Generating ROC Curves for Artificial Neural Networks

Kevin Woods,* *Member, IEEE*, and Kevin W. Bowyer, *Senior Member, IEEE*

***Abstract*—Receiver operating characteristic (ROC) analysis is an established method of measuring diagnostic performance in medical imaging studies. Traditionally, artificial neural networks (ANN's) have been applied as a classifier to find one "best" detection rate. Recently researchers have begun to report ROC curve results for ANN classifiers. The current standard method of generating ROC curves for an ANN is to vary the output node threshold for classification. In this work, we propose a different technique for generating ROC curves for a two-class ANN classifier. We show that this new technique generates better ROC curves in the sense of having greater area under the ROC curve (AUC), and in the sense of being composed of a better distribution of operating points.**

***Index Terms*—Neural networks, receiver operating characteristic (ROC) curves.**

I. Introduction

One method of specifying the performance of a classifier is to note its true positive (TP) rate and false positive (FP) rate for a data set. The TP rate is the percentage of target samples that are correctly classified as target samples. The FP rate is the percentage of nontarget samples that are incorrectly classified as target samples. For particular applications, we may require the classifier to operate at some point other than the one to which it naturally trained. Statistical classifiers have parameters that can be varied to alter the TP and FP rates. Each set of parameter values may result in a different (TP, FP) pair, or operating point.

An ROC curve is a plot of operating points showing the possible tradeoff between a classifier's TP rate versus its FP rate. The TP rate is commonly referred to as "sensitivity," and (1—FP rate) is called "specificity." Two typical ROC curves are shown in Fig. 1.

In practice, the errors that can be made by the classifier [FP and false negative (FN)] often have different "costs." In such cases, "profits" can be maximized by selecting the appropriate operating point on the ROC curve. In practical application, this requires that the underlying parameters of the classifier be easily manipulable to facilitate selection of the operating point [1]. The generation of ROC curves for traditional statistical classifiers is well understood. Analogous techniques are not so well understood for nonparametric classifiers, such as artificial neural networks (ANN's).

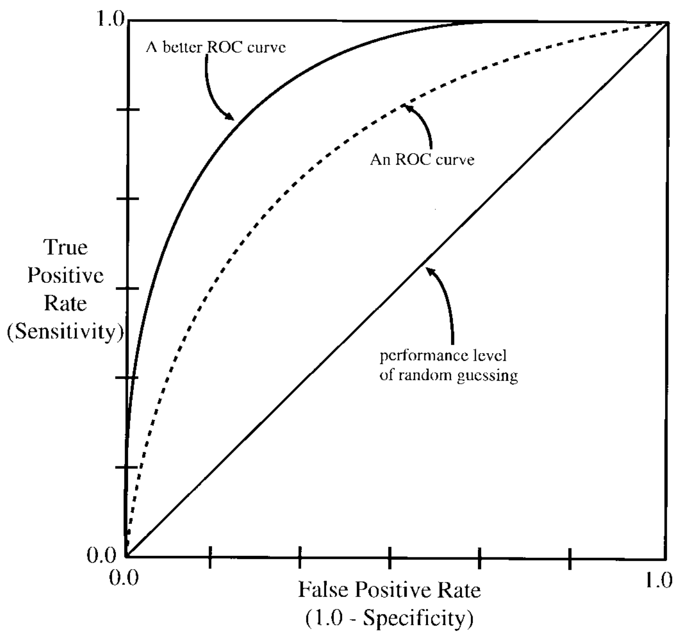


Fig. 1. Two typical ROC curves and the performance level that could be expected from random guessing. The greater the area of the unit square that lies below the ROC curve, the greater the power of the classifier.

The use of ANN's has recently become popular in medical imaging [2]–[6]. Since receiver operating characteristic (ROC) analysis is an established method of measuring diagnostic performance in medical imaging studies, this work examines methods for generating ROC curves for ANN's in a two-class problem. In this work, ROC points are generated from a single trained ANN by systematically varying some underlying parameter(s) in the network. The most common current method [3]–[6] is to vary a threshold value for the output node. Our work shows that a different method generates a better ROC curve in terms of both the area under the curve (AUC) and the distribution of operating points across the TP or FP range. Rather than varying the threshold value at the output node, we scale the bias weight for nodes on the first hidden layer of the ANN.

Section II gives some background on ANN decision boundaries, and how they can be manipulated to generate an ROC curve. Section III introduces the current standard method and our proposed method for generating ROC points for ANN classifiers. Section IV discusses the experimental methods, the

Manuscript received May 16, 1994; revised October 17, 1996. This work was supported by the Department of the Army under Grants DAMD17-94-J-4328 and DAMD17-94-J-4015. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was M. Viergever. *Asterisk indicates corresponding author.*

*K. Woods is with the Department of Computer Science and Engineering, University of South Florida, 4202 E. Fowler Ave., ENB 118, Tampa, FL 33620-5399 USA (e-mail: woods@bigpine.csee.usf.edu).

K. W. Bowyer is with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL 33620-5399 USA.

Publisher Item Identifier S 0278-0062(97)03949-9.

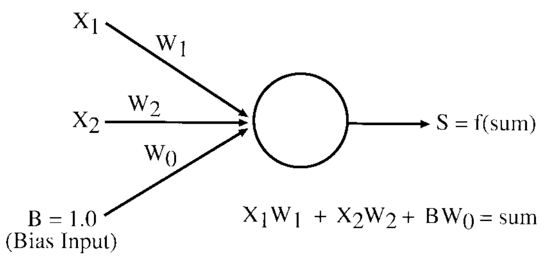


Fig. 2. A single node with a sigmoid activation function is capable of generating a hyperplane (a line in this 2-D case) in feature space. The weights W_1 and W_2, and the offset W_0 define the position and orientation of the hyperplane in feature space.

ANN's, and data sets. Section V presents experimental results for both methods of ROC generation on simulated datasets and several datasets from real applications. Section VI summarizes the work and draws conclusions about ROC generation for ANN's.

