Multiobjective Genetic Optimization of Diagnostic Classifiers with Implications for Generating Receiver Operating Characteristic Curves

Matthew A. Kupinski,* *Student Member, IEEE* and Mark A. Anastasio, *Student Member, IEEE*

***Abstract*— It is well understood that binary classifiers have two implicit objective functions (sensitivity and specificity) describing their performance. Traditional methods of classifier training attempt to combine these two objective functions (or two analogous class performance measures) into one so that conventional scalar optimization techniques can be utilized. This involves incorporating *a priori* information into the aggregation method so that the resulting performance of the classifier is satisfactory for the task at hand. We have investigated the use of a niched Pareto multiobjective genetic algorithm (GA) for classifier optimization. With niched Pareto GA's, an objective vector is optimized instead of a scalar function, eliminating the need to aggregate classification objective functions. The niched Pareto GA returns a set of optimal solutions that are equivalent in the absence of any information regarding the preferences of the objectives. The *a priori* knowledge that was used for aggregating the objective functions in conventional classifier training can instead be applied post-optimization to select from one of the series of solutions returned from the multiobjective genetic optimization. We have applied this technique to train a linear classifier and an artificial neural network (ANN), using simulated datasets. The performances of the solutions returned from the multiobjective genetic optimization represent a series of optimal (sensitivity, specificity) pairs, which can be thought of as operating points on a receiver operating characteristic (ROC) curve. All possible ROC curves for a given dataset and classifier are less than or equal to the ROC curve generated by the niched Pareto genetic optimization.**

***Index Terms*—Diagnostic classifiers, genetic algorithms, multiobjective optimization, ROC analysis.**

I. Introduction

THE task in medical diagnostic decision making is typically one of employing multiple features to classify an observation as normal or abnormal. A radiologist may, for example, note the size, shape and margin sharpness of a potential breast lesion in a mammogram and somehow use this information to determine whether a cancer is present. In computer-aided diagnosis (CAD) [1]–[3], computers take features extracted from medical images and determine whether pathology is present by using automated classifiers [4], [5]. It is well known that the optimal method for classifying is to use the likelihood ratio or any monotonic transformation of the likelihood ratio as the discriminant function [4]. The goal in training a diagnostic classifier is to employ a limited dataset to determine the parameters of the classifier such that it approximates the likelihood ratio decision rule. For the most part, these classifiers work in a similar fashion. A dataset of features extracted from both normal (without disease) and abnormal (with disease) images is used for determining the classifier parameter values, or for "training" the classifier, so that it correctly classifies future datasets of unknown pathology.

Classifier training can be viewed as an optimization problem where the quantity to be maximized is the performance of the classifier on an independent dataset. There are, however, numerous problems with representing classifier performance by a single (scalar) objective function, which is needed so that one can use a scalar optimizer [6], [7]. Binary classifiers [4] have, in essence, two implicit objective functions: one describing how well they classify the abnormal cases (sensitivity) and one describing how well they classify the normal cases (specificity). These two objective functions are noncommensurable, implying that it may not be possible to simultaneously improve both the sensitivity and specificity. Traditional methods of classifier training attempt to combine these two objective functions (or two analogous class performance measures) into a single scalar objective function that permits the use of conventional (scalar) optimization techniques [8]. A drawback to this approach is that the proper way of aggregating the objective functions is usually unknown. There are, in fact, an infinite number of ways of mapping two objective functions to a single scalar function. Even when *a priori* information about the relative importance of the two objective functions is available, it is not always clear how to incorporate it in the aggregating approach to objective function design. Sometimes, numerous *ad hoc* combination functions are tried until a suitable objective function is found [8]. Most classifiers do not aggregate sensitivity and specificity directly. Artificial neural networks, for example, typically employ a sum-of-squares error function [5], which can still be thought of as a sum of two noncommensurable objectives, i.e., one objective is to map abnormal observations to a value close to

Manuscript received July 21, 1998; revised June 30, 1999. This work was supported in part by the US Army Medical Research and Materiel Command under grants DAMD 17-96-1-6058 and DAMD 17-97-1-7202 and in part by the USPHS under Grants CA24806 and RR11459. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was A. Burgess. *Asterisk indicates corresponding author*.

*M. A. Kupinski is with Kurt Rossmann Laboratories, Department of Radiology, The University of Chicago, Chicago, IL 60637 USA.

M. A. Anastasio is with the Graduate Programs in Medical Physics, Department of Radiology, The University of Chicago, Chicago, IL 60637 USA.

Publisher Item Identifier S 0278-0062(99)08511-0.

0278–0062/99$10.00 © 1999 IEEE

one and the other objective is to map normal observations to a value close to zero.

Genetic algorithms (GA's) [9] have been applied to many diagnostic and classification problems [8], [10]–[15]. A conventional GA, however, is a scalar optimization technique. It thus possesses the undesirable features of an aggregating-based approach. One method of avoiding this is to adopt a multiobjective approach [16], [17] to the optimization problem. In a multiobjective optimization approach, the objective function is vector valued and the independent objectives (sensitivity and specificity) are optimized simultaneously. Thus, the need to aggregate the independent objective functions is removed. Unlike a scalar optimization that returns a single solution, the solution to the multiobjective optimization problem is a set of solutions called the Pareto-optimal set. The Pareto-optimal set is defined as the set of solutions for which no other solution exists that is better in both objectives. In the absence of any preference information about the objectives, the members of the Pareto-optimal set are equally valid solutions to the optimization problem; no other solutions exist that are better in all of the objectives. In the context of diagnostic classifier optimization, the members of the Pareto-optimal set correspond to operating points on an optimal receiver operating characteristic (ROC) curve, whose performances describe the limiting sensitivity–specificity tradeoffs that the classifier can provide for the given training dataset. Conventional nonevolutionary optimization techniques have not been successfully extended to the multiobjective case because they are not designed to operate on multiple solutions. Because GA's are population based, they have formed the basis of several multiobjective optimization techniques, collectively referred to as multiobjective GA's (MOGA's) [16]–[19].

