Proc. International Joint Conference on Neural Networks 2006, Vancuver, Canada, July 2006

Data Fusion for Outlier Detection through Pseudo-ROC Curves and Rank Distributions

Paul F. Evangelista
Department of Systems Engineering
United States Military Academy
West Point, NY 10996
E-mail: paul.evangelista@usma.edu

Mark J. Embrechts
Department of Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute
Troy, NY 12180
E-mail: embrem@rpi.edu

Boleslaw K. Szymanski
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180
E-mail: szymansk@rpi.edu

***Abstract*— This paper proposes a novel method of fusing models for classification of unbalanced data. The unbalanced data contains a majority of healthy (negative) instances, and a minority of unhealthy (positive) instances. The applicability of this type of classification problem with security applications inspired the naming of such problems as security classification problems (SCP). The area under the ROC curve (AUC) is the metric utilized to measure classifier performance, and in order to better understand AUC and ROC behavior, pseudo-ROC curves created from simulated data are introduced. ROC curves depend entirely upon the rankings created by classifiers. The rank distributions discussed in this paper display classifier performance in a novel form, and the behavior of these rank distributions provides insight into classifier fusion for the SCP. Rank distributions, which illustrate the probability of a particular rank containing a positive or negative instance, will be introduced and used to explain why synergistic classifier fusion occurs.**

I. Introduction

This paper discusses methods utilized to achieve synergistic classifier fusion and the underlying theory explaining why this synergistic classifier fusion occurs. The fusion methods described provide consistently robust solutions to the security classification problem (SCP). A security classification problem is a classification problem which involves a majority of negative instances and a minority of positive instances. The severity of the imbalance confounds feature selection and model validation. This problem occurs simply due to our inability to learn from few or no positive instances. The broad application of this problem to the security domain inspired titling such problems as security classification problems (SCP).

Combinations of multiple classifiers, or ensemble techniques, is a very active field of research today. However, the field remains relatively loosely structured as researchers continue to build the theory supporting the principles of classifier combinations [20]. Significant work in this field has been contributed by Kuncheva in [18], [19], [20], [21]. Bonissone et. al. investigated the effect of different fuzzy logic triangular norms based upon the correlation of decision values from multiple classifiers [2]. Evangelista et. al. discussed fuzzy aggregation concepts for unsupervised classifier fusion in [8], [9]. The majority of work in this field has been devoted to supervised learning, with less effort addressing unsupervised problems [24]. The research that does address unsupervised ensembles involves clustering almost entirely. There is a vast amount of literature that discusses subspace clustering algorithms [22].

Recent work involving ensemble methods which contain combinations of feature subsets include Breiman's work on Random Forests [5] and Ho's work with Random Subspaces [16]. Both of these methods use a random approach for subspace selection, which is different from the method proposed in this paper. Furthermore, both of these authors utilize decision trees and the average function to fuse or aggregate the ensemble. This paper will show that the average aggregator alone is not the best aggregator for the security classification problem. The novel contributions of this paper include ROC curve analysis with simulated data creating pseudo ROC curves, improved understanding of classifier rankings through rank distributions, and an ensemble method accompanied by a robust fusion metric which consistently creates synergistic performance when solving the SCP.

This paper is organized as follows: Sections II and III introduce pseudo-ROC curves and rank distributions, respectively, and section IV discusses the behavior of fused ranks. Section V introduces the leave-l-features-out ensemble method which leads to the experimental results observed with a range of fuzzy-logic inspired fusion metrics. All of this work ties together in an effort to illustrate why certain rank fusion methods work well.

II. Pseudo-ROC Curves

A study of pseudo-ROC curves and rank distributions will provide support and insight to the underlying behavior of classifier fusion for the security classification problem. This discussion is critical in understanding why certain classifier fusion metrics work best when fusing multiple models in the security classification domain.

ROC curves are based upon ranks. The area under the ROC curve, which is equivalent to the Mann-Whitney U statistic, is calculated entirely by ranks[1]. Given N total instances, p positive instances, and $b = N - p$ negative instances, the

[1]A more thorough discussion of ROC curve theory can be found in [3], [7], [8], [12]. AUC calculations are also illustrated in Table II.

rank values will be defined as follows: Let $S_j, j = 1, ..., b,$, represent the rank of each negative instance.

$$\sum_{j=1}^{b} S_j = W_r$$

$$W_r - \frac{1}{2}b(b+1) = W_{YX}$$

$$W_{YX}/pb = AUC = U$$

Furthermore, a well known property of U supports the following statement: U equals the probability that any random positive instance is ranked higher than a negative example. Stated formally, let us refer to $R(\mathbf{x}_i)$ as the rank of observation $\mathbf{x}_i$. The aforementioned property of the Mann-Whitney U statistic indicates the following:

$$U = P\{R(\mathbf{x_i}|y_i = 1) < R(\mathbf{x_j}|y_j = -1)\}$$

This property enables the creation of pseudo-ROC curves. If an assumption is made that a classifier has a certain discriminating ability reflected in the AUC or Mann-Whitney U statistic, artificial ranks can be created with pseudo-random numbers. The simplest way to accomplish this is assuming normal distributions for the sake of creating artificial ranks. Suppose A and B are two random variables such that $P(A > B) = U$, or equivalently $P(A - B > 0) = U$. If $W = A - B$, and it is assumed that W is a standard normal random variable, clearly A and B can be defined as normal random variables with a variance of .5. The following table provides examples of this relationship ($w \sim N(\mu, \sigma^2)$ represents a normal distribution with a mean of μ and a variance of σ^2, with the same notation for the distributions of a and b).