II. Theoretical Analysis: ANN Decision Boundaries

For the following discussions, it is assumed that the reader is familiar with basic ANN concepts and the backpropagation training algorithm. For a basic introduction to backpropagation neural nets, the reader is referred to [7].

An ANN classifier defines a potentially complicated decision boundary in the feature space. This boundary is formed as the nonlinear combination of a set of basic hyperplanes. Each basic hyperplane is defined by a node in the first hidden layer. A nonlinear decision boundary is formed by combining the basic hyperplanes via weighted connections to nodes in subsequent layers of the network.

Fig. 2 depicts a single sigmoidal node with two inputs, and an offset which is represented by a bias unit connection. The sigmoid activation function of the node is defined by

$$S = \frac{1}{1 + e^{-(\text{sum})}}. \qquad (1)$$

The input to the hidden node is

$$\text{sum} = W_1X_1 + W_2X_2 + W_0B \qquad (2)$$

which is the equation of a hyperplane in feature space (a line in two dimensions). The hidden node input can be put into the form of a line equation $y = mx + b$ as

$$X_2 = \frac{W_1}{-W_2}X_1 + \frac{W_0B}{-W_2}. \qquad (3)$$

In this form, it is easy to see that the orientation and location of the linear boundary defined by one node is determined by the connection weights from the network inputs and the bias offset. When an input to the node produces a weighted sum of zero, it corresponds to a point on the hyperplane, and the activation function outputs a 0.5. If the weighted sum is greater than zero, the point lies on one side of the hyperplane and the activation function outputs a value in the [0.5, 1.0] range. Similarly, if the weighted sum is negative, the point lies on the opposite side of the hyperplane and the activation function outputs a value in the [0.0, 0.5] range.

Now consider what happens in an ANN with multiple layers. Each node on the first hidden layer computes an output value which is a nonlinear scaling of the distance of the input point from the hyperplane defined by that node's weights. Each node on the next layer computes some nonlinear function of these nonlinearly scaled distances from the original hyperplanes. The value computed by the output node retains this character of being a nonlinear function of (a nonlinear function of ...) nonlinearly scaled distances from the original hyperplanes in feature space.

This simple example in two-dimensional (2-D) feature space allows us to more easily observe the roles that the hidden and output nodes take in dividing up the feature space into decision regions. A network with more first hidden layer nodes will simply have more hyperplanes that can be combined to create the final decision boundary. Methods of generating ROC curves for ANN's could manipulate parameters of the ANN controlling any or all of 1) the definition of the basic hyperplanes, 2) the definition of the combination of hyperplanes, or 3) the threshold value at the output node. It should be clear that what is desired is a manner of systematically shrinking/enlarging a decision region in feature space. The question, then, is how to do this most effectively and directly in the ANN framework.

III. Methods: Generating ROC Curves

The ANN must be trained before the ROC curve can be generated. The resulting network is referred to as a "basic trained network." This initial instance of the ANN provides one operating point. Based on the training data, each of the two methods we discuss manipulates one or more parameters of the basic trained network to give additional instances of the ANN. The result is a set of instances of the network chosen to represent points on the ROC curve. The goodness of this set of network instances is then evaluated using separate test data. So, ANN training involves both the learning of the network connection weights, and the estimation of classifier parameter settings for the purpose of generating an ROC curve.

A. The Standard Method: Output Node Thresholding

Generally, an ANN for a two-class problem has a single output node. The commonly accepted method of generating ROC points [3]–[6] is to vary a threshold (T_{out}) over the range of the output node activation (0.0–1.0). For each value of T_{out}, any feature vector which produces an output greater than or equal to T_{out} is classified a target, otherwise it is classified a nontarget.

An optimal set of T_{out} values, which correspond to successively higher levels of sensitivity, can be found by sorting the set of network outputs found when every training sample from the target class is input to the trained network. Each *distinct* value in this sorted set corresponds to a T_{out} that results in a new point in an ROC plot *for the training data*. Additional points, if desired, could be found by interpolation. For example, assume the training data provides T_{out} values of 0.6 and 0.7 corresponding to sensitivity levels of 40% and 50%, but nothing in between. A T_{out} of 0.65 might be assumed to produce an interpolated ROC point with a sensitivity of 45%. Note that this newly estimated value of T_{out} will not result in a new sensitivity level for the training data, but it *may* for a different data set with a similar distribution (such as the test data).

For the experiments reported here, we first find all of the distinct T_{out} values for the training set. Then, interpolating as described above, we estimate a set of T_{out} values corresponding to TP rates ranging from 0% to 100% in 1% intervals.[1] The goodness of this set of 101 T_{out} values as an ROC curve is then evaluated using the test set. Of course, it would be possible to generate an "optimal" ROC curve for the test data by selecting T_{out} values based on the test set. However, this would constitute learning classifier parameters directly from test data.

Analysis of the Standard Method: Standard ANN training algorithms are *not* designed to vary the strength of the output according to a training sample's proximity to the final decision surface. Ideally, all target samples would produce identical outputs of "1," and all nontarget samples would produce identical outputs of "0."

The typical ANN implementation contains an inherent weakness which may cause problems when the standard method is used to generate ROC curves. Since the sigmoid activation function asymptotically approaches zero and one, all inputs above a saturation value generate an output of one, while all inputs less than another saturation value generate an output of zero. These saturation values depend on the precision of the ANN implementation. For example, using the activation function of (1), all inputs greater than 14 generate outputs identical to six significant digits. The overall effect is that many different input samples appear identical to a hidden node, even with a double precision floating-point implementation. This grouping together of samples due to saturation occurs at all nodes in the network, resulting in multiple samples having the same network output regardless of proximity to the decision surface. As a result, varying a threshold on the output node activation may not generate many distinct ROC points.

