Data Set Fusion

Fusion must be performed at two levels. First, fusion must be performed on the OOL declarations. Second, fusion must be performed on the IL declarations. For the OOL level fusion, we examine two fusion rules: the logical AND rule, and the logical OR rule. For the logical AND rule, both the HH and VV classifiers must declare a target OOL for the fusion to declare that target OOL. For the logical OR rule, if either HH or VV declares a target OOL, the fusion will declare that target

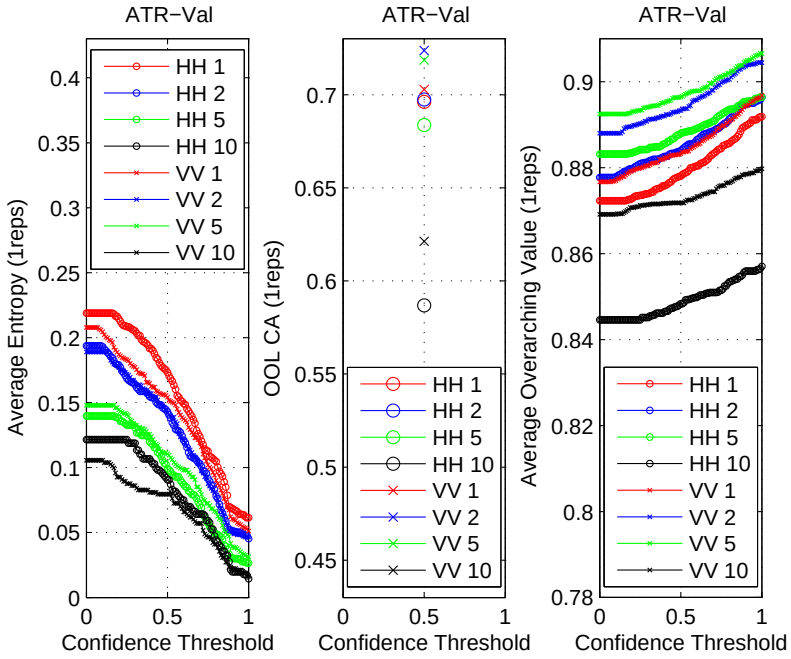

Figure 7.23 ATR-Second Three Performance Parameters vs Confidence Threshold-Validation.

OOL. Results for both of these OOL fusion rules are reported. The HH and VV IL classifiers are fused using the basic ensemble method (BEM) [50] which Friend [22] describes as "mean" fusion. BEM fusion simply averages the posterior probabilities from the individual classifier. Using these simple fusion rules, the HH and VV classifiers are fused.

7.14 HH and VV Fusion Results-OOL AND

7.14.1 HH and VV Fusion Results-1 Look-OOL AND

Figure 7.24 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 1 look as well as the BEM fusion of the two using

the OOL AND rule. Figure 7.25 shows three different performance parameters vs confidence threshold for the HH and VV classifiers with 1 look as well as the BEM fusion of the two using the OOL AND rule.

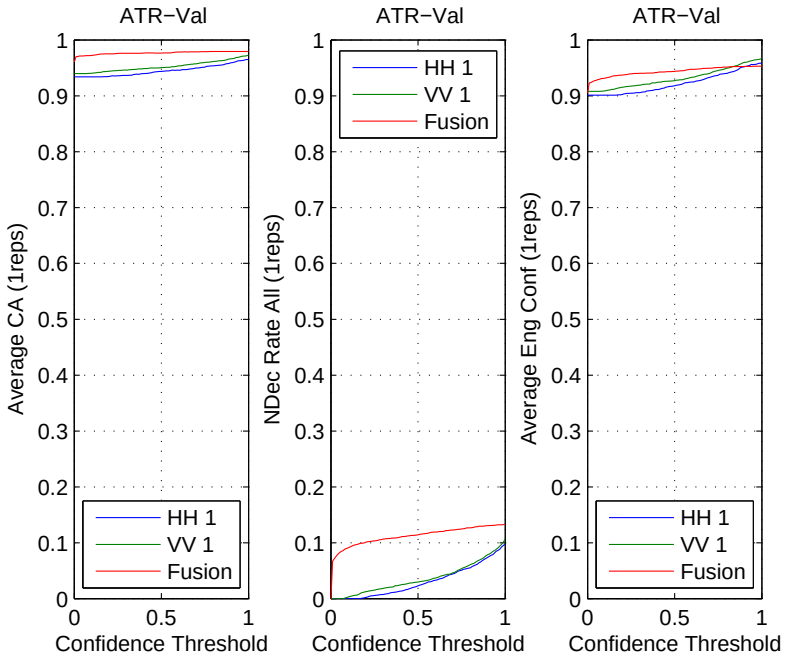

Figure 7.24 ATR-First Three Performance Parameters-1 Look-OOL AND-Validation.

As expected, the BEM fusion provides an increase in CA compared to the individual HH and VV classifiers. With a confidence threshold of 0, the BEM fusion has more entropy than the individual classifiers. Since BEM fusion simply averages the posterior probabilities, this increase in entropy is expected. However, as the confidence threshold increases slightly, the entropy drops quickly as the non-declaration increases quickly. This causes engineering confidence to increase quickly as well. This fusion provides no increase in OOL CA for the 1 look case. In terms of over-

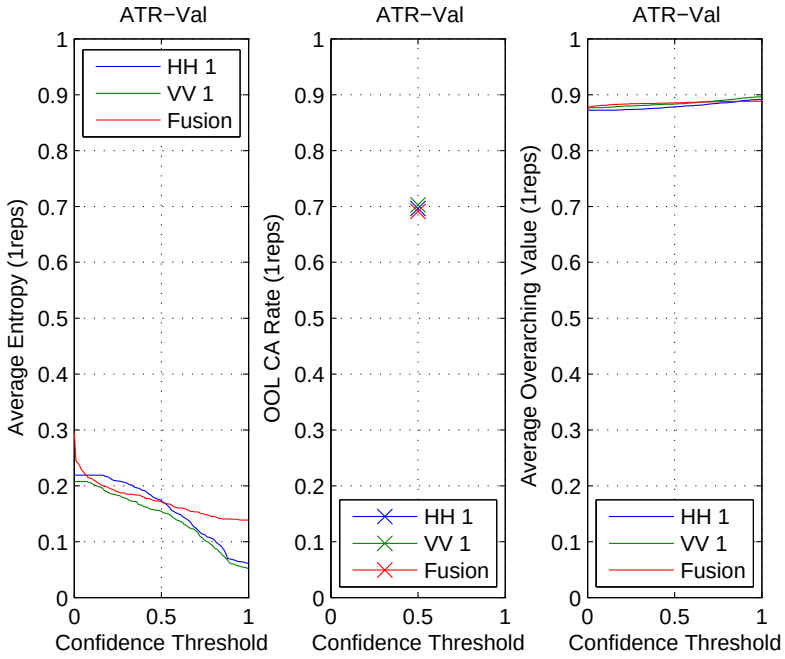

Figure 7.25 ATR-Second Three Performance Parameters-1 Look-OOL AND-Validation.

arching value, BEM fusion is preferred to the individual classifiers only slightly over most of the confidence threshold range.

7.14.2 HH and VV Fusion Results-2 Look-OOL AND

Figure 7.26 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 2 looks as well as the BEM fusion of the two using the OOL AND rule. Figure 7.27 shows three different performance parameters vs confidence threshold for the HH and VV classifiers with 2 looks as well as the BEM fusion of the two using the OOL AND rule.