In this paper, we investigate the application of a MOGA called a niched Pareto GA (NP-GA) for optimizing the performance of two popular diagnostic classifiers. The paper is organized as follows. Section II contains a general introduction to automated classifiers and a brief description of the NP-GA. Section III describes the two classifiers that were studied and it describes how the NP-GA was employed to train them. The results of the two optimizations are presented in Section IV. Sections V and VI contain a discussion of the results and a summary of the advantages and drawbacks of the proposed approach to diagnostic classifier training and ROC curve generation.

II. Background

A. Automated Classifiers

An automated binary classifier separates two classes of observations (e.g. images) and assigns new observations to one of the two classes. In this paper, we will label the two classes as normal (no disease evident) and abnormal (indicative of disease), denoted by π_n and π_a, respectively. Certain characteristics of the observations, called features, are used in making the classification decision. The set of features corresponding to an observation can be expressed by a vector $\vec{x} = [x_1, x_2, \ldots, x_p]$. In order for the classifier to be trained,

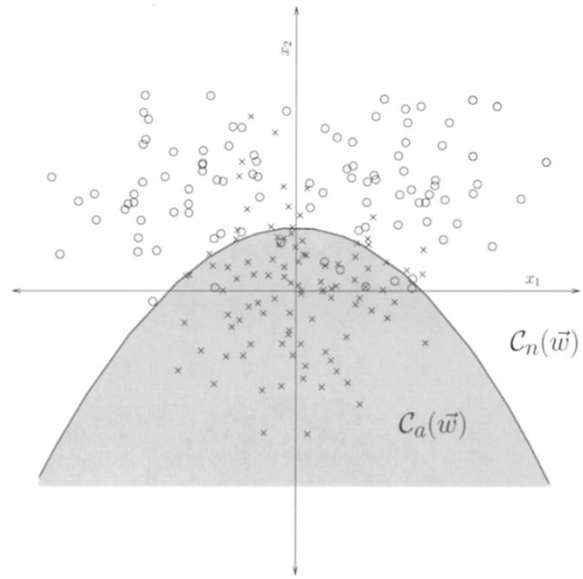

Fig. 1. The job of an automated classifier is to partition the multidimensional feature space into two partitions $\mathcal{C}_a(\vec{w})$ belonging to class π_a and $\mathcal{C}_n(\vec{w})$ belonging to class π_n. These partitions $\mathcal{C}_n(\vec{w})$ and $\mathcal{C}_a(\vec{w})$ are shown by the shaded and unshaded regions. The two classes π_a and π_n are represented by different symbols (x's and o's). The decision boundary is denoted by the solid line separating the shaded from the unshaded region.

we start with a dataset of known pathology, called the training dataset. A graphical depiction of an automated classifier for a two-feature example $(p = 2)$ is shown in Fig. 1. The $\{x_1, x_2\}$ space spanned by the feature vectors is denoted by $\mathcal{S}$. An automated classifier uses a parameter vector $\vec{w}$ to partition this space into the sets $\mathcal{C}_n(\vec{w})$, the set of observations that belong to class π_n and $\mathcal{C}_a(\vec{w})$, the set of observations belonging to class π_a. The parameters $\vec{w}$ of a classifier can represent, for example, the weights of an artificial neural network (ANN) or the threshold values in a rule-based classifier. For a fixed $\vec{w}$, $\mathcal{C}_n(\vec{w}) \cup \mathcal{C}_a(\vec{w}) = \mathcal{S}$ and $\mathcal{C}_n(\vec{w}) \cap \mathcal{C}_a(\vec{w}) = \emptyset$.

Given a measurement $\vec{x}$, the classifier assigns $\vec{x}$ to class π_n if $\vec{x} \in \mathcal{C}_n(\vec{w})$ or to class π_a if $\vec{x} \in \mathcal{C}_a(\vec{w})$. The probability that an observation belonging to class π_a is correctly classified is referred to as the sensitivity of the classifier, denoted by $Sens(\vec{w})$. Similarly, the probability that an observation is correctly classified as belonging to class π_n is referred to as the specificity of the classifier, denoted by $Spec(\vec{w})$. Note that both the sensitivity and specificity of the classifier depend explicitly on the choice of $\vec{w}$ and implicitly on the underlying distribution of the normal and abnormal observations. The sensitivity is a measure of how well the classifier performs on abnormal cases, whereas the specificity is a measure of how well a classifier performs on normal cases. In practice, the fraction of class π_a observations that are correctly classified is used as an estimate of $Sens(\vec{w})$. Likewise, the fraction of class π_a observations that are correctly classified is used as an estimate of $Spec(\vec{w})$.

A popular construct used for describing the performance of a diagnostic classifier is the ROC curve [6], [7], [20], [21]. A ROC curve is generated by varying the value of one (or more) of the components of the parameter vector $\vec{w}$, and plotting the corresponding $Sens(\vec{w})$ and $Spec(\vec{w})$ values. For example, the

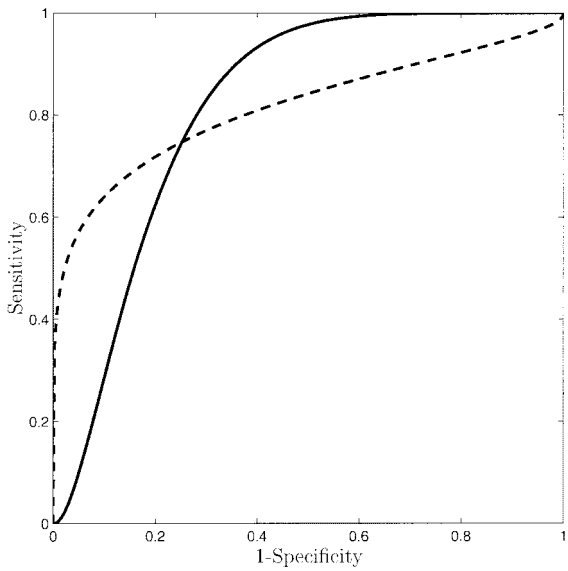

Fig. 2. The two ROC curves have equal A_z values, but, depending upon the relative preferences concerning the sensitivity or specificity of the detection task, one curve is typically preferred over the other.