TABLE I
EXAMPLES OF DISTRIBUTIONS FOR CREATING ARTIFICIAL RANKS

$P(A > B) = .9$	$w \sim N(1.28, 1)$	$a \sim N(1.28, .5)$	$b \sim N(0, .5)$
$P(A > B) = .8$	$w \sim N(.84, 1)$	$a \sim N(.84, .5)$	$b \sim N(0, .5)$
$P(A > B) = .7$	$w \sim N(.52, 1)$	$a \sim N(.52, .5)$	$b \sim N(0, .5)$

Given the distributions shown in table I, it is now possible to create random numbers which will behave with the desired probability of $P(A > B) = U$. This will also enforce that $P\{R(A) > R(B)\} = U$.

These rankings will now enable the creation of pseudo-ROC curves with an area under the curve equivalent to U. The essence of this method is that it allows for the study of ROC curves where control variables consist of U, the number of positive examples, and the number of negative examples. An example of five pseudo-ROC curves with $U = .9$ is shown in figure 1.

Pseudo-ROC curves serve a multitude of purposes. ROC curves are a very popular method to assess the performance of a binary classifier. Empirical research involving ROC curves has largely been limited to the analysis of curves created by the output of models with real data. The study of ROC curves solely created from the output of classification models limits our ability to fully understand and explore the complete behavior of ROC curves and ranks. The study of pseudo-ROC curves places a number of parameters into the hands of the researcher - the discriminating power (reflected in the U statistic), proportion of the classes, and total number of observations are all parameters controlled by the researcher with pseudo-ROC curves.

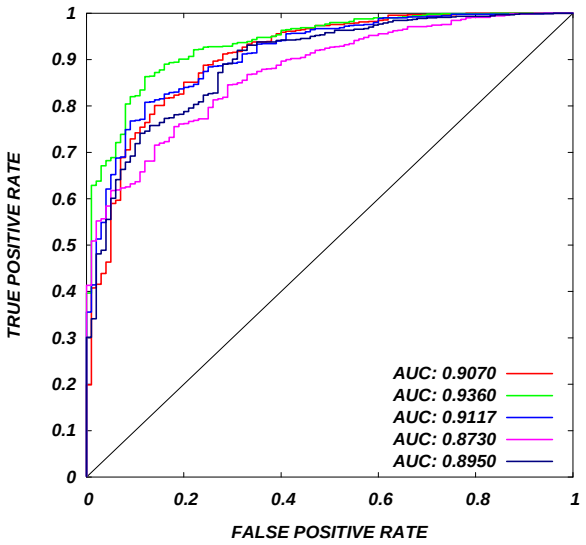

Fig. 1. Five pseudo-ROC curves

Researchers who discuss ROC theory have largely confined their discussion to the topic of the nonparametric statistics which impact ROC curves. This is primarily the Wilcoxon Rank Sum statistic and Mann-Whitney U statistic [3], [12], [14]. Exceptions to this include [7], [11], [13] where the authors have taken a deeper look at the meaning of the ROC curve, but the concept of the pseudo-ROC curve and use of this method to improve our understanding of ROC curves is a novel approach.

III. RANK DISTRIBUTIONS

Simulated rank distributions, created in the same spirit as pseudo-ROC curves, expose another dimension of analysis which supports the ultimate goal of this research, synergistic fusion of classifiers. Given a U statistic and desired number of positive and negative instances, it is possible to create rank distributions. Let us consider p positive instances, b negative instances (choosing the letter b to signify a benign or negative observation), and $N = p + b$ total observations. The rank distribution will be a discrete probability distribution, or probability mass function, with $1...N$ possible states, or ranks. Rank distributions reflect the likelihood that a particular rank is a positive or negative observation. For every case there is a given U, p, and b. This information is all that is necessary to create two rank distributions, one for positive observations and one for negative observations.

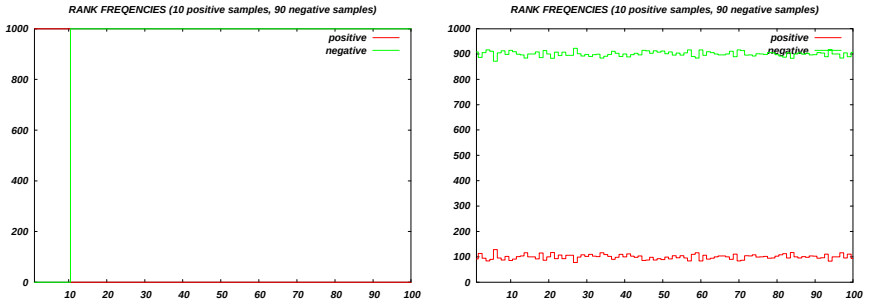

Fig. 2. The histograms on the left represent perfect classification($U = 1$), and the histograms on the right represent a random (meaningless) classifier ($U = .5$). These histograms resulted from simulations each with 90 positive instances, 10 negative instances, and 1000 simulation runs. The histograms clearly indicate uniform distributions.