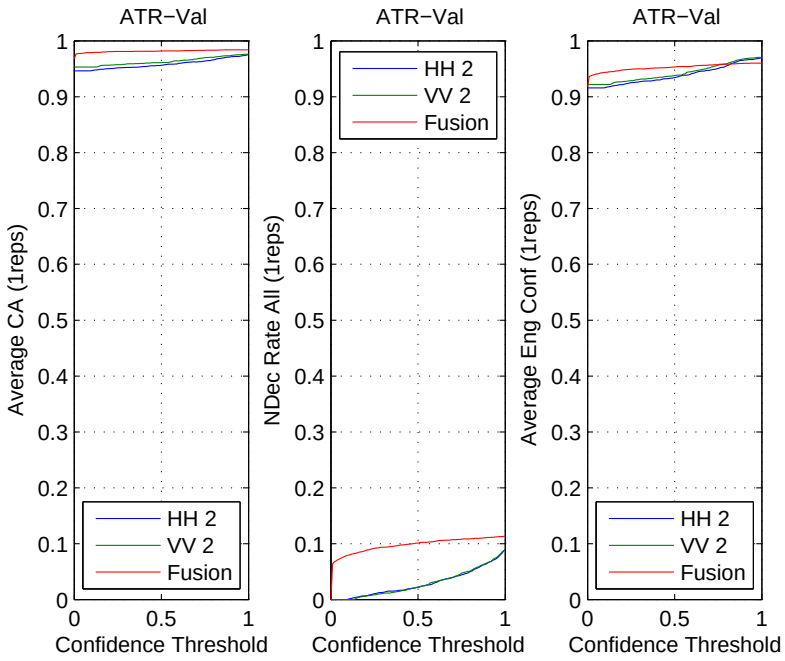

Figure 7.26 ATR-First Three Performance Parameters-2 Look-OOL AND-Validation.

The trends for BEM fusion using 2 looks are similar to the trends for BEM fusion using 1 looks. As expected, the BEM fusion provides an increase in CA compared to the individual HH and VV classifiers. With a confidence threshold of 0, the BEM fusion has more entropy than the individual classifiers. Since BEM fusion simply averages the posterior probabilities, this increase in entropy is expected. However, as the confidence threshold increases slightly, the entropy drops quickly as the non-declaration increases quickly. This causes engineering confidence to increase quickly as well. This fusion provides no increase in OOL CA for the 2 look case. In terms of overarching value, BEM fusion is preferred to the individual classifiers only slightly over most of the confidence threshold range.

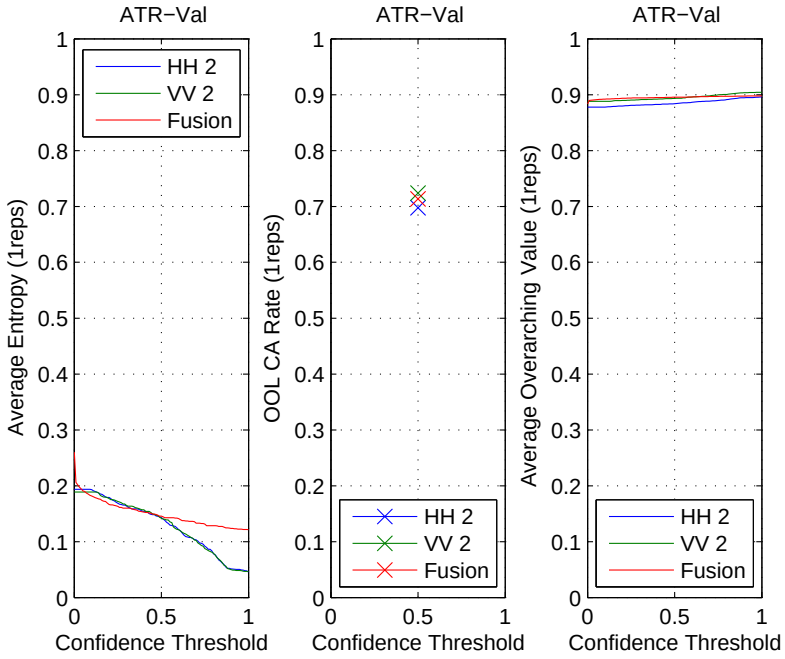

Figure 7.27 ATR-Second Three Performance Parameters-2 Look-OOL AND-Validation.

7.14.3 HH and VV Fusion Results-5 Look-OOL AND

Figure 7.28 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 5 looks as well as the BEM fusion of the two using the OOL AND rule. Figure 7.29 shows three different performance parameters vs confidence threshold for the HH and VV classifiers with 5 looks as well as the BEM fusion of the two using the OOL AND rule.

The trends for BEM fusion using 5 looks are similar the trends for BEM fusion using 2 looks. There is one change of note. For the first time, we see an increase in OOL CA due to the fusion. In terms of overarching value, the fusion is preferred over the entire confidence threshold range.

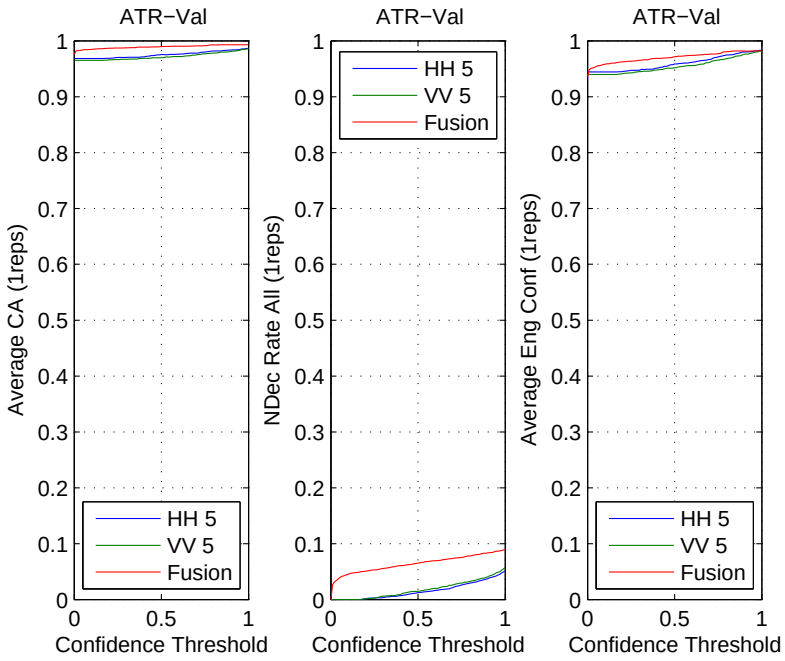

Figure 7.28 ATR-First Three Performance Parameters-5 Look-OOL AND-Validation.

7.14.4 HH and VV Fusion Results-10 Look-OOL AND

Figure 7.30 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 10 looks as well as the BEM fusion of the two using the OOL AND rule. Figure 7.31 shows three different performance parameters vs confidence threshold for the HH and VV classifiers with 10 looks as well as the BEM fusion of the two using the OOL AND rule.

The trends for the BEM fusion using 10 looks are very similar to the trends for BEM fusion using 5 looks. We see an increase in OOL CA due to the fusion. In terms of overarching value, the BEM fusion is preferred over most the entire confidence threshold range.