output threshold is usually varied to generate a ROC curve for ANN's [22]. Traditionally, the classifier is trained prior to the generation of the ROC curve [22], [23]. In this situation, all but one point on the ROC curve represent operating points other than the one to which the classifier was naturally trained. A ROC curve that was generated with the same dataset that was used to train the classifier is referred to as a consistency ROC curve. A validation ROC curve is obtained when the curve is generated from an independent data set and represents an unbiased estimate of classifier performance [24]. Two typical ROC curves are shown in Fig. 2. The area under a ROC curve, or A_z, is an accepted way of comparing overall classifier performance [6], [7], [20], [21]. Two curves may have equal A_z values, as shown in Fig. 2. However, one of the curves will typically be preferred over the other, depending upon the relative preference of the sensitivity and the specificity needed for the task at hand.

For certain types of classifiers, such as rule-based systems [3], [25], it may not be clear how $\vec{w}$ should be varied to sweep out the ROC curve that best represents the sensitivity–specificity tradeoffs that are achievable by the classifier on the specified dataset. The ROC curves generated by varying different sets of components of $\vec{w}$ will generally be different, representing different sensitivity–specificity tradeoffs that are possible. In this work, we demonstrate that this ambiguity can be removed if one uses the performances of the solutions returned by a multiobjective optimization of the classifier to define the ROC curve.

B. The Niched Pareto GA

We have implemented a multiobjective optimization technique called an NP-GA, which is described in detail by Horn *et al*. [26]. Other types of MOGA's have been proposed and are described in [18]. The NP-GA can be viewed as a conventional (scalar) GA that uses a modified tournament selection mechanism and ranking scheme. Readers not familiar with GA's may consult [9]. In the remainder of this section we review the NP-GA proposed by Horn *et al*. [26].

In order to directly address the multiobjective nature of the optimization problem, NP-GA's employ the concept of dominance. A solution to the optimization problem is called nondominated if there is no solution superior to it in all objectives. It is the goal of the NP-GA to discover the set of all nondominated solutions, referred to as the Pareto-optimal set, all of which are considered to be equally valid solutions to the problem in the absence of any *a priori* information about the relative merits of the different objectives. If a solution is not nondominated, it is referred to as being dominated. A nondominated solution is said to dominate a dominated solution. Equivalence classes of dominated solutions are formed by grouping them according to the number of solutions that dominate them.

This grouping of solutions into distinct classes establishes a partial order on the set of all solutions that is used to determine rank. We assume that the Pareto-optimal set corresponds to equivalence class zero and that all other solutions have an equivalence class greater than zero. The rank of a particular solution is then equal to its equivalence class number. This ensures that solutions within the same equivalence class have the same rank, which reflects the fact that solutions within the same class are equally good in the absence of any other information.

To perform selection, the NP-GA uses a modified tournament selection method. In a scalar GA, tournament selection is one of the methods commonly used for choosing a subset of solutions in the current generation to be placed in the following generation. Implicit in its formulation is the assumption that there exists a single solution to the optimization problem; diversity among solutions in the population will be lost after a certain number of generations. This is undesirable in a multiobjective optimization where we wish to discover all of the members of the Pareto-optimal set, not simply a single solution. To circumvent this difficulty, Horn *et al*. proposed the use of a Pareto domination tournament in conjunction with a form of fitness sharing called equivalence class sharing. A Pareto domination tournament is a modified conventional tournament selection method that uses the concept of dominance to determine the winner of the tournament. First, t_{dom} randomly selected solutions are compared and the solution with the highest rank wins (is carried over to the next generation). The rank, being based on the concept of dominance, incorporates the multiobjective nature of the problem into the selection mechanism. For situations when a certain tournament size provides insufficient domination pressure, the size of the tournament (t_{dom}) can be increased.

When two or more solutions in a tournament belong to the same equivalence class (i.e., have the same rank), there will not be a clear winner. A winner cannot simply be chosen at random because genetic drift will cause the population to converge to a localized region of the Pareto-optimal set, thus obscuring other potential solutions to the optimization problem. Instead, a form of fitness sharing, called equivalence class sharing, is employed to determine the winner of a tied tournament. In

equivalence class sharing, the winner of a tied tournament is the solution that has the smallest niche count. The niche count estimates the density of solutions in a localized region (niche) around a given solution. As described in [26], the niche count m_i for the ith solution is given by

$$m_i = \sum_{j \in Pop} s(d_{ij}) \tag{1}$$

where d_{ij} is the distance (in objective space) between solutions i and j and $s(\,)$ is the so-called sharing function given by $s(d) = 1 - d/\sigma_{\text{share}}$ for $d \leq \sigma_{\text{share}}$ and $s(d) = 0$ otherwise. Here, σ_{share} is called the niche radius, which represents the maximum distance between solutions that will result in an increase in their niche counts. By employing fitness sharing in this way, the Pareto-optimal set is more likely to be uniformly sampled, thus providing a more diverse set of potential solutions to the optimization problem from which the user can choose.