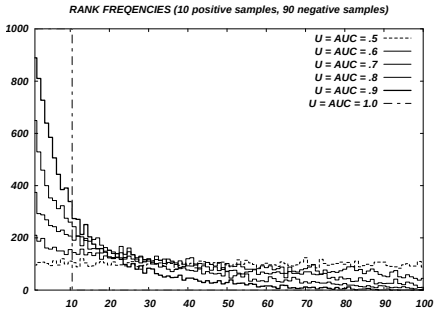

Fig. 3. The histograms shown above illustrate how the rank frequencies for the **minority class** transition as U spans the spectrum between .5 and 1.

A. Utilizing Simulation to Create Rank Distributions

Simulation will be utilized to study these distributions. As stated in [1], estimating probabilities by simulation due to pragmatic necessity (because the analytical solution is very difficult) is an acceptable approach. Given a simulation that models behavior based upon true probabilities, the simulation estimates these probabilities with high accuracy. The combinatoric complexity and implications of order statistics involved with these rank distributions become problematic in creating an analytical solution for the probability mass functions. This complexity will be briefly discussed to expose an open problem for future research. Suppose that we are interested in a rank distribution with $p = 1$ positive instances and b negative instances. This rank distribution can be solved using the binomial probability distribution. Let r represent the rank of the one positive instance, where $r \in (0, 1, 2, ...b)$. If $r = 0$, the positive instance is ranked first and has come out on top of all negative instances; if $r = b$, the opposite is true (lowest ranking). Given a prediction model with some accuracy, intuition indicates that $P(r = b)$ should be small, and $P(r = 0)$ or at least the probability that r is close to 0 should be large. This can be modeled as follows:

$$P(r|b) = \binom{b}{r} U^{b-r}(1-U)^r$$

However, this is a trivial case when $p = 1$, which is typically never the case. Given a value of $p > 1$, complexity of the analytical solution grows quickly. There are two apparent ways to solve the problem analytically for $p > 1$. The first involves attempting to create a discrete probability distribution, similar in essence to the binomial distribution above. This involves managing a distribution that will contain a high degree of combinatoric complexity. The other approach involves studying the order statistics of the underlying distributions that generated the ranks. If q is a continuous random variable which generated the ranks for positive instances, and c is the same for the negative instances, then $(q_{(1)}, q_{(2)}, ..., q_{(p)},)$ and $(c_{(1)}, c_{(2)}, ..., c_{(b)},)$ represent the ordered values of the positive and negative instances, respectively. Defining $P(r = 0)$ as the probability a positive instance ranks above all other instances, the solution is relatively simple, assuming that the underlying distributions of q and p are known. However, solving the probability that r equates to (1,2,...,b+p) is not as simple.

$$P(r = 0) = P(q_{(1)} > c_{(1)})$$

$$P(r = 1) = P\{(q_{(2)} > c_{(1)}) \cup (c_{(1)} > q_{(1)} > c_{(2)})\}$$

$$P(r = 2) =$$

$$P\{(q_{(3)} > c_{(1)}) \cup (c_{(1)} > q_{(2)} > c_{(2)}) \cup (c_{(2)} > q_{(1)} > c_{(3)})\}$$

...

The study of these probabilities through simulated rank distributions is much more practical and demonstrates sufficient evidence of the behavior of these rank probabilities.

B. Behavior of Rank Distributions

These rank distributions have interesting behavior which directly impact the outcome of ROC curves. First consider two extreme cases. The first extreme case involves a classifier with no predictive power, and in this case $U = .5$. The resulting rank distributions would simply be two uniform distributions ranging between 1 and N (shown as the right plot in Figure 2). The other extreme case would be the perfect classifier where $U = 1$. This case would also create two uniform distributions, however the distribution for the ranks of the positive observations would range from $1...p$ and the uniform distribution of the negative observations would range from $(p + 1)...N$ (shown as the left plot in Figure 2).

Typically, however, U ranges somewhere between .5 and 1. As U increases from .5 to 1, the histogram representing the positive class experiences a reduction in the frequencies of the larger ranks and an increase in the frequencies of the smaller ranks. It is also evident as this shift occurs that the distribution of the positive (minority) class begins to take on the familiar form of the exponential distribution. Figure 3 illustrates exactly this phenomenon. This figure illustrates an example which involves 10% positive instances, and each line represents the rank distribution of the positive instances for different values of U. Notice the familiar shape of the exponential distribution emerging as U transitions.

IV. Behavior of Fused Ranks

Analyzing how these rank distributions behave provides insight for model fusion. The fusion metric utilized in the most popular ensemble techniques such as random forest, bagging, and the random subspace method, is the average [4], [5], [16]. The average is a powerful aggregator, especially if all of the models possess roughly the same predictive power.

Model fusion involves considering several models, all of which measure the same observations, and for each observation fuse the results of each model to arrive at a final decision value for each observation.

Good prediction occurs for a model when the decision value distribution of the positive class achieves separation from the decision value distribution of the negative class. Fusion with the average function invokes the properties of the central limit theorem, and improved separation occurs as a result of variance reduction. This can be further explained in a brief example. Assume that three models each create a distribution for the decision values of the negative class with a mean of -1 and a variance of 1. Assume the distributions of the positive class have a mean of 1 and a variance of 1. The fused model, using the average aggregator, will create a distribution for the decision values of the negative class with a mean of -1 and a variance of 1/3. The positive class will have a mean of 1 and variance of 1/3. Tighter distributions for both the positive and negative classes creates improved prediction.