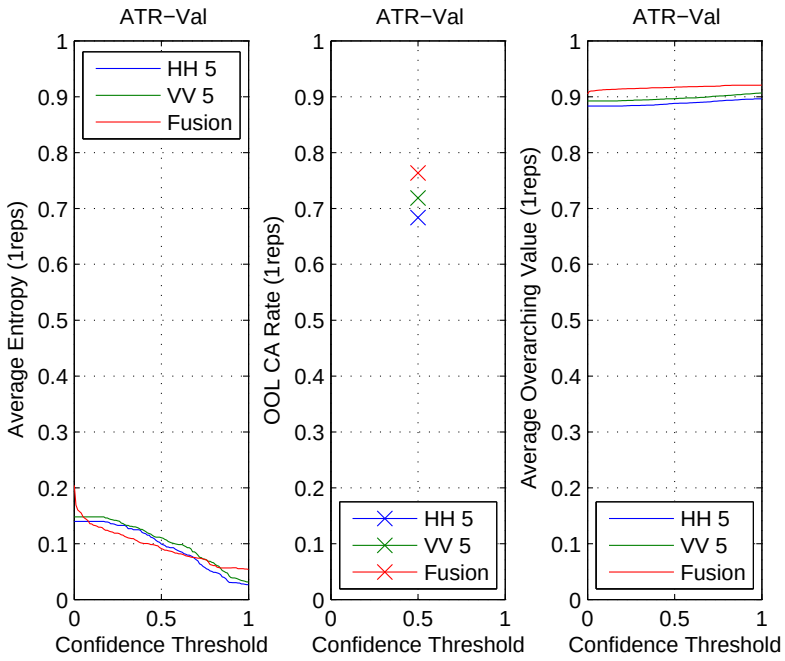

Figure 7.29 ATR-Second Three Performance Parameters-5 Look-OOL AND-Validation.

7.14.5 HH and VV Fusion Results-OOL AND

When using the OOL AND rule, we can make some general observations across the number of looks. First, using overarching value, we always prefer the fusion to the individual classifiers. However, we do not always prefer an increasing number of looks. In fact, the overarching value peaks at 5 looks. This is another indication that averaging over too many looks in this data set can cause degradation in performance.

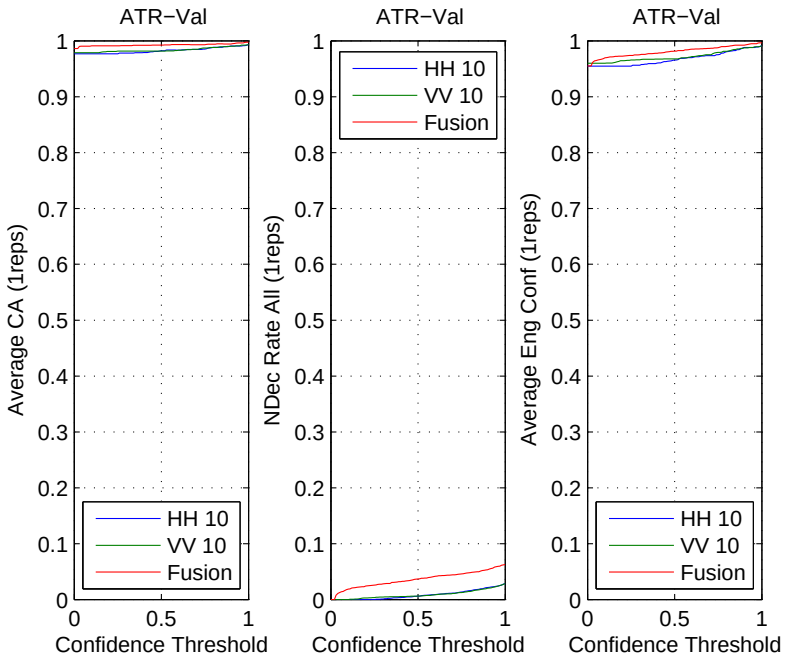

Figure 7.30 ATR-First Three Performance Parameters-10 Look-OOL AND-Validation.

7.15 HH and VV Fusion Results-OOL OR

7.15.1 HH and VV Fusion Results-1 Look-OOL OR

Figure 7.32 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 1 look as well as the BEM fusion of the two using the OOL OR rule. Figure 7.33 shows three different performance parameters vs confidence threshold for the HH and VV classifiers with 1 look as well as the BEM fusion of the two using the OOL OR rule.

BEM fusion performance using the OOL OR is very similar to BEM fusion performance using the OOL AND. The only significant difference is that the OOL CA for the fusion marginally outperforms the individual classifiers. Because of this,

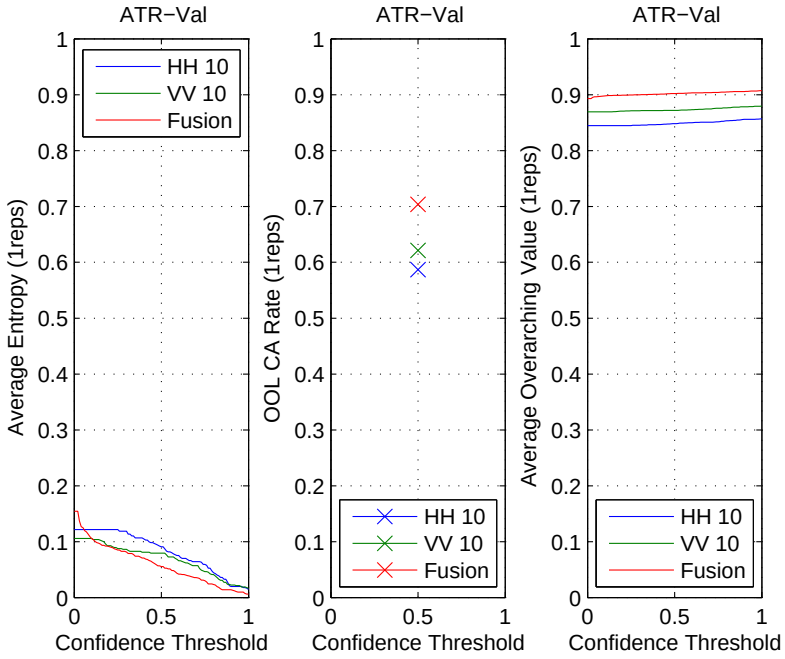

Figure 7.31 ATR-Second Three Performance Parameters-10 Look-OOL AND-Validation.

in terms of overarching value, the BEM fusion is preferred to the individual classifiers across the confidence threshold space.

7.15.2 HH and VV Fusion Results-2 Look-OOL OR

Figure 7.34 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 2 looks as well as the BEM fusion of the two using the OOL OR rule. Figure 7.35 shows three different performance parameters vs confidence threshold for the HH and VV classifiers with 2 looks as well as the BEM fusion of the two using the OOL OR rule.