B. The Proposed Method: Scaling Bias Weights for First Hidden Layer Nodes

Consider a basic trained network. At each node, the weighted sum of inputs is passed through a sigmoid function which determines the node activation

$$\text{node activation } = \frac{1}{1+e^{-(W_1X_1+W_2X_2+\cdots+W_dX_d+W_0B)}}. \tag{4}$$

In this expression, the $X_i, i = 1\cdots d$, are the inputs (other than the bias) to the node, the $W_i, i = 1\cdots d$, are the weights on the inputs, and W_0 is the weight on the bias input. The bias input, B, is fixed at 1.0 during the backpropagation learning phase. Scaling the bias weight, W_0, of a node to make it greater than the value learned during the training phase results in a greater node activation for a given input. Scaling W_0 to less than the value learned during training results in a lower node activation for a given input. Depending on other weights in the network, scaling the bias weight for a given node to make it greater could either increase or decrease the output node activation for a given input vector.

Testing Individual Nodes: The mechanics of the proposed method of generating ROC curves are as follows. First, for each hidden node on the first layer, we determine whether it is necessary to increase or decrease its bias weight in order to cause more samples to be classified as targets (i.e., increase TP and FP rates). This is accomplished in the following manner. The training set data is passed through the basic trained network, and the resulting TP and FP rates are noted. This TP/FP pair represents the "natural" ROC point to which the ANN has trained. Next, each first hidden layer node is considered individually, its bias weight is modified, and the resulting TP and FP rates are observed. In our implementation, the bias input for all nodes in the ANN is kept at a constant value of 1.0 during training. However, we can vary this value individually for each node on the first hidden layer. In effect, the bias input to a given node becomes a "scale factor" for the bias weight. We use scale factors of -9.0 and 11.0 to decrease and increase, respectively, the bias weight for this first step of the algorithm. We should note that for some nodes the TP and FP rates may not be affected when the bias weight is changed. These nodes have effectively been "turned off" during training and play no role in determining the network output. Such nodes are not considered in subsequent steps of the algorithm. So, at this point we know which "direction" (increasing, decreasing, or not at all) the bias weight of each first hidden layer node must be scaled in order to increase or decrease the ANN sensitivity.

Sweeping Out an ROC Curve: Since ANN training results in an initial point on the ROC curve, generating the rest of the curve involves changing the TP rate of the ANN and observing the corresponding FP rate. Thus, the next step to generate a ROC curve is to determine sets of scale factors that change the ANN's TP and FP rates in a desirable manner. First, we determine scale factors that increase the TP rate from the initial ROC point while increasing the FP rate as little as possible. Then, we determine scale factors that decrease the TP rate from the initial ROC point while decreasing the FP rate as much as possible. Conceptually, we are "sweeping out" an ROC curve by attempting to find operating points for the training data that change the TP rate from 0% to 100% while maintaining as low an FP rate as possible.

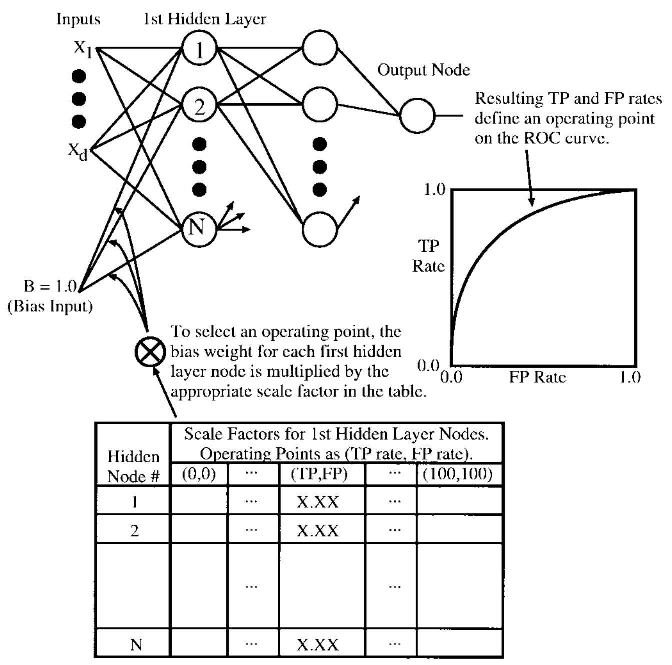
Implementation Details: Our implementation for determining sets of scale factors uses a lookup table. The table has a row for each first hidden layer node. Each column in the table contains the set of scale factors that correspond to a particular operating point. The completed lookup table specifies how to generate a set of ROC points from the basic trained network (see Fig. 3).

The following algorithm is used to dynamically build the lookup table. The lookup table initially has a single column with all entries set to 1.0. This column represents the operating point to which the ANN naturally trained. Next, the scale factors for all first hidden layer nodes are moved *simultaneously*

[1] In principle, any desired number of T_{out} values with any predicted spacing in TP rates can be created by appropriate interpolation between actual T_{out} values that are found directly from the training data. A set of 101 values with 1% intervals in TP rate was judged sufficient for our purposes here.

TABLE I
PORTION OF A LOOKUP TABLE THAT SPECIFIES THE SCALE FACTORS FOR THE BIAS WEIGHT INPUTS OF EACH FIRST HIDDEN LAYER NODE. EACH COLUMN IN THE LOOKUP TABLE CORRESPONDS TO AN OPERATING POINT FOUND FOR THE TRAINING SET DATA

Scale Factors to Decrease <---- <---- <---- TP and FP Rates					Default Scale Factors	Scale Factors to Increase ----> ----> ----> TP and FP Rates			
Hidden Unit #	... ...	TP=65.3 FP=12.0	TP=67.0 FP=12.7	TP=68.0 FP=14.0	TP=68.7 FP=16.0	TP=71.3 FP=17.3	TP=72.7 FP=21.3	TP=76.0 FP=28.0	... ...
Unit 9	...	1.0	1.0	1.0	1.0	1.0	1.06	1.06	...
Unit 10	...	1.0	1.0	1.0	1.0	1.1	1.16	1.26	...
Unit 11	...	0.97	0.97	0.97	1.0	1.0	1.06	1.06	...
Unit 12	...	1.6	1.6	1.0	1.0	1.0	0.94	0.94	...
Unit 13	...	1.0	1.0	1.0	1.0	1.0	1.06	1.06	...
Unit 14	...	1.0	1.0	1.0	1.0	1.0	1.06	1.06	...
Unit 15	...	1.0	1.0	1.0	1.0	1.0	1.06	1.06	...
Unit 16	...	1.0	1.0	1.0	1.0	1.0	1.06	1.06	...
Unit 17	...	1.0	1.0	1.0	1.0	1.0	1.06	1.06	...
Unit 18	...	0.94	1.0	1.0	1.0	1.0	1.03	1.03	...