A. Why the Average and $\min$ Fusion Metrics Work

The disadvantage of the average aggregator involves the equal weighting and inclusion of all models, good and bad. When fusing security classification problem models, it is likely that some of the models are poor classifiers. Therefore, it is desirable to utilize fusion that is robust against poor classifiers without knowing which classifiers are poor. This is precisely what the $\min$ aggregator accomplishes. Given an unbalanced classification problem, the rank distribution for the positive class, assuming the model predicts well, clearly favors the smallest rankings and quickly tail off (see figure 3). The behavior is remarkably similar to the exponential distribution. An interesting property of the exponential distribution involves the distribution of the $\min$ of this distribution. Given an exponential random variable x distributed with a mean (and standard deviation) of θ, the distribution of the $\min$ of x, $x_{(1)}$, is exponential with a mean (and standard deviation) of θ/n. The brief proof of this follows:

$$f(x) = \frac{1}{\theta} e^{-x/\theta}$$

$$F(x) = 1 - e^{-x/\theta}$$

$$F(x_{(1)}) = 1 - (1 - F(x))^n$$

$$F(x_{(1)}) = 1 - e^{-nx/\theta}$$

$$f(x_{(1)}) = \frac{n}{\theta} e^{-nx/\theta}$$

This property indicates that the distribution of $x_{(1)}$ contains less dispersion, concentrating in a tighter range. This concentration enables separation, however more importantly the $\min$ fusion metric creates robustness against poor classifiers. In the security classification problem, we now understand from our study of rank distributions that given a good model the probability of encountering a large rank value for a positive instance is small. It is more likely to observe a small rank value. The $\min$ fusion metric indiscriminately eliminates large rank values. This indiscriminant elimination works based on the fact that there are a small number of positive instances. There is also another subtle contribution created. Haykin indicates in [15] that a fundamental in every classification or pattern recognition problem involves ensuring the inclusion of all available information. The $\min$ aggregator works based upon our assumption that there is a small number of positive instances. This information, although obvious, contributes and improves performance by assisting our fusion metric selection.

B. Experimental Results with Simulated Data

Figure 4 illustrates rank distributions for several fusion metrics. This experiment involved fusing a model set where 60% of the models predicted well at a rate of $U = .9$, and 40% of the models create random prediction at a rate of $U = .5$. It is assumed that 10% of the observed instances are positive. Figure 4 illustrates the performance of the $\min$, $\max$, and average fusion metrics. The $\max$ fusion metric does not work well. This metric flattens the dispersion of the positive instances, creating poor separation between the positive and negative rank distributions. The average fusion metric clearly works well, creating two normal distributions produced from the effect of the central limit theorem. These distributions contain minimal overlap. The $\min$ aggregator creates the familiar exponential distribution effect, and although there is separation, the quality of the separation is questionable. In the spirit of fuzzy logic, it is possible to combine these metrics. This combination, created simply by computing $\frac{\min + avg}{2}$, creates two distributions which appear tighter than those created from the average fusion metric without any obvious improvement in performance.

The improvement in performance is not evident until plotting the ROC curves from these rank distributions. Figure 5 contains these ROC curves. There are five curves shown, with two models (40%) providing no predictive power ($U \cong .5$). It is apparent that the average and $\min$ aggregators perform well, achieving performance close to the convex hull of the ROC curves. However, it is the fusion metric of $\frac{\min + avg}{2}$ which clearly performs best, exceeding the convex hull and creating a synergistic fusion effect. Paired t test results of this experiment for 100 iterations indicated that significance of the difference between $min + avg/2$ and avg was nearly 1.00.

V. Experimental Results

Before detailing the results obtained with rank fusion, it is important to describe the approach taken to apply rank fusion to typical classification datasets, and in particular, SCP