III. Methods

We trained a linear classifier and an ANN by using both conventional optimization techniques and the NP-GA. Two-dimensional (2-D) exclusive-OR data [27], sampled from the density functions shown in Fig. 3, were used for this study, because classifiers typically have difficulty in adequately classifying both classes of data for this problem. Two-dimensional isotropic standard normal distributions with mean (μ_{x_1}, μ_{x_2}) and variance one were sampled in the four regions of the exclusive-OR problem. The normal class (dashed lines in Fig. 3) occupied the regions centered at (1.3,1.3) and (−1.3,−1.3). The abnormal class occupied the regions centered at (1.3,−1.3) and (−1.3,1.3). A total of 100 normal and 100 abnormal samples were generated for training data. An additional 10 000 normal and 10 000 abnormal samples were generated for testing the classifiers after they had been trained. The performances of the conventionally optimized and NP-GA optimized classifiers were evaluated on both the training and the testing datasets.

A. NP-GA Implementation

The NP-GA was employed to simultaneously maximize the sensitivity and specificity of a linear classifier and an ANN with a single hidden layer. The value of each component of $\vec{w}$ was restricted to remain within a maximum and minimum value, determined prior to the optimization. A binary representation of the chromosomes [9] was utilized so that each real-valued parameter in $\vec{w}$ was encoded by a binary number of fixed length. The range of each component of $\vec{w}$ and the length of its binary representation determined that parameter's floating-point precision. The encoding was accomplished by linearly scaling the floating point number using its specified range to an integer between zero and $2^n - 1$ where n is the number of bits. Standard single-point crossover and standard mutation were employed as the genetic operations [9]. The rates of the genetic operations were determined empirically by performing multiple optimizations. A crossover rate of 30% and a mutation rate of 5% were found suitable for the problems

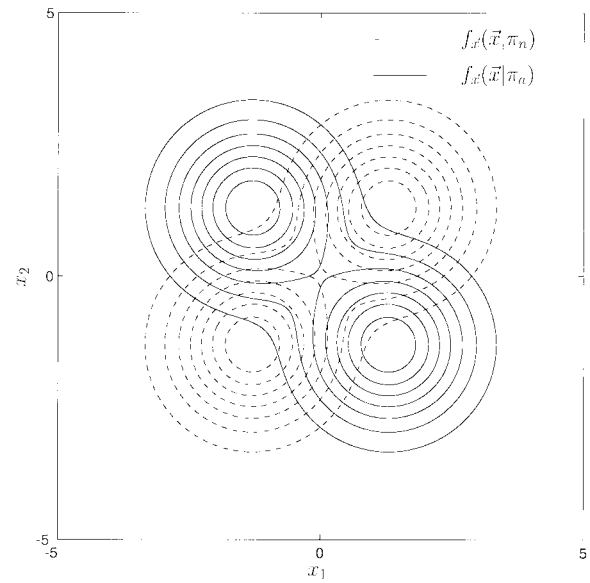

Fig. 3. Contour diagrams of the two density functions that make up the exclusive-OR problem. The abnormal class (solid lines) occupies the upper left and lower right quadrants, whereas the normal class (dashed lines) occupies the upper right and lower left quadrants.

studied. A t_{dom} value of four and a σ_{share} value of 0.1 (or 10% of the range of each objective) were also found to work well for the optimization problems discussed in this paper. A discussion of these parameter settings is presented later.

B. Classifiers

A linear classifier attempts to separate the two classes of observations by using a linear decision boundary. We employed logistic discriminants [5] in order to implement this classification. A logistic discriminant projects the data onto a decision variable, and then a threshold is applied for determining whether a given observation belongs to π_a or π_n. The abnormal set for a logistic discriminant with parameter vector $\vec{w}$ is defined as

$$\mathcal{C}_a(\vec{w}) = \{\vec{x} : g(\vec{x}'\vec{w}^{\mathrm{T}}) \geq 0.5\} \tag{2}$$

where $\vec{x}' = [x_1, x_2, \ldots, x_p, -1] = [\vec{x}, -1]$ and $g(\,)$ is a sigmoidal function with an output bound between zero and one [5]. The normal set is defined as $\mathcal{C}_n(\vec{w}) = \mathcal{S} - \mathcal{C}_a(\vec{w})$. The conventional method for generating a ROC curve for a logistic discriminant is to vary the final parameter in the vector $\vec{w}$, which results in a translation of the decision boundary.

The NP-GA was used to optimize the parameters of a logistic discriminant so as to work with the exclusive-OR data described previously. All three components (for 2-D problems, $\vec{w}$ has three components) of the parameter vector $\vec{w}$ were allowed to range between −3 and 3. With a population size of 500 solutions, we ran the NP-GA for a total of 100 generations. Conventional logistic discriminant training, as described in [5], was employed to compare with the NP-GA results.

An ANN is a set of connected nodes that is loosely based on the human neuron system [5], [27]–[30]. For classification purposes, an ANN can be thought of as a mapping function that uses the weights $\vec{w}$ to map an input vector $\vec{x}$ to a scalar

quantity to which a threshold is applied to determine whether $\vec{x}$ belongs to class π_a or π_n. Unlike logistic discriminants, an ANN can separate the two classes of observations using a nonlinear decision boundary. The abnormal set of observations for an ANN using the weights $\vec{w}$ is given by

$$\mathcal{C}_a(\vec{w}) = \{\vec{x} : h(\vec{x}; \vec{w}) \geq 0.5\} \tag{3}$$

where $h(\vec{x}; \vec{w})$ represents the nonlinear mapping of the input features to the single output value bound between zero and one.