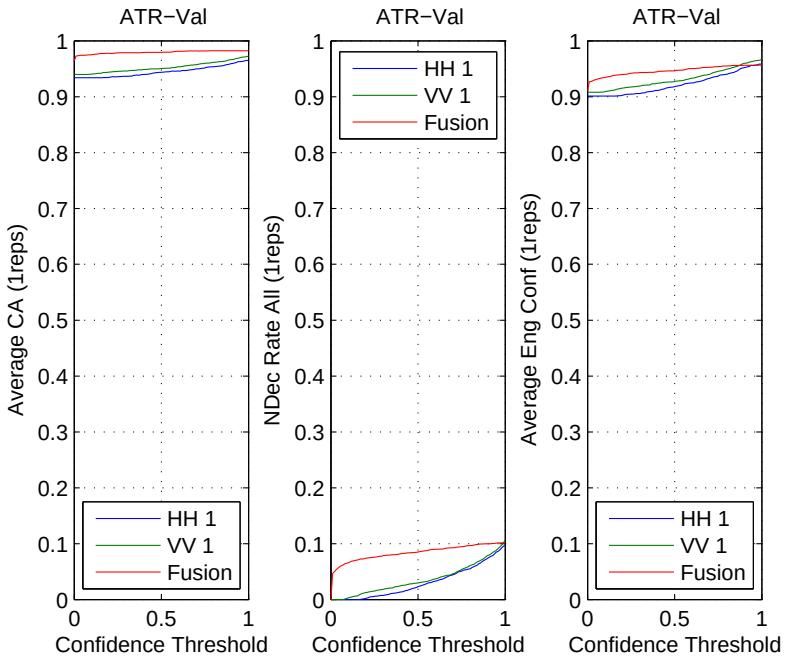

Figure 7.32 ATR-First Three Performance Parameters-1 Look-OOL OR-Validation.

The performance using 2 looks and the OOL OR is very similar to the performance using 1 look and the OOL OR. One difference is that the OOL CA for the fusion is between the OOL CA for the individual classifiers. Because of this, in terms of overarching value, the BEM fusion performs nearly identically to the best classifier.

7.15.3 HH and VV Fusion Results-5 Look-OOL OR

Figure 7.36 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 5 looks as well as the BEM fusion of the two using the OOL OR rule. Figure 7.37 shows three different performance parameters vs

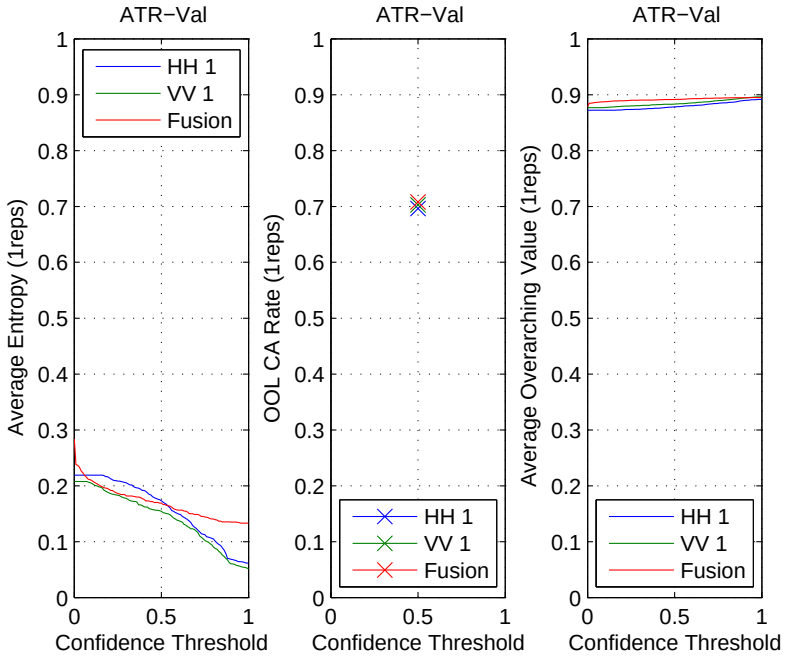

Figure 7.33 ATR-Second Three Performance Parameters-1 Look-OOL OR-Validation.

confidence threshold for the HH and VV classifiers with 5 looks as well as the BEM fusion of the two using the OOL OR rule.

The performance of the BEM fusion using the OOL OR continues to decrease as we increase to 5 looks. The OOL CA for the BEM fusion is less than the OOL CA for the individual classifiers. In terms of overarching value, the BEM fusion using 5 looks is never preferred to the individual classifiers.

7.15.4 HH and VV Fusion Results-10 Look-OOL OR

Figure 7.38 shows three performance parameters vs confidence threshold for the HH and VV classifiers with 10 looks as well as the BEM fusion of the two using

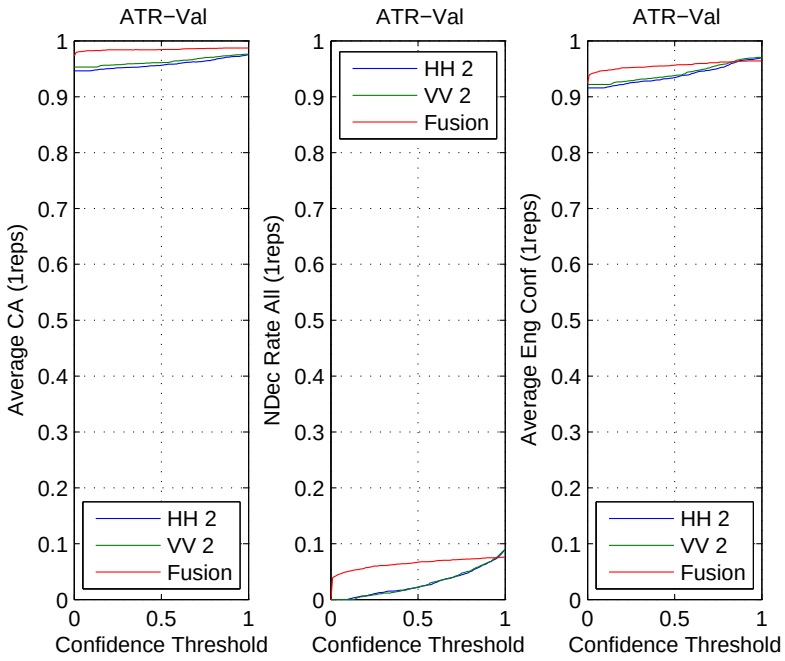

Figure 7.34 ATR-First Three Performance Parameters-2 Look-OOL OR-Validation.

the OOL OR rule. Figure 7.39 shows three different performance parameters vs confidence threshold for the HH and VV classifiers with 10 looks as well as the BEM fusion of the two using the OOL OR rule.

The performance of the BEM fusion using the OOL OR decreases as we increase to 10 looks. The contributing factor here is that the OOL CA for the BEM fusion using 10 looks is less than the OOL CA for the individual classifiers. Again, in terms of overarching value, the BEM fusion using 10 looks is never preferred to the individual classifiers.

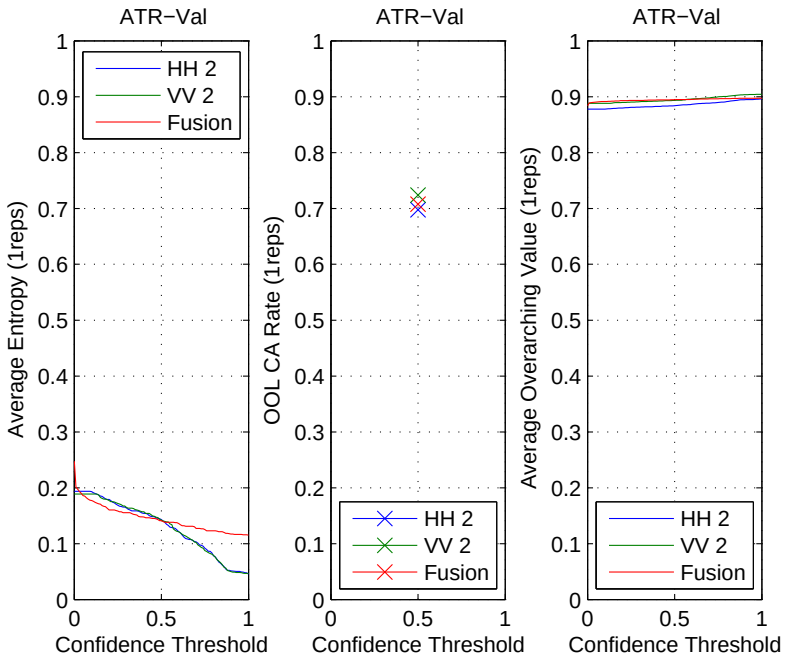

Figure 7.35 ATR-Second Three Performance Parameters-2 Look-OOL OR-Validation.