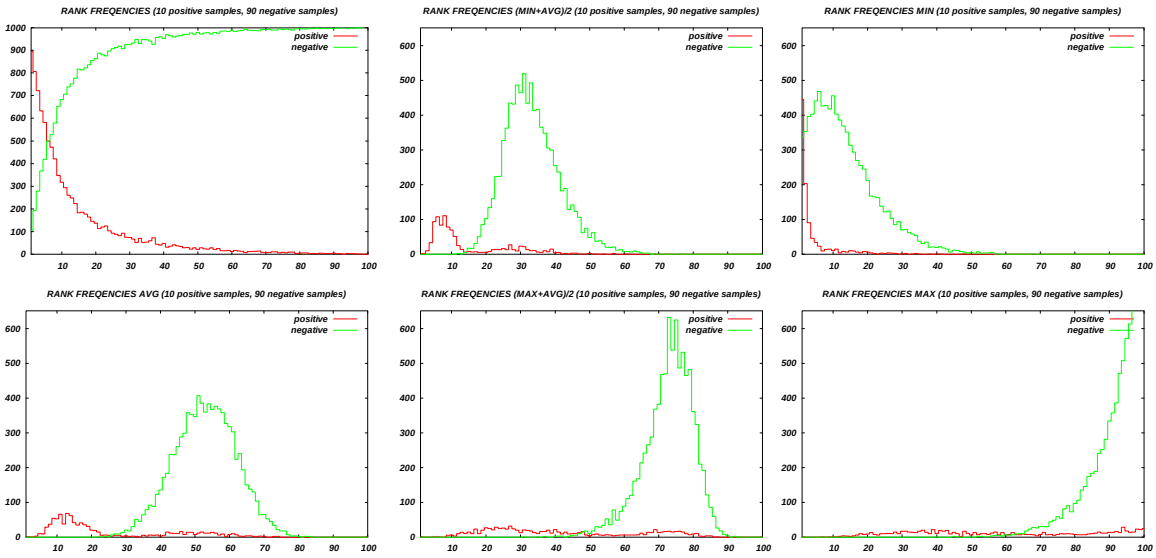

Fig. 4. Rank histograms of fusion methods generated with 1000 iterations (each iteration considered 10 positive samples and 90 negative samples, and assigned a rank value to each; 400 iterations assigned ranks randomly, and 600 discriminated at a rate of U = .9

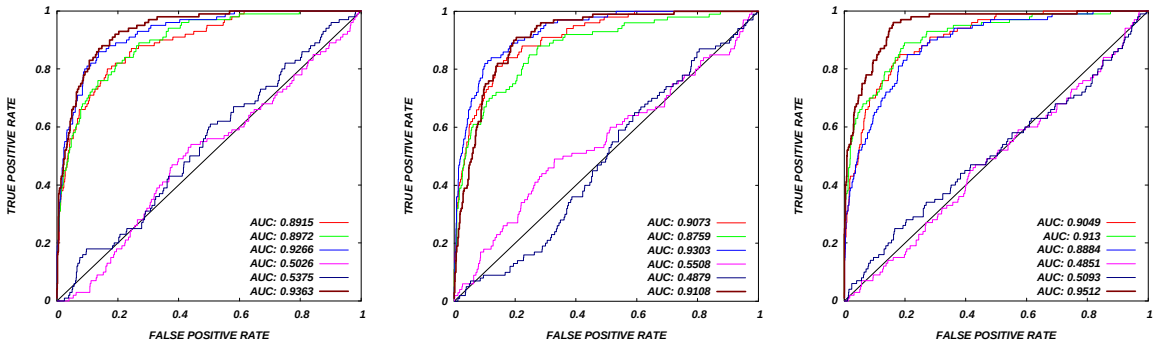

Fig. 5. ROC curves created from three meaningful models, two random models, and the fused model (bold line). From left to right, the fusion aggregator creating the bold ROC curve was the average, min, and $(\min + avg)/2$

datasets. A common problem in the security classification domain involves managing a large number of features and difficulty in feature selection. Feature selection is difficult because selecting salient features for a classification problem typically requires an adequate labeled sample of the positive and negative classes in training data. In the security classification domain, the positive class is a minority and often may not exist or may exist very sparsely in the training data. In either case, feature selection quickly becomes problematic. The solution proposed in this research involves using a subset of features in multiple models and then fusing these features to create a final decision value.

Working in the security classification problem framework, it is assumed that a dataset, $\mathbf{X} \in \mathbb{R}^{N \times m}$, exists. $\mathbf{X}$ contains N instances or observations, $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N$, where $\mathbf{x}_i \in \mathbb{R}^{1 \times m}$. There are m variables to represent every instance. For every instance there is a label or class, $y_i \in \{-1, +1\}$. The unbalanced nature of this problem indicates a prevailing negative or healthy class and minimal instances of the positive or unhealthy class.

The next paragraphs describe a method to develop multiple subsets of data from a consolidated dataset for the purpose of creating an ensemble. This will be referred to as the "leave-l-features-out ensemble method". This method of building the ensemble is very similar to the leave one out model validation technique, except features instead of instances are left out. A small number, typically less than half and notated by l, of the features are removed for each model in the ensemble. This method to create multiple data subsets from an original dataset is very similar to Ho's technique proposed in [16]. Creating data subsets in this manner is a proposed technique which quickly creates multiple distinct representations of the same observations. An example illustrates this best.

Suppose ten features, $\{a, b, c, d, e, f, g, h, i, j\}$ existed in a dataset. Given $m = 10$, assume we choose a leave out quantity of $l = 2$. The first step is to randomly order the features: $\{g, j, a, b, i, d, c, e, f, h\}$. Next, create $m/l = F = 5$ models with the following features:

$\{a, b, i, d, c, e, f, h\}$
$\{g, j, i, d, c, e, f, h\}$
$\{g, j, a, b, c, e, f, h\}$
$\{g, j, a, b, i, d, f, h\}$
$\{g, j, a, b, i, d, c, e\}$

These models would all be trained with the same data observations, each model built with the aforementioned leave out strategy. This leave out strategy creates $F = 5$ models each with different performance, and typically several of the models predict well and several are likely to perform poorly. Next, introduce the test data to each model and collect the results (decision values for each model). For each observation within each subspace, the classifier will produce a decision value, D_{ij}, where D_{ij} represents the decision value from the j^{th} classifier for the i^{th} observation. o_{ij} represents the ordinal position, or rank, of D_{ij} compared within the j^{th} classifier. The purpose of using the rank rather than the actual value, or scaled value, of the decision value hinges on the issue of fusing models. When fusing models it is important that the variables which are being fused represent a similar unit of scale and magnitude. Using an old but applicable clichè, it is important to ensure that the fusion combines "apples and apples" rather than "apples and oranges". The problem with decision values involves the underlying (and unknown) distributions of these values. It is possible to normalize the decision values from each model, however even then there is still influence from the parametrics of the underlying distribution. In order to get away from this "apples and oranges" problem, ranks are fused. Ranks eliminate parametric influence. Algorithm 1 illustrates the leave-l-features-out ensemble method.