We applied the NP-GA to optimize an ANN on the exclusive-OR data. A two-layered ANN with two inputs, two hidden units, and one output unit was employed. This corresponded to a total of nine parameters to be optimized. The magnitudes of the weights were forced to lie between -5 and 5 in order to simplify the optimization task and to regularize the problem somewhat, because large weight values represent complex decision boundaries [28]. A population size of 3000 solutions was run for a total of 100 generations for this study. Conventional error-backpropagation ANN training [5], [27], [29], [30] was also employed numerous times, using different initial conditions. A comparison of the performances of the NP-GA results with the best conventional results will be shown, along with a comparison of the NP-GA performances with a conventional optimization that was trapped in a local minimum. The conventional ROC curves were generated by varying the output bias weight value, which corresponds to one component of $\vec{w}$. This is equivalent to varying the neural network output threshold. It should be noted that Woods and Bowyer [23] studied the effect of varying weight values other than the output bias weight in generating ROC curves. Their study concluded that varying a subset of the weights can produce better ROC curves than the ROC curves produced by varying the output threshold, as is conventionally done. By applying the NP-GA to ANN's, however, we are effectively allowing all the weights to vary when generating the ROC curve, including both the output threshold and the hidden layer bias weights studied in the Woods and Bowyer work.

IV. Results

A. Linear Classifier

Fig. 4 shows the performances of the nondominated solutions returned by the NP-GA and the ROC curve that resulted from the conventional training, generated by thresholding the output value. The operating points obtained by the NP-GA are seen to be better than the corresponding operating points on the conventional ROC curve in the high-sensitivity region. Fig. 5 demonstrates the same behavior when the NP-GA solutions and the conventional solution are evaluated on the independent data set. This is evidence that the performance improvement achieved by the NP-GA training was not simply a result of over-training. However, because the training data were sparse between the four regions of the exclusive-OR data, a few of the solutions returned by the NP-GA show slight signs of overfitting when tested on the 20 000 testing samples, as is demonstrated by the fact that a few solutions are dominated when evaluated on the test set. The majority of the solutions, however, do not show signs of overtraining.

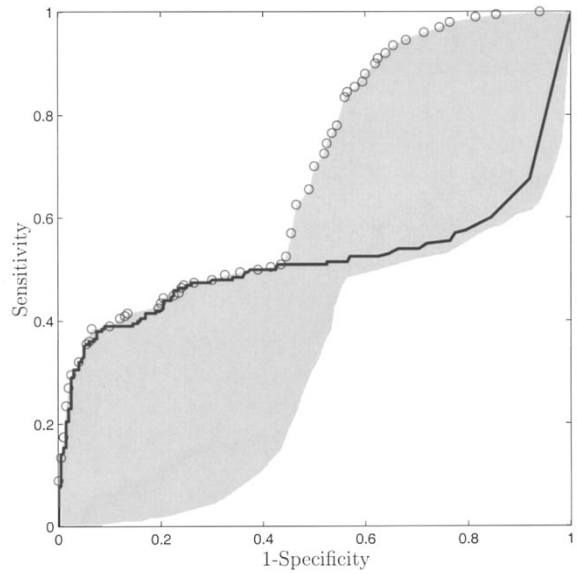

Fig. 4. Consistency results of the logistic discriminant training using exclusive-OR training data. The circles represent the performances of the nondominated solutions returned by the NP-GA based training. The solid line is the conventional ROC curve produced by varying the output threshold value of the logistic discriminant after it was trained using a scalar optimization technique. The shaded region shows the performances achievable by all possible weight vectors $\vec{w}$.

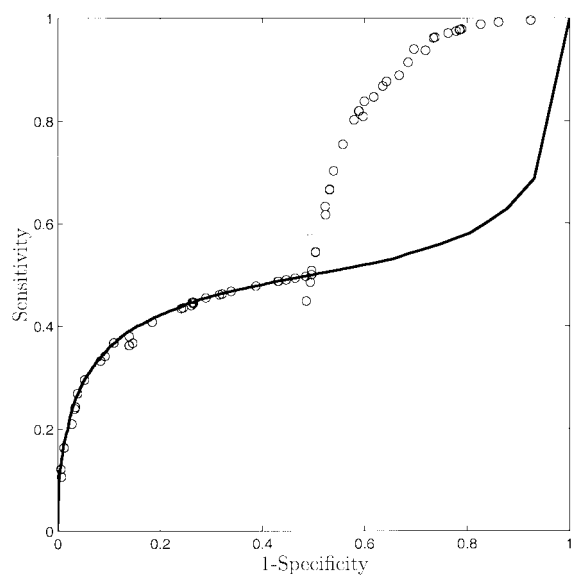

Fig. 5. Validation results of the logistic discriminant training for 20 000 samples from the exclusive-OR data distribution to evaluate the performances. The circles represent the performances of the nondominated solutions returned by the NP-GA based training. The solid line is the conventional ROC curve produced by varying the output threshold value of the logistic discriminant after it was trained using a scalar optimization technique.

The ROC curve for the conventionally trained logistic discriminant was generated by varying the output threshold (final parameter in $\vec{w}$) and plotting the corresponding sensitivity and specificity values. Fig. 6 shows the decision boundaries at various output thresholds for the conventionally trained logistic discriminant. Decision boundaries corresponding to

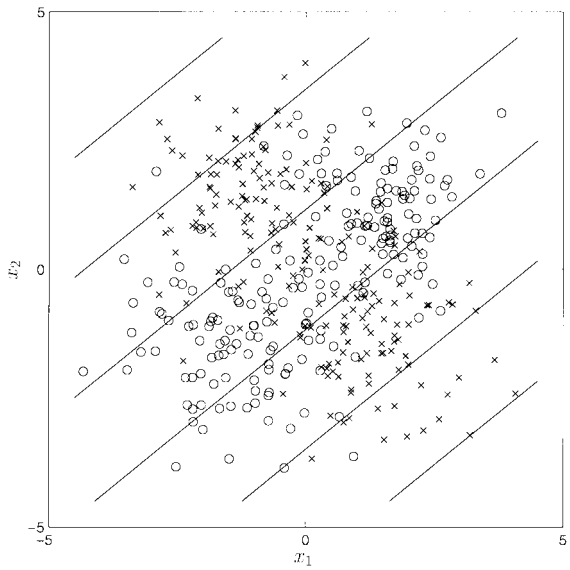

Fig. 6. An explanation of why the conventionally trained logistic discriminant only performs well in the low-sensitivity region. The decision boundaries corresponding to different output threshold values of the discriminant are shown superimposed on the data distribution. The o's represent normal signals and the x's represent the abnormal signals. The abnormal region is to the left of each decision boundary and the normal region is to the right of each decision boundary. When the threshold value is varied, the decision boundary is simply translated with its orientation remaining fixed. By analyzing the sensitivities and specificities for each decision boundary, one can generate the conventional ROC curve shown in Fig. 4. In order for the classifier to perform well in the high-sensitivity region, the decision boundaries would have to be rotated by 90°, which would result in the classifier performing poorly in the low-sensitivity region.