7.15.5 HH and VV Fusion Results-OOL OR

When using the OOL OR rule, we can make some general observations across the number of looks. As the number of looks increases, there is a continuing decrease in OOL CA for the BEM fusion as the number of looks increases. Since the OOL OR rule declares an exemplar OOL when either the HH or VV classifier declares an exemplar OOL, there are more OOL declarations. We can see by the OOL CA, more error is introduced to the OOL process using the OR rule especially for a larger number of looks.

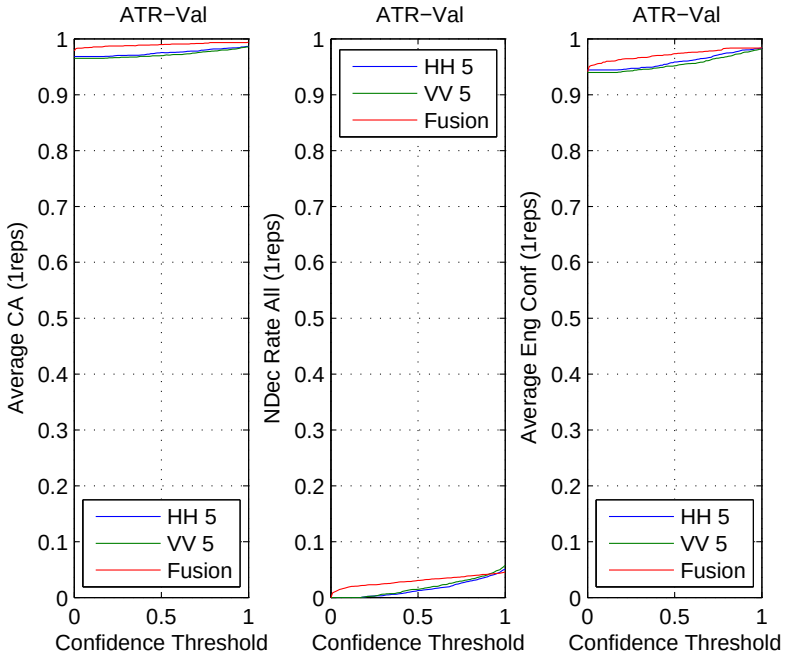

Figure 7.36 ATR-First Three Performance Parameters-5 Look-OOL OR-Validation.

7.15.6 HH and VV Fusion Results Summary

We examined HH and VV classifier results as well as BEM fusion of the HH and VV classifiers. In addition, we looked at two ways to combine OOL declarations from the two classifiers. Each individual performance parameter performed consistent with a rational confidence paradigm. That is, classification accuracy, engineering confidence, and non-declarations rates increased as the confidence threshold increased, and average entropy decreased as the confidence threshold increased. We did observe some interesting observations between the number of looks and the OOL fusion rule. First, we saw that for both the AND and OR OOL fusion rule, performance did not always increase as the number of looks increased. This is because each

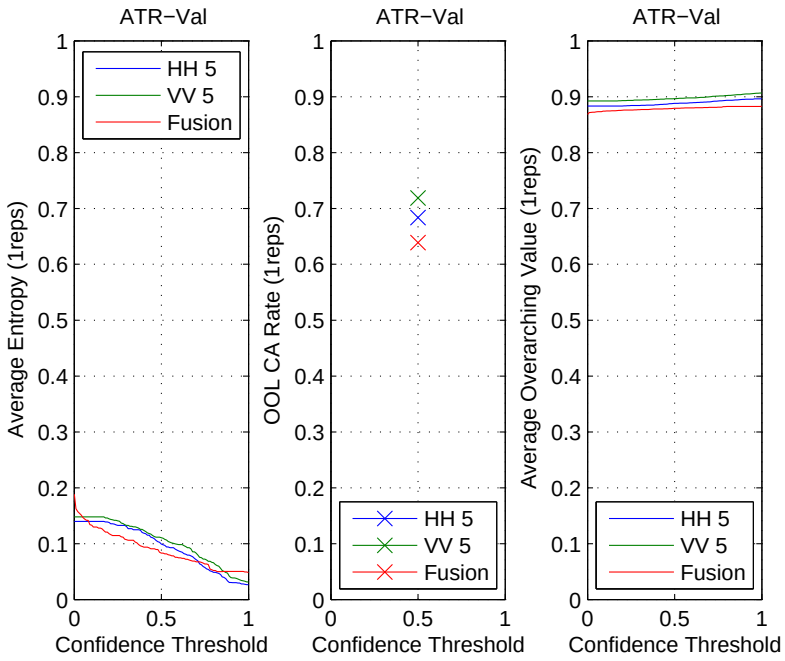

Figure 7.37 ATR-Second Three Performance Parameters-5 Look-OOL OR-Validation.

exemplar corresponds to a specific aspect angle, and the features are heterogeneous across the aspect angles. The features become more and more heterogeneous as the number of looks increases so we see a decrease in performance for too many looks. We saw optimal performance at 5 looks for the AND rule and 2 looks for the OR rule. In general, the AND rule provided no decrease in OOL CA when compared to the individual classifiers. In general, the OR rule provided no increase in OOL CA when compared to the individual classifiers.

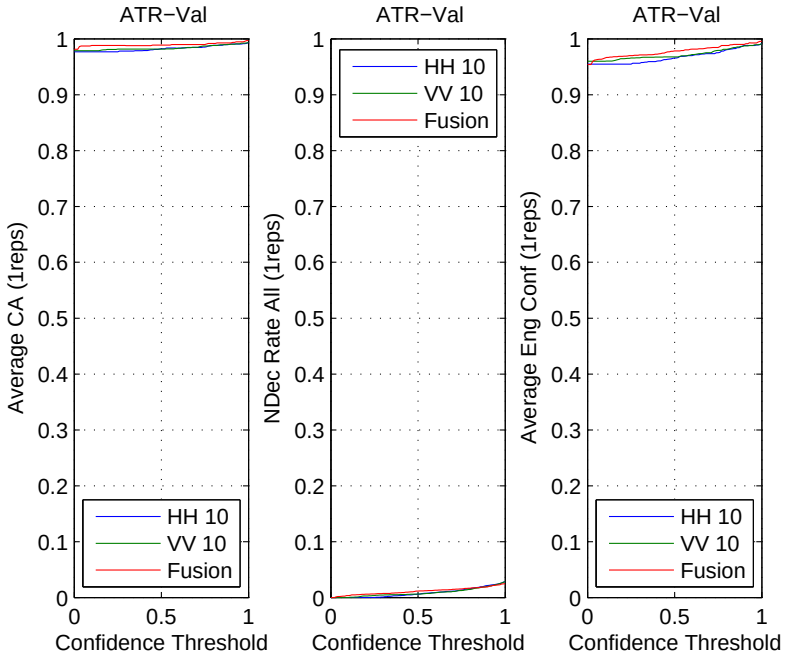

Figure 7.38 ATR-First Three Performance Parameters-10 Look-OOL OR-Validation.