Algorithm 1 The leave-l-features-out ensemble

1: Select a number of features, l, to leave out from every model
2: Build $F = (m/l)$ (rounding up if m is not divisible by l) models from a training data set
3: Create consistent feature subsets in the test data and apply F associated models
4: Map $D_{ij} \rightarrow o_{ij}$
5: **for** $i = 1$ to N **do**
6: fuse o_{ij} for $j = 1..F$
7: **end for**

This ensemble method alleviates a number of problems which exist with the SCP. Leaving out a quantity of features reduces dimensionality for each model. It is certain that if a meaningless or falsely correlated feature exists in the data, at least one model will not consider that feature. Multiple models provide robust and redundant pattern recognition. One model created from a lump sum of all features (referred to as the base model) is susceptible to the curse of dimensionality and potentially pursuing meaningless patterns. When conducting supervised pattern recognition with balanced data, model validation enables feature selection (eliminating curse of dimensionality) and reduces the risk of meaningless pattern pursuit. Model validation is often not possible with the SCP, and therefore it is necessary to pursue other means which create generalization and robust models. The leave-l-features-out ensemble method creates this for the SCP.

A. A Toy Problem

It is useful to illustrate rank fusion through a toy problem. Table II shows three models, two of which perform adequately and one which appears to have no predictive power. The resulting ranks created by the $\min$, avg, and $\min + avg/2$ functions are also shown. Realize that the resulting columns do not equate to $f(R_1(\mathbf{x}_i), R_2(\mathbf{x}_i), R_3(\mathbf{x}_i)) = f(o_{i1}, o_{i2}, o_{i3})$. The aggregation columns represent $R(f(o_{i1}, o_{i2}, o_{i3}))$. Particularly for the $\min$ function, ties must be solved which is done simply at random. Immediately following any aggregation, decision values are immediately mapped to ranks, or o_{ij}. Table II is consistent with Algorithm 1; if desired, an interested reader could recreate the last three columns to reinforce the concept.

TABLE II
A TOY RANK FUSION PROBLEM

True Class	$\mathbf{R}_1$	$\mathbf{R}_2$	$\mathbf{R}_3$	$\min$	avg	$\frac{\min + avg}{2}$
0	8	15	10	14	11	14
0	16	16	17	19	19	19
0	17	18	9	15	17	17
0	15	20	1	1	13	9
0	11	14	2	5	10	6
0	14	9	18	16	16	15
0	6	6	3	8	1	4
0	13	7	5	11	8	10
0	18	17	13	18	18	18
0	7	8	7	13	6	11
0	19	19	19	20	20	20
0	12	11	15	17	14	16
0	4	10	11	9	8	7
0	5	13	6	12	7	8
0	20	3	16	7	15	13
1	3	2	12	4	2	2
1	9	4	20	10	11	12
1	1	5	14	3	5	3
1	10	1	8	2	4	1
1	2	12	4	6	3	5
AUC	0.867	0.880	0.427	0.867	0.840	0.893

B. Exploratory Experimentation of leave-l-features-out Ensembles

Initial experimental results with this fusion method explored a range of fusion methods, primarily focused on the $\min$, average (annotated as avg), $\max$, and the spectrum spanning these functions. Four datasets were examined in this experiment. The Network Intrusion dataset was used twice with a different leave out value.

The learning model used for each dataset was the one-class SVM with a linear kernel. The theory discussing one-class SVMs can be found in [23], [25]. The software utilized was LIBSVM [6]. Seven different fusion techniques were examined to include the $algebraic\ product$, $\min$, $(\min + avg)/2$, avg, $(\max + avg)/2$, $\max$, $algebraic\ sum$. When fusing models with fuzzy logic aggregators (especially the $algebraic\ sum$