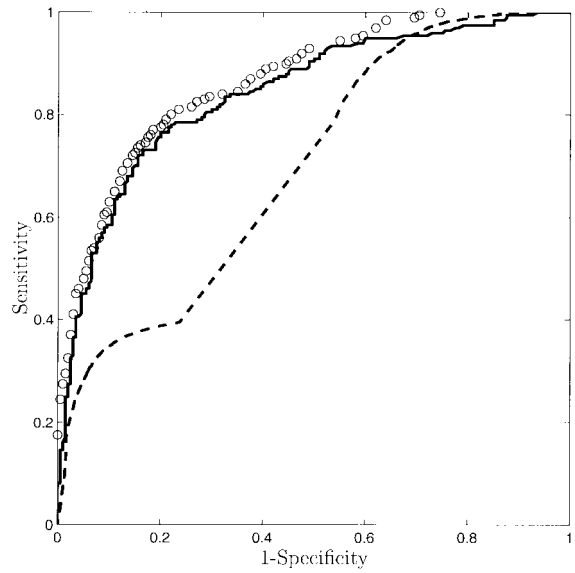

Fig. 7. Consistency results of the ANN training, using exclusive-OR training data. The circles represent the performances of the nondominated solutions returned by the NP-GA based training. The solid line is the conventional ROC curve produced by varying the output threshold value of the ANN after it was trained using a scalar optimization technique. The dashed line represents the result of a conventionally trained ANN trapped in a local minimum. The conventional training became trapped in local minima in approximately 30% of the conventional optimizations performed.

different threshold values are seen to be parallel. Because of this, the classifier only performs well in the low-sensitivity region. If, however, the decision boundaries were rotated by 90° to those shown in Fig. 6, the classifier would, instead, perform well in the high-sensitivity region. The advantage of the NP-GA is that, at different ROC operating points, the orientation of the decision boundary can be different. Thus, the NP-GA trained logistic discriminant can perform optimally in both the high- and low-sensitivity regions. This is because, with the NP-GA, all components of $\vec{w}$ are effectively allowed to vary when generating the ROC curve, rather than just varying the value of one of the parameters and keeping the other two fixed.

B. Artificial Neural Network

The performances of the NP-GA results on the 200 training samples is shown in Fig. 7. The best conventional ANN optimization ROC curve, created by varying the output threshold, is also shown in Fig. 7. The NP-GA result is either equal to or better than the best conventional result at all points. The differences are small in most regions, but substantial in the very-high-sensitivity region of the ROC curve. No regularization techniques were applied to the conventional optimization. Therefore, one would typically be concerned about overtraining. Fig. 8 shows the validation ROC curves generated by applying the optimized results to the 20 000 testing samples. Again, the NP-GA result is closely matched

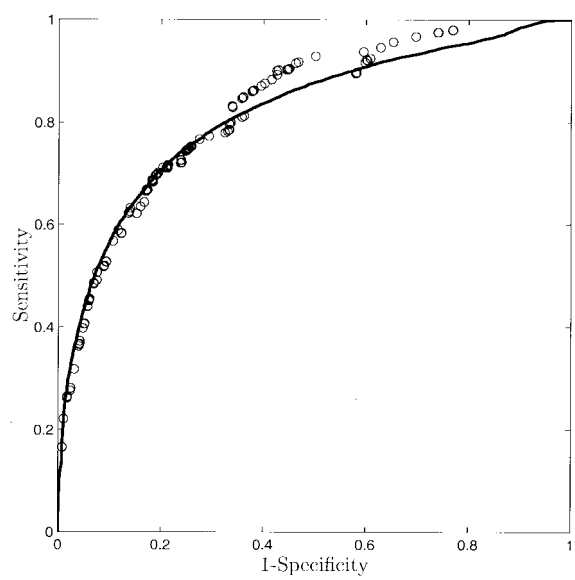

Fig. 8. Validation results of the ANN training on 20 000 samples from the exclusive-OR data distribution to evaluate the performances. The circles represent the performances of the nondominated solutions returned by the NP-GA based training. The solid line is the conventional ROC curve produced by varying the output threshold value of the ANN after it was trained using a scalar optimization technique.

with the conventional result at most places in the ROC space, except in the high-sensitivity region where the NP-GA result is noticeably better than the conventional result. Overtraining was not a noticeable problem in both of these optimizations because the structure of the ANN was limited (two hidden nodes) in both runs and the parameter range of the NP-GA was limited as well.