7.16 Results Summary

This chapter provided results for all the methodologies described in Chapters 4, 5, and 6. Four different data sets were described: Cancer data set, Diabetes data set, MVN data set, and ATR data set. Results are provided on three two-class problems without out-of-library (OOL) targets: Cancer, Diabetes, and MVN. Results are provided on a MVN problem with OOL targets when an OOL detector is used. GRNN OOL detector results on the ATR data set are provided. Finally, results are provided on the ATR data set with OOL targets when an OOL detector is not available. In all cases, results are consistent with a rational confidence paradigm.

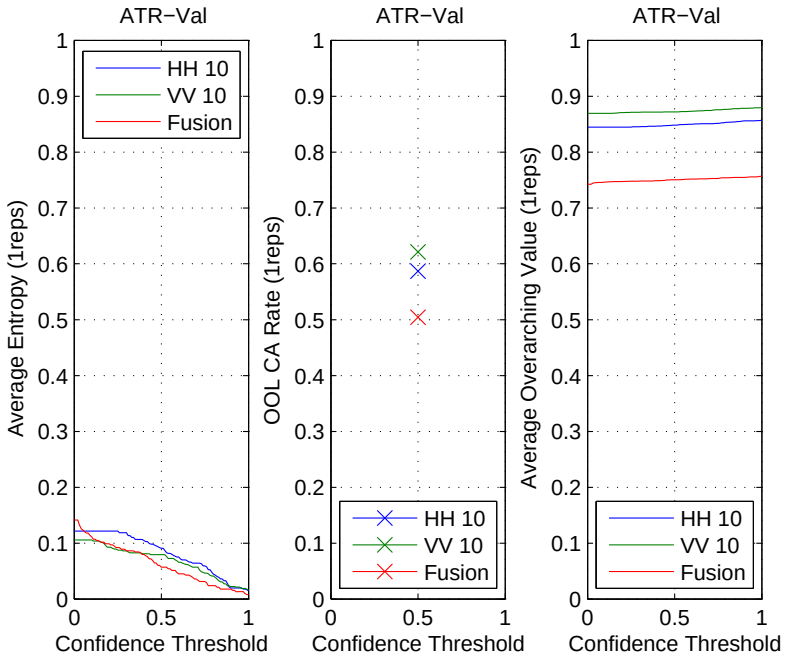

Figure 7.39 ATR-Second Three Performance Parameters-10 Look-OOL OR-Validation.

8. Contributions and Future Work

This chapter provides a summary of the contributions made to the fields of pattern recognition and automatic target recognition through the research of this dissertation. In addition, areas of future work related to this research effort are presented.

8.1 Contributions

In this research, we developed a confidence paradigm for classification systems. As such, this research makes several contributions to the fields of pattern recognition and automatic target recognition. A brief summary of each contribution is provided.

8.1.1 Confidence as a Function of Posterior Probabilities

We provide a gradient descent algorithm that minimizes the Binned Error in the Posteriors (BEP) as defined by Ross and Minardi [53]. This research proves that training a multiple layered perceptron to minimize sum of squared error also minimizes BEP. We also show that in minimizing BEP, the outputs of the neural network are posterior probability estimates; hence, the best confidence score is the posterior probability estimate. However, BEP does not consider the desired posterior probability distributions (i.e., distributions that lead to separation between the classes). Thus, confidence is modelled as a function of the posterior probabilities.

8.1.2 Confidence Paradigm

This research develops a confidence paradigm that encompasses and generalizes current practices under the assumption that system confidence acts like a value and that indication confidence can be modelled as a function of the posterior probability estimates. One output of this paradigm is a new confidence measure that unites traditional decision analysis techniques and current pattern recognition techniques. The main axiom of this contribution is as follows: the average confidence in the output of a classifier, by class, is approximately equal to the confidence in the classifier as it operates on that class.

8.1.3 Tactical Issues of Fitting the Confidence Function

This research develops a methodology to determine the parameters of a generalized confidence function within the confidence paradigm. A methodology is developed to find both the vertex and multiplier for the quadratic confidence function.

8.1.4 Stochastic Non-Declaration Procedure

One novel application of the generalized confidence function is a new non-declaration methodology using a stochastic implementation. This involves interpreting the confidence function as the probability of a non-declaration given a posterior probability, $P(DEC|p)$, and is implemented by comparing the output of the confidence function to a random number draw.

8.1.5 New Use of the Kullback-Liebler Distance

This research develops a new use of the popular Kullback-Liebler (KL) distance to determine how well a classifier generalizes to an independent data set. In this application, we measure the KL distance between the histogram of the training set and the histogram of the independent test set. The KL distance between these two is a measure of how well the classifier is generalizing to an independent data set.

8.1.6 New Out-of-Library Detector

We develop a new methodology to determine out-of-library (OOL) targets by bounding and discretizing the feature space, designating those discrete points as either in-library (IL) or OOL, and providing those discrete features to a generalized regression neural network. The effectiveness of this OOL detector is demonstrated on a synthetic example.

8.1.7 Out-of-Library Non-Declarations

Typically, non-declarations are considered in an IL target setting. The confidence paradigm is applied to an out-of-library problem which leads to a demonstration of the new concept of OOL non-declarations. This OOL non-declaration region of the features space is combined with the typical IL non-declaration region of the feature space to form a combined non-declaration region of the feature space.

8.1.8 Overarching Value Model

This research unites multiple performance measures, specifically, the engineering confidence values and non-declaration rates, under an overarching value model that leads to choosing the optimal confidence threshold. Overarching value models are developed for cases considering IL and OOL target types.

8.2 Future Work

There were other research areas that could have been investigated, but due to time constraints, these were left as items of future research. This section details the items of future work.

8.2.1 Different Forms of the Confidence Function

The only confidence function form investigated in this research is the quadratic confidence function. A sigmoid may also be appropriate and is left as an item of future work. In changing forms, new tactical issues of how to find the appropriate parameters will need to be addressed.

8.2.2 Engineering Confidence and Expectation of Random Variables

This research sets the expected value of the confidence function equal to the output of an engineering confidence model, C^{ENG}. Some of the inputs to the engineering confidence model are random variables (i.e., classification accuracy and

average entropy). Rather than set the expected value of the confidence function equal to C^{ENG}, it could be set equal to $E[C^{ENG}]$.

8.2.3 Finding Function Parameters in Parallel

We parameterized the confidence function by first fixing the vertex and then solving an optimization problem conditioned on the fixed vertex. The optimization problem can also be solved by considering both of the parameters at the same time. However, this will lead to multiple optimal solutions. Thus, some additional constraint or condition must be enforced to get to a single solution. Performance across the confidence threshold could be studied across the set of multiple optimal parameters.

8.2.4 Expansion of the Engineering Confidence Model

While we model engineering confidence as a function of three attributes, there may be other considerations that could be incorporated. Robustness and context are two attributes that could be possible additions to the engineering confidence model.

8.2.5 Generalization of the Confidence Paradigm

Classification systems can have a variety of outputs. Our paradigm was developed to operate on posterior probability estimates. The paradigm can be generalized to operate on any output from a classification system. Of course, if the output of the classification system is not bounded between 0 and 1, the expected value of the

output is not bounded between 0 and 1. This poses a problem since the output of the engineering confidence value is designed to be a value between 0 and 1. This issue will need to be resolved to generalize the confidence paradigm. In addition, the confidence paradigm was only applied to two-class problems. Application of the paradigm to problems with more than two classes is left as an item of future work.