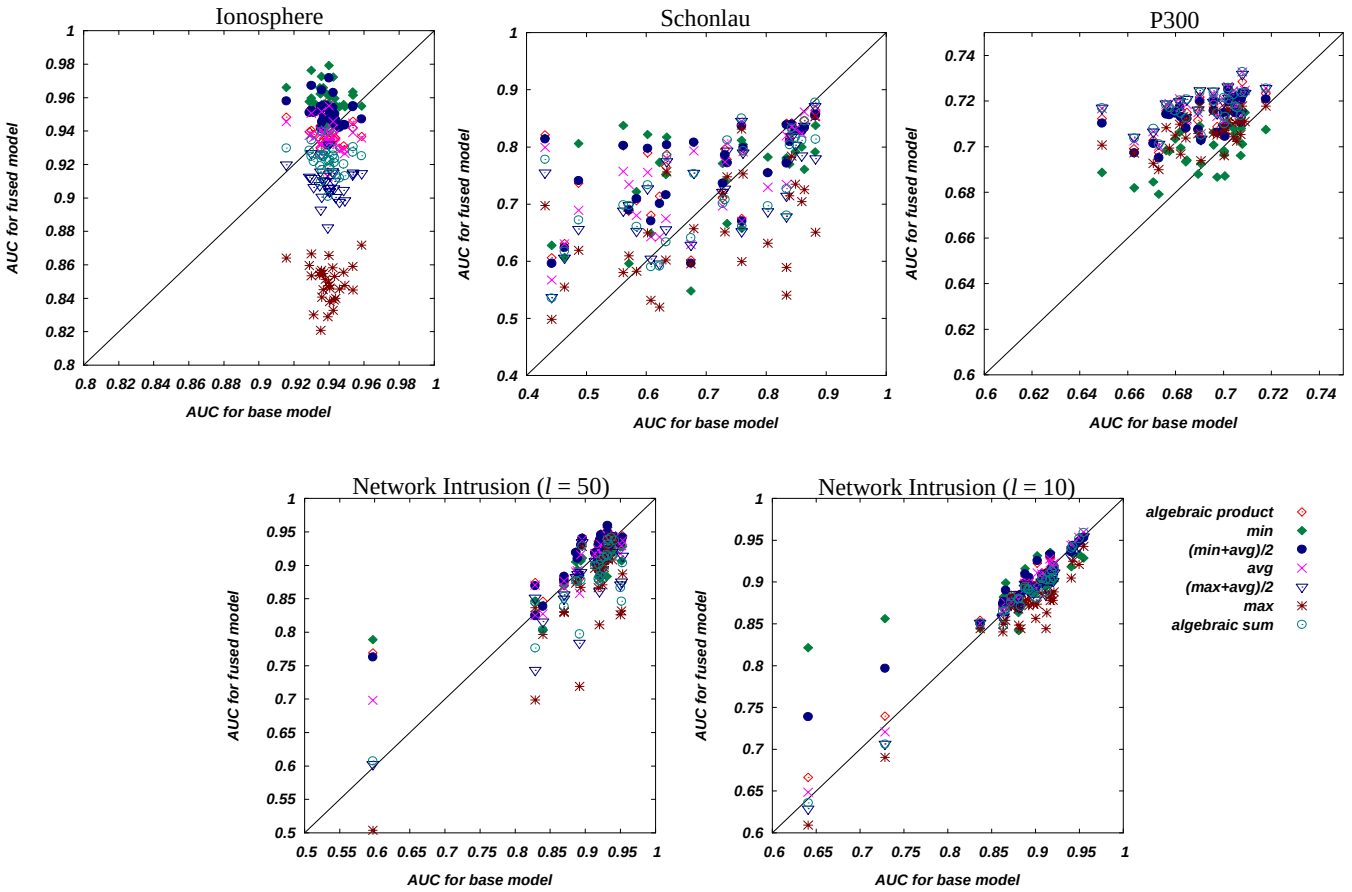

Fig. 6. Scatterplot comparison of fused models versus the base model.

Dataset	m	N	$p^{\dagger}$	l	**comment**
P300	100	4200	2100	10	Courtesy of Wadsworth Lab, www.wadsworth.org
Ionosphere	34	351	126	5	UCI Repository
Schonlau	26	5000	231	5	www.schonlau.net,[8], [9], [10]
Network Intrusion	137	5393	250	10	see ([17])
Network Intrusion	137	5393	250	50	see ([17])

TABLE III
DATASETS EXAMINED († p REPRESENTS THE NUMBER OF POSITIVE INSTANCES IN THE DATASET.)

Dataset	*algebraic product*	min	$\frac{\min+avg}{2}$	avg
Ionosphere	NS	0.000	0.000	NS
Schonlau	0.002	0.039	0.004	0.010
Network Intrusion, $l = 50$	0.042	NS	0.044	0.630
Network Intrusion, $l = 10$	0.000	0.280	0.015	0.004
P300	0.000	0.004	0.000	0.000

Dataset	$\frac{\max+avg}{2}$	max	*algebraic sum*	
Ionosphere	NS	NS	NS	
Schonlau	0.268	NS	0.149	
Network Intrusion, $l = 50$	NS	NS	NS	
Network Intrusion, $l = 10$	NS	NS	NS	
P300	0.000	0.000	0.000	

TABLE IV
PAIRED t-TEST VALUES FOR THE COMPARISONS IN FIGURE 6.

and $algebraic\ product$), it is necessary to map the rank values into a range between 0 and 1. This is a simple scaling which can be done without the loss of any information. In order to measure the benefit of these fusion methods, the experiment involved 30 iterations of each fusion method, each with a different random training and test sample. The benchmark comparison for each fusion method was a one class SVM model built from all available variables in the dataset. This is referred to as the **base model** in the plots. Essentially the comparison involved whether the ensemble of one-class SVMs created with leave-l-features out improved performance over the base model. The plots in figure 6 indicate performance of the fusion methods for each of these datasets.

This experiment provided strong indications that certain fusion methods perform best. The paired t-test values in Table IV shows that the $(\min + avg)/2$ aggregator was the only function which illustrated a significant difference (improvement with fusion) for every dataset. Experiments with a t-test value of 'NS' experienced better performance from the base model than

the fused model. 'NS' indicates that the fused method was 'Not Significantly' better than the base model.