Inputs
1st Hidden Layer
x_1
x_d
1
2
N
Output Node
Resulting TP and FP rates define an operating point on the ROC curve.
1.0
TP Rate
0.0
0.0
FP Rate
1.0
B = 1.0 (Bias Input)
To select an operating point, the bias weight for each first hidden layer node is multiplied by the appropriate scale factor in the table.
Hidden Node #
Scale Factors for 1st Hidden Layer Nodes. Operating Points as (TP rate, FP rate).
(0,0)
(TP,FP)
(100,100)
1
2
N
X.XX

Fig. 3. The lookup table of scale factors is used to set a desired operating point for the ANN classifier. In this figure, the bias connections to the second hidden layer and output nodes are not shown.

by the same amount in the direction that will increase the TP and FP rates. Based on the first step in our algorithm we know which direction the scale factors should be changed for each first hidden layer node, but not by how much. Assuming we would like to raise the TP rate in some specific increment, say 1%, we need to change the scale factors until at least 1% more targets from the training set are correctly classified as targets.

In order to do this efficiently, we perform a "binary search" over a range of possible scale factor changes until we zero in on the smallest scale factor change (within some desired degree of accuracy) that produces the desired increase in the TP rate. Once the new set of scale factors has been determined, the new TP and FP rates of this operating point are recorded.

If the density of the training data in feature space is the same near all regions of the decision boundary, then changing the scale factors of all nodes simultaneously makes sense. However, if the density of training data is different near different regions of the decision boundary, then it may be better to scale only a selected subset of the bias weights. To check for this possibility, the scale factor is changed in the appropriate direction for each first hidden layer node *individually* until the desired increase in the TP rate is found. Again, we record the new TP and FP rates.

So, if we have N nodes on the first hidden layer, then there are $N+1$ possible sets of scale factors, and therefore $N+1$ candidates from which to choose the next operating point. The candidate selected is the one which increases the TP rate by the desired amount while increasing the FP rate by the least amount. In the case of a tie, changing the scale factors for all first hidden layer nodes simultaneously takes precedence over changing the scale factor for a single node, and ties between changing different individual node scale factors are resolved arbitrarily. Finally, the next column to the right in the lookup table is created and filled in with the scale factors that correspond to the selected ROC point. This procedure is continued until reaching a TP rate of 100% on the training set.

A similar strategy is employed to create the columns of the lookup table to the left of the initial column. We find new ROC points to decrease the TP rate until either a TP rate or an FP rate of 0% is found for the training set. The only difference when selecting from the $N+1$ candidate operating points is that we select the one that reduces the TP rate the desired amount while reducing the FP rate the most.

Table I shows an example of part of an actual lookup table. Notice that sometimes all scale factors are changing simultaneously between successive operating points, and sometimes only a single scale factor is changing. Also, moving from left to right, notice that the scale factor increases for some nodes and decreases for others.

We should note that our solution for obtaining sets of scale factors for the bias weights is a heuristic. It is possible that better operating points may be found by changing more than one but less than all scale factors. Considering all possible subsets would require checking 2^N combinations, where N is the number of first hidden layer nodes, an exhaustive search would be out of the question for large N. Our solution should generally give good classification performance at reasonable computational cost. More sophisticated heuristics might be used to determine scale factor changes for varying numbers of first hidden layer nodes.

Once the lookup table has been completed, we can interpolate between scale factors for successive operating points to find a new set of scale factors that could result in a new operating point for a data set with a similar distribution as the training data. For example, using Table I and interpolating to get an operating point with a predicted TP rate of 75%, the scale factor for node ten would be approximately 1.23, while the scale factors for the other nodes remain unchanged. Using this interpolation procedure, we estimate sets of scale factors that correspond to 101 equally spaced ROC points (0%–100% TP rate in 1% increments). The goodness of the ROC curve resulting from these 101 sets of scale factors is evaluated using the test set.

Summary of the Proposed Method: From (1) and (2) we see that the weights on the inputs of a first hidden layer node define a hyperplane. Scaling the bias weight is equivalent to a parallel shifting of the hyperplane. Since the final decision surface is a combination of these hyperplanes, our proposed method determines how to manipulate the individual hyperplanes in a desirable manner. We first determine which direction each hyperplane should be shifted in order to drive the sensitivity up or down. Next, we determine how much the hyperplanes should be shifted in feature space to create a family of decision boundaries that correspond to a full set of operating points. This is accomplished by changing the scale factor in the appropriate direction for all first hidden layer nodes simultaneously and then individually, and selecting the change which produces the best new operating point.

IV. Experimental Methods

This section provides descriptions of the datasets, ANN configuration, training, testing, and the criteria for comparing the ROC curves generated by the two methods.

A. The Experimental Data

We created 28 sets of simulated data with various input dimensions, sample sizes, and data distributions. For 12 data sets, both classes have a Gaussian distribution with some overlap between classes. A 2-D example is shown in Fig. 4(a). We created training sets using sample sizes of 500, 1000, 1500, and 2000 with an equal number of samples from each class, and input dimensions of 2-D, 3-D, and 5-D. We created three sets of test data, one for each input dimension, with 2500 samples from each class. We would expect an ANN with a single hidden node to be sufficient for these data sets.

For 12 other simulated data sets, only the target samples have a Gaussian distribution. The nontarget samples are uniformly distributed throughout feature space except near the target sample mean. Fig. 4(b) shows a 2-D example. As before, we use training sets with four different sample sizes and three different input dimensions. Again, we have three sets of test data, one for each input dimension, with 5000 samples. We might expect that an ANN solution will require several hidden nodes to adequately solve these types of problems.