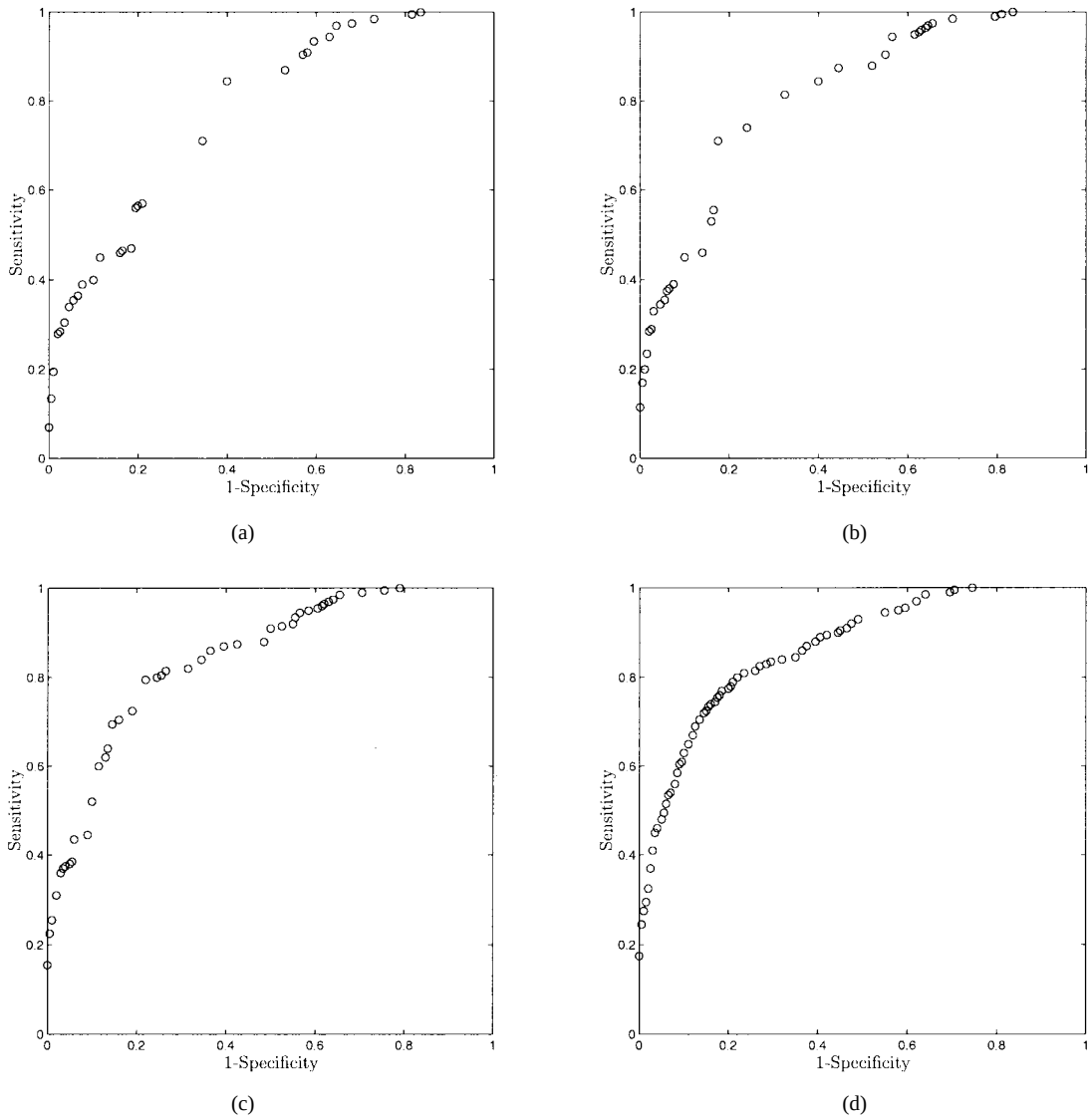

Fig. 9. Convergence of the NP-GA for the ANN training example as described in the text. (a)–(d) Performances of the nondominated solutions at generation numbers 2, 5, 13, and 100, respectively. As the generation number increases, the loci of operating points migrate upward and to the left.

Local minima often plague conventional ANN optimizations. We found that, depending upon the initial staring point, our ANN converged to local minima about 30% of the time, as was evident by comparing the ROC curves of the different ANN optimizations. The NP-GA never had a problem with local minima. Fig. 7 also shows the performance of the conventional result that resided in a local minimum in the parameter space (dashed line). The NP-GA result is substantially better at almost all points in ROC space.

C. NP-GA Performance

We conducted experiments to analyze the behavior of the NP-GA and verify that our choice of NP-GA operating parameter settings was reasonable. Fig. 9 demonstrates the convergence of the nondominated set when the ANN was trained, using the previously described training data and operating parameter settings. Fig. 9(a)–(d) shows the performances of the nondominated solutions evaluated on the training data at generations 2, 5, 13, and 100, respectively. It can be seen that the loci of operating points migrate upward and to the left as the generation number increases. Beyond 100 generations, the loci of operating points remain approximately constant, demonstrating that the NP-GA had converged to a stable set of solutions. It should also be noted that the relatively high density of the operating points returned by the NP-GA indicates that the nondominated set of solutions was adequately sampled.

Although the data described above demonstrate that the NP-GA converged when training the ANN, we do not know whether the final set of solutions represents the best possible set of solutions (i.e., the Pareto-optimal set). To verify this, one would have to evaluate the performances of all the possible combinations of parameter values of the ANN, which is not a computationally tractable problem with current computer technology. We can, however, compute this for the linear classifier because it possesses only three free parameters. The shaded region in Fig. 4 shows the operating points achievable by all possible parameter settings for the linear classifier.

Because most of the operating points returned by the NP-GA lie on the upper left boundary of the shaded region, we can conclude that, for this example, the NP-GA was successful at converging to the Pareto-optimal set.