Figure 6 provides insight into the behavior of various fusion methods. Figure 6 is the visualization for the t-test values in Table IV. For each dataset, 30 different experiments were conducted. Each of these experiments consisted of a different random split of the data (typically a 50 / 50 split between training and test data for each dataset), and a different random shuffling of the features in order to achieve different leave out ensembles (as per algorithm 1). The base model performance is shown on the horizontal axis. The vertical axis represents the performance of the fused models. Points plotted above the diagonal represent superior fused model performance. Points along the diagonal represent negligible differences in performance, and points below the diagonal illustrate superior performance of the base model. It is apparent from both Figure 6 and Table IV that the $(\min + avg)$ / 2 aggregator worked well for every model.

VI. Conclusion

Simulating rank distributions and creating pseudo ROC curves provides unique insight and a number of advantages. The insight provided by possessing control of parameters which include prediction power, balance of classes, and number of models ensembled, enables analysis which is not possible when analyzing fusion metrics and ROC curves created from actual data. Analysis with actual data limits a researchers control of the parameters.

A critical advantage of this simulation analysis involves the convincing evidence created through simulation. Since the rank distributions and ROC curves are created from first principles and model generic, there is no concern of bias due to the characteristics of the data or behavior of a particular model. Results are general, and the general results of the simulation analysis provide a broader range of applicability for the research included in this chapter.

Acknowledgment

The authors would like to thank Pat Driscoll for insightful comments and the authors of LIBSVM [6] for the use of their software.

References

[1] Dimitri P. Bertsekas and John N. Tsitsiklis. *Introduction to Probability*. Athena Scientific, 2002.

[2] Piero Bonissone, Kai Goebel, and Weizhong Yan. Classifier Fusion using Triangular Norms. Cagliari, Italy, June 2004. Proceedings of Multiple Classifier Systems (MCS) 2004.

[3] Andrew P. Bradley. The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms. *Pattern Recognition*, Volume 30(7):1145–1159, 1997.

[4] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[5] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] Chih Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. http://www.scie.ntu.edu.tw/ cjlin/libsvm, Accessed 5 September, 2004.

[7] James P. Egan. *Signal Detection Theory and ROC Analysis*. Academic Press, Inc., 2003.

[8] Paul F. Evangelista, Piero Bonissone, Mark J. Embrechts, and Boleslaw K. Szymanski. Fuzzy ROC Curves for the One Class SVM: Application to Intrusion Detection. In *Proceedings of the International Joint Conference on Neural Networks*, Montreal, Canada, August 2005.

[9] Paul F. Evangelista, Piero Bonissone, Mark J. Embrechts, and Boleslaw K. Szymanski. Unsupervised Fuzzy Ensembles and Their Use in Intrusion Detection. In *Proceedings of the European Symposium on Artificial Neural Networks*, Bruges, Belgium, April 2005.

[10] Paul F. Evangelista, Mark J. Embrechts, and Boleslaw K. Szymanski. Computer Intrusion Detection Through Predictive Models. pages 489–494, St. Louis, Missouri, November 2004. Smart Engineering System Design: Neural Networks, Fuzzy Logic, Evolutionary Programming, Data Mining and Complex Systems.

[11] Tom Fawcett. Using Rule Sets to Maximize ROC Performance. San Jose, CA, 2001. IEEE International Conference on Data Mining.

[12] Tom Fawcett. ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. Palo Alto, CA, 2003. Technical Report HPL-2003-4, Hewlett Packard.

[13] Tom Fawcett and Foster Provost. Robust Classification for Imprecise Environments. *Machine Learning Journal*, 42(3):203–231, 2001.

[14] James A. Hanley and Barbara J. McNeil. The Meaning and Use of the Area Under the Receiver Operating Characteristic Curve. *Radiology*, 143(1):29–36, 1982.

[15] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, second edition, 1999.

[16] Tin Kam Ho. The Random Subspace Method for Constructing Decision Forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[17] Alexander Hofmann, Timo Horeis, and Bernhard Sick. Feature Selection for Intrusion Detection: An Evolutionary Wrapper Approach. Budapest, Hungary, July 2004. International Joint Conference on Neural Networks.

[18] Ludmila I. Kuncheva. 'Fuzzy' vs. 'Non-fuzzy' in Combining Classifiers Designed by Boosting. *IEEE Transactions on Fuzzy Systems*, 11(3):729–741, 2003.

[19] Ludmila I. Kuncheva. That Elusive Diversity in Classifier Ensembles. Mallorca, Spain, 2003. Proceedings of 1st Iberian Conference on Pattern Recognition and Image Analysis.

[20] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley and Sons, Inc., 2004.

[21] Ludmila I. Kuncheva and C.J. Whitaker. Measures of Diversity in Classifier Ensembles. *Machine Learning*, 51:181–207, 2003.

[22] Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace Clustering for High Dimensional Data: A Review. *SIGKDD Explorations, Newsletter of the ACM Special Interest Group on Knowledge Discovery and Data Mining*, 2004.

[23] Bernhard Schölkopf, John C. Platt, John Shawe Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High Dimensional Distribution. *Neural Computation*, 13:1443–1471, 2001.

[24] Alexander Strehl and Joydeep Ghosh. Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions. *Journal of Machine Learning Research*, 3:583–617, December 2002.

[25] David M.J. Tax and Robert P.W. Duin. Support Vector Domain Description. *Pattern Recognition Letters*, 20:1191–1199, 1999.