To test a more difficult type of classification problem, an additional four data sets were created with the 2-D exclusive-or distribution shown in Fig. 4(c). Here, the target class is composed of two normally distributed clusters, one in the lower-left quadrant and one in the upper-right quadrant of feature space, while the two normally distributed nontarget clusters occupy the upper-left and lower-right quadrants. As before, we have four different size training sets, and a test set with 5000 samples.

In addition to the simulated data, experimental results are reported for six real data sets taken from applications involving medical diagnosis. Four data sets were obtained from the UCI Repository of Machine Learning Databases (available via anonymous ftp to ftp://ics.uci.edu/pub/machine-learning-databases). Another two data sets were taken from applications in mammogram image analysis [8], [9].

Since training and test data is required, each data set was randomly divided into two subsets, retaining the same class distributions as the full data sets whenever possible.[2] Each subset is alternately used as training data and test data. Results are reported when each subset is used as training data. Therefore, we have results for 12 experiments involving data from real applications.

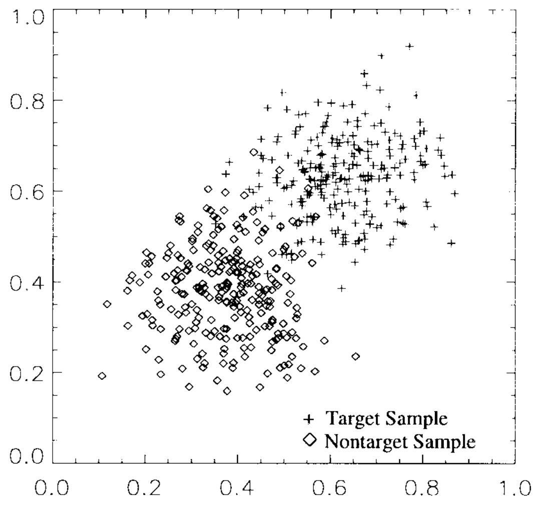
B. ANN Topology Selection and Training

All of the experiments reported here use fully connected backpropagation networks with sigmoid activation functions (output range: 0.0–1.0), a bias unit with weighted connections to all nodes, and a single output node. The value produced by the output node would typically be thresholded so that values greater than or equal to 0.5 are labeled "target," and values less than 0.5 are labeled "nontarget."

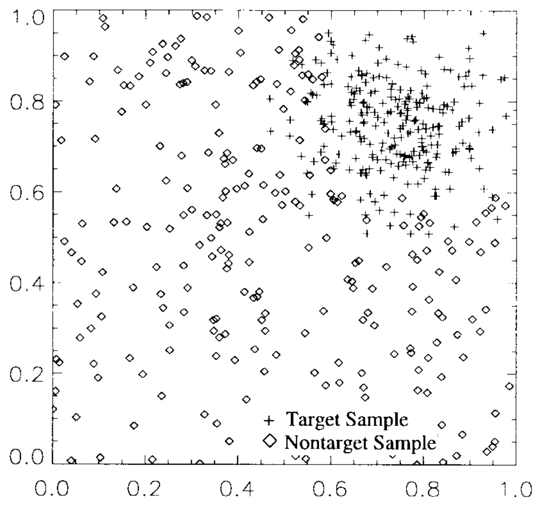
One important aspect of neural network applications is the determination of a suitable topology (i.e., the number of hidden layers and the number of nodes per hidden layer). Theoretically, a single hidden layer is all that is required, but in practice two or more hidden layers are often employed. While some rules of thumb exist, an optimal solution to this problem cannot be determined in any reasonable amount of time. Generally, a network topology is selected by trial and error. Many different networks are trained, and the "best" one is selected. This is the approach we have taken, but for brevity we will omit the details. For each set of training data, about 100 different one- and two-hidden layer network configurations were systematically evaluated. The best one-hidden layer network and the best two-hidden layer network found for each problem are utilized in the subsequent ROC experiments.

Once a topology has been determined, the ANN's are trained. Given a set of training data, we first standardize each feature value by subtracting the feature mean and dividing by the feature standard deviation. Next, the data is divided

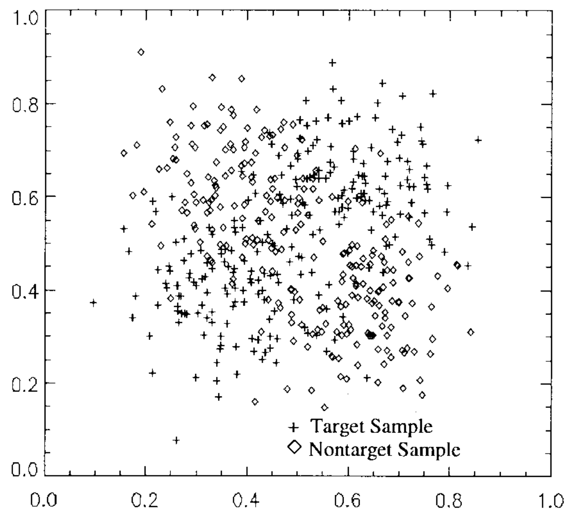
[2]The nature of mammography data did not permit division into two equal subsets. It was necessary to prohibit samples from the same mammogram image from belonging to the training data and the test data in the same experiment.



(a)



(b)



(c)

Fig. 4. Examples of 2-D simulated data sets in which (a) both classes have Gaussian distributions. (b) The target class has a Gaussian distribution, and the nontarget class has a uniform distribution. (c) The two classes exhibit an exclusive-or distribution.

into training and validation subsets. The ANN is trained using a standard backpropagation learning algorithm. Between each training epoch, the error rate (sum of squared error) is computed for the validation data. The network weights are saved each time this error rate drops. Training continues until there has been no improvement in the validation error rate for 500 epochs. The use of validation data to halt the training process prevents the ANN from over-fitting the solution to the training data, and leads to better generalization. The results for a trained ANN are naturally dependent on the initial values of the weights, which are usually random. This being the case, each ANN is trained five times with different random initializations, and the one with the best validation error rate is selected.