8.2.6 Generalized Regression Neural Network OOL Detector

We successfully demonstrated the effectiveness of the OOL detector on a synthetic example. However, the GRNN OOL detector did not perform as favorably on the ATR data set. Improving this methodology on the ATR data set is something that can be done as an item of future work.

8.2.7 Improving Implementation of Multiple Looks in ATR Data Set

Currently, we implement multiple looks by calculating a Mahalanobis distance for consecutive aspect angles and average the Mahalanobis distances across the number of looks. These average Mahalanobis distances are used to find posterior probability estimates. This process can be improved and should be examined as an item of future work.

8.2.8 Expansion of the Overarching Value Model

While we modelled overarching value as a function of engineering confidence and non-declaration rates, attributes such as critical and non-critical error rates could also be added to the model.

Bibliography

1. (2006). *The Holy Bible: New International Version*. Zondervan, Grand Rapids, MI.

2. (2006). Webster's online dictionary. http://www.websters-online-dictionary.org.

3. (2006). Wikipedia online. http://www.wikipedia.org.

4. Albrecht, T. W. (2005). *Combat Identification with Sequential Observations, Rejection Option, and Out-of-Library Targets*. Ph.D. thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH.

5. Arenas-Garcia, J., Gomez-Verdejo, V., Munoz-Romero, S., Ortega-Moral, M., and Figueiras-Vidal, A. (2005). Fast classification with neural networks via confidence rating. In *8th International Work-Conference on Artificial Neural Networks*, volume 3512. 622–629.

6. Arribas, J. I. and Cid-Sueiro, J. (2005). A model selection algorithm for a posterior probability estimation with neural networks. *IEEE Transactions on Neural Networks*, **16**(4).

7. Atukorale, A. S. and Suganthan, P. N. (1999). Combining classifiers based on confidence values. In *Proceedings of the Fifth International Conference on Document Analysis and Recognition*, volume 09. 37–40.

8. Bailey, T. C., Everson, R. M., Fieldsend, J. E., Krzanowski, W. J., Partridge, D., and Schetinin, V. (2007). Representing classifier confidence in the safety critical domain: an illustration from mortality prediction in trauma case. *Neural Computing and Applications*, **16**(1).

9. Baraghimian, G. (1994). Heuristics for text recognition using contextual information. In *Image and Information Systems, Proceedings of SPIE*, volume 2368.

10. Bassham, B., Jr., K. W. B., and Miller, J. O. (2006). Automatic target recognition system evaluation using decision analysis techniques. *MORS*, **11**(1).

11. Black, M. and Franklin, M. (2004). Perceptron-based confidence estimation for value prediction. *IEEE*.

12. Callari, F. G. and Ferrie, F. P. (1996). Active recognition: Using uncertainty to reduce ambiguity. *IEEE*.

13. Daugman, J. (2001). High confidence recognition of persons by iris patterns. *IEEE*.

14. Daugman, J. G. (1993). High confidence visual recognition of persons by a test of statistical independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**(11).

15. Delany, S. J., Cunningham, P., and Doyle, D. (2005). Generating estimates of classification confidence for a case-based spam filter. *Lecture Notes in Artificial Intelligence*, **3620**, 177–190.

16. Do, T. D., Hui, S. C., and Fong, A. C. (2006). Prediction confidence for associative classification. In *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, IEEE*, volume 3930.

17. Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification*. John Wiley and Sons, New York, NY, second edition.

18. Dyer, J. S. and Sarin, R. K. (1979). Measurable multiattribute value functions. *Operations Research*, **27**, 810–822.

19. Everson, R. M. and Fieldsend, J. E. (2006). Multi-class roc analysis from a multi-objective optimisation perspective. *Pattern Recognition Letters*, **27**, 918–927.

20. Fenton, N. and Wang, W. (2006). Risk and confidence analysis for fuzzy multicriteria decision making. *Knowledge-Based Systems*, **19**.

21. Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the Accoustical Society of America*, **35**.

22. Friend, M. A. (2007). *Combat Identification with Synthetic Aperture Radar, Out-of-Library Identification, and Non-Declarations*. Ph.D. thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH.

23. Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2000). *Baysian Data Analysis*. Chapman and Hall/CRC Press, Boca Raton, Florida.

24. Ghosh, A., Verma, A., and Sarkar, A. (2001). Using likelihood l-statistics to measure confidence in audio-visual speech recognition. In *IEEE Fourth Workshop on Multimedia Signal Processing*. 27–32.

25. Grunwald, D., Klauser, A., Manne, S., and Pleszkum, A. (1998). Confidence estimation for speculation control. *SIGARCH Comput. Archit. News*, **26**(3).

26. Haspert, J. K. (2000). Optimum id sensor fusion for multiple target types. Technical Report D-2451, Institute for Defense Analysis (IDA).

27. Ho, T. (1992). *A Theory of Multiple Classifier Systems and Its Application to Visual Word Recognition*. Ph.D. thesis, State University of New York at Buffalo.

28. Ho, T. K., Hull, J. J., and Srihari, S. N. (1992). On multiple classifier systems for pattern recognition. In *Proceedings, 11th IAPR International Conference on Pattern Recognition Methodology and Systems*, volume 2. 84–87.

29. Huang, S., Xie, X., and Kuang, J. (2007). Novel method to combine phone-level confidence scores using support vector machines. In *IEEE Proceedings of ICSP*.

30. Huang, Y. S. and Suen, C. Y. (1993). Combination of multiple classifiers with measurement values. *IEEE*.

31. Jaeger, S. (2004). Using informational confidence values for classifier combination: An experiment with combined on-line/off-line Japanese character recognition. In *Proceedings of the 9th International Workshop on Frontiers in Handwriting Recognition, IEEE*.

32. Johnson, R. A. and Wichern, D. W. (2002). *Applied Multivariate Statistical Analysis*. Prentice Hall, Upper Saddle River, NJ, fifth edition.

33. Kanungo, T. and Haralick, R. M. (1995). Receiver operating characteristic curves and optimal bayesian operating points. In *Proceedings of the International Conference on Image Processing, IEEE*. 256–259.

34. Keeney, R. L. and Raiffa, H. (1999). *Decisions with Multiple Objectives*. Cambridge University Press, Cambridge, United Kingdom.

35. Kim, T.-Y. and Ko, H. (2004). Combining of confidence measures under bayesian framework for speech recognition. In *Proceedings of International Symposium on Intelligent Signal Processing and Communication Systems, IEEE*.

36. Kimball, R. and Rothkopf, M. H. (1976). Utterance classification confidence in automatic speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processsing*, **24**, 188–190.

37. Kingshy Goh, B. L. and Chang, E. Y. (2005). Semantics and feature discovery confidence-based ensemble. *ACM Transactions on Multimedia Computing, Communications and Applications*, **1**(2), 168–189.

38. Kirkwood, C. W. (1997). *Strategic Decision Making*. Duxbury Press, Belmont, California.

39. Krzanowski, W. J., Fieldsend, J. E., Bailey, T. C., Everson, R. M., Partridge, D., and Schetinin, V. (2006). Confidence in classification: A Bayesian approach. *Journal of Classification*, **23**(1), 199–220.

40. Laine, T. I. (2005). *Optimization of Automatic Target Recognition With a Reject Option Using Fusion and Correlated Sensor Data*. Ph.D. thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH.

41. Lane, I., Kawahara, T., Matsui, T., and Nakamura, S. (2006). Out-of-domain utterance detection using classification confidences of multiple topics. *IEEE Transactions on Audio, Speech, and Language Processing*.