At this point, we have two basic trained networks for a given training set: the best one-hidden layer ANN, and the best two-hidden layer ANN. In this work, training also involves determining the appropriate parameter settings (T_{out} for the current standard method, and sets of bias weight scale factors for the proposed method) required to generate ROC curves. For the two ANN's, both methods of ROC curve generation use all the training data to determine appropriate parameter sets which correspond to predicted sets of 101 ROC points at 1% increments in the TP rate. Therefore, given a trained ANN and set of test data, both methods have the same potential to produce 101 evenly spaced ROC points.

C. Testing and Comparing ROC Curves

Given a set of test data, four ROC curves are generated (two methods each applied to the best one-hidden layer ANN and the best two-hidden layer ANN) which may or may not have 101 distinct points. We would like to determine if one particular method of generating ROC curves is better than the other, or if the ROC curves generated for the one-hidden layer ANN's are better than those generated for the two-hidden layer

TABLE II
SUMMARY OF ALL 80 ROC EXPERIMENTS

Results for ANNs with 1-Hidden Layer	# of times AUCs are stat. sig. different
Bias Scaling Method is better 33 times	19
Output Thresholding is better 6 times	0
Methods are equivalent 1 time	-
Results for ANNs with 2-Hidden Layers	# of times AUCs are stat. sig. different
Bias Scaling Method is better 28 times	20
Output Thresholding is better 10 times	2
Methods are equivalent 2 times	-

ANN's. The AUC is an accepted way of comparing overall classifier performance [10], [11]. Hanley and McNeil [12], [13] describe a method for comparing the AUC's of two ROC curves that have been derived from the same set of cases. We use this approach to determine if the difference between two AUC's is statistically significant.

The "best" classifier for an application may well depend on the particular combination of TP and FP rates that are required. Additionally, classifier performance at very low sensitivities or very high FP rates is usually not of practical interest. Thus, while the AUC is useful as a single number for making a general comparison of classifier performance, the distribution of the *selectable* operating points is also important. A finely sampled range of operating points is important because a TP/FP trade-off other than the one the ANN naturally trained to may be needed for a particular application. We will also compare the distribution of operating points generated by the two methods.

V. RESULTS

We have a total of 40 data sets for the purpose of comparing the ROC curve generating capabilities of the two methods. Since we have two ANN's for each data set, we begin by comparing the performance of the two methods for all 80 ROC experiments. For the 80 ANN's, the proposed method results in the best AUC 61 times, 39 times the difference is statistically significant. The current standard method results in the greatest AUC 16 times, only two of which show statistical significance. The methods produced equivalent AUC's three times. So, using the proposed method a statistically equivalent or better ROC curve was found for 78 of the 80 ANN's, or 97.5% of the time. By contrast, the current standard method results in statistically equivalent or better ROC curves for 41 of the ANN's, or 51.25% of the time. Table II summarizes these results separately for one- and two-hidden layer ANN's.

In practice, one would only select the single best ANN for a particular task. Table III summarizes the results when the single best ROC curve is selected for each data set. For each data set, we show: 1) the selected ANN topology, 2) the sample size of the training set, and 3) which method of ROC curve generation produced a higher AUC measure. We denote when the performances of the one-hidden layer network versus the two-hidden layer network are statistically significantly different. We also denote when the performances of the two ROC generation methods are statistically significantly different.

From Table III, we note the following. For 16 of the 40 experiments, our proposed method generates a statistically significantly better ROC curve than the current standard method. By contrast, the current standard method did not generate a statistically significantly better ROC curve than our proposed method for any data set examined in this work. Looking at the ANN topologies selected for each problem, we see that a network with two hidden layers resulted in a statistically significantly better ROC curve for only four of 40 experiments, or 10% of the time.

For the simulated data with overlapping Gaussian's, a practical upper bound on the AUC of an ROC curve can be established using a linear Bayes classifier in which the class-conditional probability density functions are assumed to have Gaussian distributions [14]. For example, an ROC curve generated from a linear Bayes classifier on the 2-D overlapping Gaussian data with 500 samples has an AUC of 0.987 851. Our proposed method generated an ROC curve with an AUC of 0.987 727, which is slightly less than the upper bound. The current standard method generated an ROC curve with an AUC of 0.966 757. These results are typical of the relative performance obtained for the simulated data with overlapping

TABLE III
RESULTS FOR 28 SIMULATED DATA SETS, 12 REAL APPLICATION DATA SETS. AN ASTERISK IN COLUMN 2 INDICATES THE PERFORMANCE OF THE SELECTED NETWORK IS STATISTICALLY SIGNIFICANTLY BETTER THAN THE OTHER NETWORK (I.E., BEST ONE-HIDDEN LAYER ANN VERSUS BEST TWO-HIDDEN LAYER ANN). AN ASTERISK IN COLUMN FOUR INDICATES THE SELECTED METHOD OF ROC CURVE GENERATION IS STATISTICALLY SIGNIFICANTLY BETTER THAN THE OTHER METHOD

Data Set	ANN Topology	# of Samples	Best Method
2-d Overlapping Gaussians	2 : 2 : 2 : 1	500	*Bias*
2-d Overlapping Gaussians	2 : 19 : 1	1000	*Bias*
2-d Overlapping Gaussians	2 : 28 : 24 : 1	1500	*Thresh*
2-d Overlapping Gaussians	2 : 1 : 1	2000	*Bias**
3-d Overlapping Gaussians	3 : 1 : 1	500	*Bias*
3-d Overlapping Gaussians	3 : 15 : 1*	1000	*Bias**
3-d Overlapping Gaussians	3 : 9 : 9 : 1*	1500	*Bias**
3-d Overlapping Gaussians	3 : 1 : 1*	2000	*Bias**
5-d Overlapping Gaussians	5 : 1 : 1	500	*Bias*
5-d Overlapping Gaussians	5 : 4 : 4 : 1	1000	*Bias*
5-d Overlapping Gaussians	5 : 1 : 1	1500	*Bias**
5-d Overlapping Gaussians	5 : 1 : 1	2000	*Bias**
2-d Gaussian and Uniform	2 : 5 : 1*	500	*Bias*
2-d Gaussian and Uniform	2 : 7 : 1*	1000	*Bias**
2-d Gaussian and Uniform	2 : 23 : 18 : 1*	1500	*Bias**
2-d Gaussian and Uniform	2 : 39 : 1*	2000	*Bias*
3-d Gaussian and Uniform	3 : 9 : 3 : 1	500	*Bias*
3-d Gaussian and Uniform	3 : 10 : 1*	1000	*Bias**
3-d Gaussian and Uniform	3 : 15 : 1*	1500	*Bias*
3-d Gaussian and Uniform	3 : 49 : 1	2000	*Bias*
5-d Gaussian and Uniform	5 : 6 : 1	500	*Thresh*
5-d Gaussian and Uniform	5 : 37 : 13 : 1	1000	*Thresh*
5-d Gaussian and Uniform	5 : 34 : 1	1500	*Bias*
5-d Gaussian and Uniform	5 : 23 : 1*	2000	*Bias*
2-d Exclusive-OR distribution	2 : 8 : 4 : 1	500	*Bias*
2-d Exclusive-OR distribution	2 : 5 : 1*	1000	*Bias**
2-d Exclusive-OR distribution	2 : 13 : 1*	1500	*Thresh*
2-d Exclusive-OR distribution	2 : 18 : 1*	2000	*Bias*
Heart disease diagnosis 1	6 : 66 : 2 : 1*	135	*Thresh*
Heart disease diagnosis 2	6 : 1 : 1	135	*Bias*
BUPA liver disorders 1	6 : 6 : 6 : 1	172	*Thresh*
BUPA liver disorders 2	6 : 1 : 1	171	*Tie*
Wisconsin Diag. Breast Cancer 1	10 : 2 : 2 : 1	284	*Bias*
Wisconsin Diag. Breast Cancer 2	10 : 2 : 2 : 1*	283	*Bias**
Pima Indians Diabetes 1	8 : 1 : 1*	384	*Bias**
Pima Indians Diabetes 2	8 : 2 : 2 : 1	384	*Bias**
Mammography Application A 1	7 : 1 : 1*	3719	*Bias**
Mammography Application A 2	7 : 98 : 1	1572	*Bias*
Mammography Application B 1	8 : 22 : 1	22736	*Bias**
Mammography Application B 2	8 : 15 : 1	25187	*Bias**

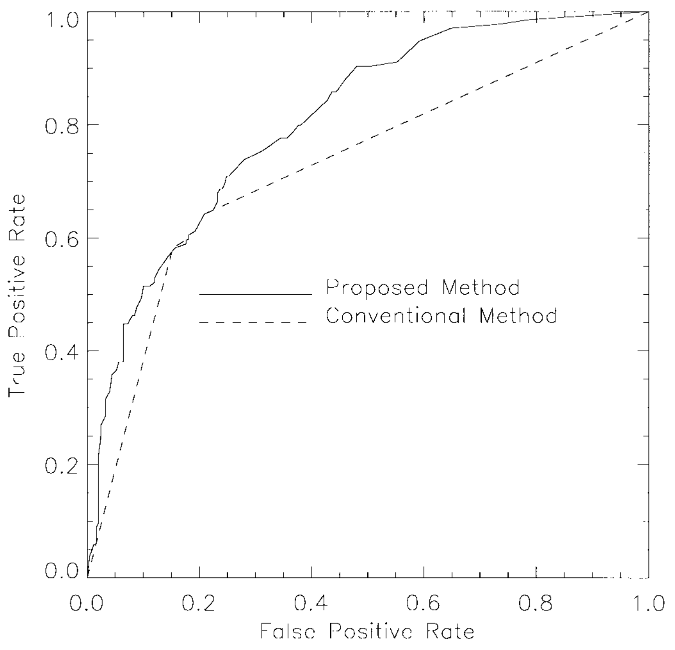


Fig. 5. ROC curves generated by both methods for one of the Pima Indians Diabetes data sets. Here, the current standard method was unable to generate any operating points with TP rates between 0%, 58.2%, or between 64.2% and 100%.

Gaussian's.

For the simulated data in which the target class has a Gaussian distribution and the nontarget class is uniformly distributed, it should be possible to generate a reasonably good ROC curve using a quadratic Bayes classifier in which the class-conditional probability density functions are assumed to have Gaussian distributions [14]. As an example, an ROC curve generated from a quadratic Bayes classifier on the 2-D data with 500 samples has an AUC of 0.987 603. Our proposed method generated an ROC curve with an AUC of 0.990 667, which is slightly better than the Bayesian classifier. The current standard method generated an ROC curve with an AUC of 0.988 119. A linear Bayes classifier generated an ROC curve with an AUC of 0.960 837. Again, these results are typical.

The current standard method occasionally had problems in finding a full range of operating points. As an example, using the current standard method for an ANN trained on one of the Pima Indians Diabetes data sets, operating points with only 8 distinct TP rates were found on the test data: 0%, 58.2%, 59.7%, 60.4%, 61.2%, 63.4%, 64.2%, and 100%. Using the proposed method, operating points corresponding to 72 distinct TP rates were obtained. These points run from 0% to 100% with a gap of at most 3% between successive operating points. Fig. 5 shows the ROC curves for this experiment. This example illustrates that it may not be possible for an ANN to operate near a desired sensitivity if the current standard method is used to generate operating points.

As we noted before, the results for a trained ANN are naturally dependent on the initial values of the weights. Therefore, we might expect slightly different test results should all these experiments be run again with different random initializations. Table IV summarizes the results of a second run of the 80 ROC experiments. This time the proposed method a statistically equivalent or better ROC curve was found for 76 of the ANN's, or 95.0% of the time. The current standard method results in statistically equivalent or better ROC curves for 48 of the ANN's, or 60.0% of the time. So, the current standard method faired slightly better than in the first round of experiments. Even so, the test results overwhelmingly favor the proposed method.