42. Law, A. M. and Kelton, W. D. (2000). *Simulation Modeling and Analysis*. McGraw Hill, Boston, MA, third edition.

43. Leap, N., Clemans, P., Kenneth W. Bauer, J., and Oxley, M. (2008). An investigation of the effects of correlation and autocorrelation in classifier fusion. *International Journal of General Systems-Intelligent Systems Design*, **37**(4), 475–498.

44. Li, M. and Sethi, I. K. (2006). Confidence-based classifier design. *Pattern Recognition*, **39**.

45. Looney, C. G. (1997). *Pattern Recognition Using Neural Networks*. Oxford University, New York, NY.

46. Melnik, O., Vardi, Y., and Zhang, C.-H. (2004). Mixed group ranks: Preference and confidence in classifier combination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**(8).

47. Nash, S. G. and Sofer, A. (1996). *Linear and Nonlinear Programming*. McGraw Hill, New York, NY.

48. Newman, D., Hettich, S., Blake, C., and Merz, C. (1998). UCI repository of machine learning databases. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

49. Parker, D. R., Gustafson, S. C., and Ross, T. (2005). Receiver operating characteristic and confidence error metrics for assessing the performance of automatic target recognition systems. *Optical Engineering*, **44**(9).

50. Perrone, M. P. and Cooper, L. N. (1993). When networks disagree: Ensemble methods for hybrid neural networks. *In Neural Networks for Speech and Image Processing*.

51. Richards, J. A., Jr., W. J. B., and Bray, B. K. (2003). An informative confidence metric for ATR. In *Proceedings of SPIE: Algorithms for Synthetic Aperture Radar Imagery X*, volume 5095.

52. Roberts, S. J. (1996). Assessing the confidence of classification in artificial neural networks. In *Proceedings of the 1996 IEE Colloquium on Artificial Intelligence Methods for Biomedical Data Processing*, volume 100.

53. Ross, T. and Minardi, M. E. (2004). Discrimination and confidence error in detector reported scores. In *Algorithms for Synthetic Aperture Radar Imagery XI, Proceeding of SPIE*, volume 5427.

54. Ross, T., Westerkamp, L., Dilsavor, R., and Mossing, J. (2002). Performance measures for summarizing confusion matrices–the AFRL COMPASE approach. In *Proceedings of SPIE*, volume 4727. 310–321.

55. Ruck, D. W., Rogers, S. K., Kabrisky, M., Oxley, M. E., and Suter, B. W. (1990). The multilayer perceptron as an approximation to the Bayes optimal

discriminant function. *IEEE Transactions on Neural Networks*, **TNN-1**(4), 296–298.

56. Sadowski, C. (2005). Combat identification: Where are we? AFIT OPER 596 Guest Lecture.

57. Sadowski, C. (2008). Combat identification. AFIT OPER 786 Guest Lecture.

58. Schmeiser, B. and Yeh, Y. (2002). On choosing a single criterion for confidence-interval procedures. In *Proceedings of the 2002 Winter Simulation Conference*.

59. Schruben, L. W. (1980). A coverage function for interval estimators of simulation response. *Management Science*, **26**(1).

60. Simona Gandrabur, G. F. and Lapalme, G. (2006). Confidence estimation for nlp applications. *ACM Transactions on Speech and Language Processing*, **3**(3), 1–29.

61. Thomas, I. L. and Allcock, G. M. (1984). Determining the confidence level for a classification. *Photogrammetric Engineering and Remote Sensing*, **50**, 1491–1496.

62. Tubbs, J. D. and Alltop, W. O. (1991). Measures of confidence associated with combining classification results. *IEEE Transactions on Systems, Man, and Cybernetics*, **21**(3).

63. United States Air Force (1998). *Air Force Doctrine Document 2-1.2: Strategic Attack, AFDD 2-1.2*.

64. Wackerly, D. D., III, W. M., and Scheaffer, R. L. (2002). *Mathematical Statistics with Applications*. Duxbury Press, Pacific Grove, CA, sixth edition.

65. Wan, E. A. (1990). Neural network classification: A Bayesian interpretation. *IEEE Transactions on Neural Networks*, **1**(4).

66. Weintraub, M., Beaufays, F., Rivlin, Z., Konig, Y., and Stolcke, A. (1997). Neural-network based measures of confidence for word recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97)*, volume 2.

67. Wise, A. R., Fitzgerald, D., and Ross, T. D. (2004). The adaptive sar atr problem set (adaptsaps). In *Algorithms for Synthetic Aperture Radar Imagery XI, Proceeding of SPIE*, volume 5427.

68. Zhang, X., Gupta, N., and Gupta, R. (2006). Pruning dynamic slices with confidence. *ACM SIGPLAN Notices*, **41**(6).

Vita

Captain Nathan J. Leap graduated from Bishop McCort High School in Johnstown, Pennsylvania. He entered undergraduate studies at the United States Air Force Academy (USAFA) where he graduated with a Bachelor of Science Degree in Operations Research in June 1999. He was commissioned through USAFA with a Reserve Commission.

His first assignment was at Eglin AFB as an operations research analyst for the 28th Test Squadron, 53rd Test and Evaluation Group, 53rd Wing. In August 2003, he entered the Graduate School of Engineering and Management, Air Force Institute of Technology, Wright-Patterson AFB, Ohio. Upon graduation, he was assigned to HQ Air Force Materiel Command, Management Sciences Division, Wright-Patterson AFB, Ohio, as an operations research analyst. In September 2005, he entered the Graduate School of Engineering and Management, Air Force Institute of Technology. Upon graduation, he will be assigned to HQ Air Force, Studies & Analyses, Assessments, and Lessons Learned, Pentagon, Washington, DC.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
8-5-2008	**Doctoral Dissertation**	Sept 2005 – Aug 2008

4. TITLE AND SUBTITLE

A Confidence Paradigm For Classification Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Leap, Nathan J., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Street, Building 642
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/DS/ENS/08-02

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NM
Suite 325 Rm 3112
875 Randolph
Arlington VA 22203-1768

ACC/DRSA
216 Hunting Ave, Rm 105
Langley AFB VA 23665-2777

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR/NM, ACC/DRSA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

There is no universally accepted methodology to determine how much confidence one should have in a classifier output. This research proposes a framework to determine the level of confidence in an indication from a classifier system where the output is or can be transformed into a posterior probability estimate. This is a theoretical framework that attempts to unite the viewpoints of the classification system developer (or engineer) and the classification system user (or war-fighter). The paradigm is based on the assumptions that the system confidence acts like, or can be modeled as a value and that indication confidence can be modeled as a function of the posterior probability estimates.

The introduction of the non-declaration possibility induces the production of a higher-level value model that weighs the contribution of engineering confidence and associated non-declaration rate. Now, the task becomes to choose the appropriate threshold to maximize this overarching value function. This paradigm is developed in a setting considering only in-library problems, but it is applied to out-of-library problems as well. Introduction of out-of-library problems requires expansion of the overarching value model. This confidence measure is a direct link between traditional decision analysis techniques and traditional pattern recognition techniques. This methodology is applied to multiple data sets, and experimental results show the behavior that would be expected from a rational confidence paradigm.

15. SUBJECT TERMS

Confidence, classification, automatic target recognition, multiattribute preference theory, out-of-library

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**Dr. Kenneth Bauer, Jr. (AFIT/ENS)**
U	**U**	**U**	**UU**	180	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-6565, ext 4328