TABLE IV
SUMMARY OF SECOND RUN FOR ALL 80 ROC EXPERIMENTS

Results for ANNs with 1-Hidden Layer	# of times AUCs are stat. sig. different
Bias Scaling Method is better 29 times	15
Output Thresholding is better 11 times	2
Methods are equivalent 0 times	-
Results for ANNs with 2-Hidden Layers	# of times AUCs are stat. sig. different
Bias Scaling Method is better 31 times	17
Output Thresholding is better 8 times	2
Methods are equivalent 1 time	-

VI. SUMMARY AND CONCLUSION

Methods of generating ROC curves for ANN classifiers in a two-class problem are examined. Two methods are compared. These methods are 1) varying a threshold on the output node, and 2) scaling the bias weight for selected first hidden layer nodes. Varying a threshold on the output node is the current standard method [3]–[6]. Scaling the bias input weight for selected first hidden layer nodes is a new method proposed here.

Our proposed method involves the construction of a lookup table which contains a sequence of scale factors for the bias weights of each first hidden layer node. The lookup table is used as a "sensitivity dial" which facilitates the easy selection of an operating point for an ANN classifier, and thereby permits reliable classification at operating points other than the one to which the ANN naturally trained.

Based on our experimental results, we suggest the following methodology for utilizing ANN's in diagnostic applications in which it is necessary to generate an ROC curve. First, select the "best" ANN topology for the application. Our results suggest that approximately 90% of the time an network with a single hidden layer will be sufficient. Since one may avoid testing numerous two-hidden layer ANN configurations, a considerable reduction in computational effort can be realized if a trial and error approach to ANN topology selection is utilized. Since the current standard method of generating ROC curves for an ANN is simple to implement, one may want to try this approach first. However, our test results would indicate that a statistically significantly better ROC curve can be generated using the proposed method between 40% and 50% of the time. Additionally, the current standard method may, on occasion, have problems generating operating points over a large range of sensitivity.

In the process of developing this work, several other methods of ROC generation were considered and eventually discarded. One approach attempted to create ROC points by repeating target or nontarget training samples some extra number of times in extra training epochs. Another approach attempted to create ROC points by an additional training epoch in which the error value for training samples in different classes was scaled by some factor prior to backpropagation. Neither of these approaches yielded results as good as the method of adjusting the first hidden layer bias values.

We have not considered attempts to generate ROC curves by repeatedly training an ANN to the different operating points. Changing the ANN architecture for each ROC point, manipulating the training set, or weighting the different error types are all methods that would require the ANN to be trained "from scratch" many times to derive each operating point. While these methods may be feasible, they are not generally done in practice because they are potentially problematic and time-consuming. We would prefer to generate an ROC curve from an already trained classifier by manipulating some underlying parameter(s) of a classifier. The manipulation of classifier parameters will in turn move a decision boundary in feature space between the two classes and result in a new sensitivity/specificity tradeoff. Thus, our proposed method and the current standard method both start with a decision boundary found via training, and directly manipulate the decision boundary in order to obtain new operating points. This is not the case if we retrain the ANN each time from some random initialization to find a set new decision boundaries for each operating point.

References

[1] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, vol. 1. Reading, MA: Addison–Wesley, 1992.

[2] N. Asada, K. Doi, H. MacMahon, S. Montner, M. L. Giger, C. Abe, and Y. Wu, "Neural network approach for differential diagnosis of interstitial lung diseases," in *Proc. SPIE Conf. Medical Imaging IV: Image Processing*, Newport Beach, CA, Feb. 1990.

[3] I. N. Bankman, V. G. Sigillito, R. A. Wise, and P. L. Smith, "Feature-based detection of the K-complex wave in the human electroencephalogram using neural networks," *IEEE Trans. Biomed. Eng.*, vol. 39, no. 12, pp. 1305–1310, 1992.

[4] J. M. Boone, V. G. Sigillito, and G. S. Shaber, "Neural networks in radiology: An introduction and evaluation in a signal detection task," *Med. Phys.*, vol. 17, no. 2, pp. 234–241, Mar/Apr 1990.

[5] G. W. Gross, J. M. Boone, V. Greco-Hunt, and B. Greenberg, "Neural networks in radiologic diagnosis—II: Interpretation of neonatal chest radiographs," *Investigat. Radiol.*, vol. 25, pp. 1017–1023, 1990.

[6] Y. Wu, K. Doi, M. L. Giger, and R. M. Nishikawa, "Computerized detection of clustered microcalcifications in digital mammograms: Applications of artificial neural networks," *Med. Phys.*, vol. 19, no. 3, pp. 555–560, May/Jun 1992.

[7] K. Knight, "Connectionist ideas and algorithms," *Communicat. ACM*, vol. 33, no. 11, pp. 59–74, 1990.

[8] K. S. Woods, J. L. Solka, C. E. Priebe, C. C. Doss, K. W. Bowyer, and L. P. Clarke, "Comparative evaluation of pattern recognition techniques for detection of microcalcifications," in *Proc. SPIE/IS&T Symp. Electronic Imaging Science and Technology*, San Jose, CA, Jan. 31–Feb. 4, 1993, vol. 1905, pp. 841–852.

[9] W. P. Kegelmeyer, Jr., and M. C. Allmen, "Dense feature maps for detection of calcifications," in *Digital Mammography: Proc. 2nd Int. Workshop Digital Mammography, International Congress Series*, A. G. Gale, S. M. Astley, D. R. Dance, and A. Y. Cairns, Eds., 1994, vol. 1069, pp. 3–12.

[10] C. E. Metz, "ROC methodology in radiologic imaging," *Investigat. Radiol.*, vol. 21, pp. 720–733, 1986.

[11] J. A. Swets, "ROC analysis applied to the evaluation of medical imaging techniques," *Investigat. Radiol.*, vol. 14, 109–121, 1979.

[12] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve," *Radiol.*, vol. 143, pp. 29–36, 1982.

[13] J. A. Hanley and B. J. McNeil, "A method of comparing the areas under receiver operating characteristic curves derived from the same cases," *Radiol.*, vol. 148, pp. 839–843, 1983.

[14] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.