As was noticed in [19], we observed that the size of the Pareto dominant tournament (t_{dom}) significantly affected the convergence behavior of the NP-GA. Fig. 10 shows the operating points returned by two separate applications of the NP-GA to the ANN training. The upper set of solutions, discussed previously, was obtained with $t_{\mathrm{dom}} = 4$. The lower set of solutions was obtained using the same NP-GA operating settings, except with $t_{\mathrm{dom}} = 2$. With $t_{\mathrm{dom}} = 2$, the NP-GA returned a set of solutions that were clearly suboptimal. One explanation of this result is the following. When a tournament selection scheme is used, there is a nonzero probability that a solution in a given population will not be selected to compete in any of the tournaments. This can result in a potentially good solution being lost by the NP-GA. The probability of losing a solution in this way is equal to $(\frac{N-1}{N})^{t_{\mathrm{dom}}N}$, where N is the population size. When N is large, this probability converges to $e^{-t_{\mathrm{dom}}}$. For $t_{\mathrm{dom}} = 2$, this corresponds to a probability of 0.135 of losing a solution in any given population. When $t_{\mathrm{dom}} = 4$, this probability is reduced to 0.018. By increasing the size of the tournament, we reduce the probability of losing a potentially good solution which could contribute to inadequate convergence of the NP-GA.

There are problems, however, with using too large a tournament size. When we used large values of t_{dom} (for example, $t_{\mathrm{dom}} > 20$), the NP-GA converged to a solution similar to that achieved for $t_{\mathrm{dom}} = 4$, but subsequently fluctuated about that solution as a function of generation number. This instability is a result of having domination tournaments in which multiple nondominated solutions are forced to compete. When nondominated solutions are forced to compete in multiple tournaments, one or more of the members of the nondominated set will inevitably be lost. (The niche count determines the winner of a tied tournament.) The observed instability of the nondominated set is a result of losing and regaining nondominated solutions. When large values of t_{dom} are used, the value of the niche size $(\sigma_{\mathrm{share}})$ becomes increasingly important because multiple tied tournaments may arise. For $t_{\mathrm{dom}} = 4$, we found that the NP-GA performance was relatively insensitive to the value of σ_{share}.

V. Discussion

Genetic algorithm parameters are difficult to determine and few methods exist to systematically set the GA parameters. The total number of generations, the number of solutions in each generation, the crossover rate, and the mutation rate were determined experimentally. Various GA parameter combinations were tested and the results were compared. We found a set of parameters for which the results were consistent in the sense that multiple optimizations gave solutions with similar performances. If the sets returned by different NP-GA runs were not optimal, one would expect that multiple NP-GA runs would return sets with either better or poorer performances. We also attempted to use various σ_{share} values and found that the NP-GA results were robust with respect to σ_{share}.

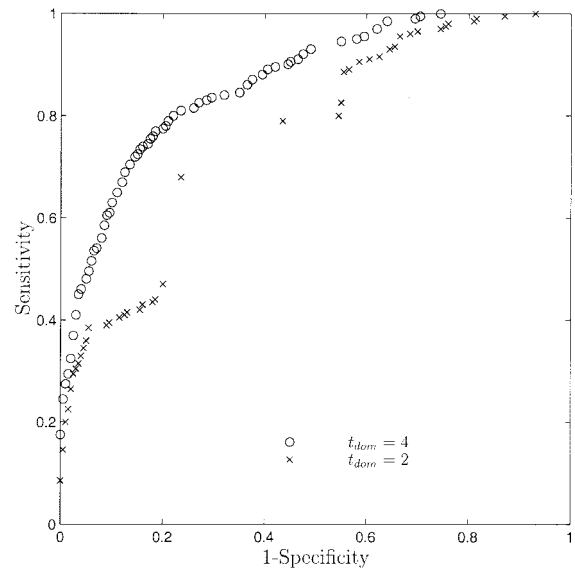

Fig. 10. Effect of t_{dom} on convergence of the NP-GA. At a t_{dom} value of two, the NP-GA converged prematurely because of the lack of domination pressure. For the problems studied in this paper, a t_{dom} value of four resulted in reliable convergence of the NP-GA. Large values of t_{dom} caused the nondominated set to fluctuate randomly.

The NP-GA exhibits several advantages over conventional classifier training techniques. One advantage is that the objective function describing the optimization task is a vector-valued function. This eliminates completely the need to aggregate the different objectives (sensitivity, specificity) into a single scalar function. Rather, *a priori* information about the relative preferences of the objectives can be used post-optimization to choose a member of the Pareto-optimal set as the ultimate solution to the problem.

Another advantage is that a set of nondominated solutions is returned, rather than a single solution. This allows one to select the solution (ROC operating point) whose performance is most clinically appropriate for the diagnostic task at hand. Conventional classifier optimizations can return a series of solutions in the form of a ROC curve obtained by varying certain components of $\vec{w}$ after the classifier has been trained. If the scalar cost function employed is an aggregation of sensitivity and specificity, then only one point in ROC space is guaranteed to be optimal. If the scalar cost function is an aggregation of two different performance measures (such as the sum-of-squares error function for ANN's), then no point is guaranteed to be optimal in ROC space. The NP-GA circumvents this problem by allowing all parameters in $\vec{w}$ to effectively vary in an optimal manner when sweeping out the ROC curve. In this sense, the consistency ROC curve returned by the NP-GA, assuming that the optimization is complete, is optimal at every point. All other possible performances for the same classifier and dataset are either equal to or less than the ROC curve returned by the NP-GA optimization. Training the classifier to operate at a particular operating point and then varying a subset of the parameters in a predetermined way to generate the ROC curve does not ensure this.

As we alluded to earlier, conventional methods of classifier optimization can, in fact, produce the Pareto-optimal operat-

ing points through multiple runs of the scalar optimization procedure with different weighting factors on sensitivity and specificity (see the Appendix for a more detailed discussion of this). Sensitivity and specificity are, however, discrete counting statistics and, hence, are not differentiable functions of $\vec{w}$. Conventional gradient-based optimization methods, such as backpropagation, cannot be employed in this situation. One is therefore left with running multiple scalar optimizations to produce the same operating points that were produced with one single run of the NP-GA. It is also not always clear how to set the relative weightings to evenly sample the Pareto-optimal set using a scalar optimization technique. Another option would be to run multiple optimizations, using a conventional cost function such as the sum-of-squares cost function with different weightings on the two objectives. No point, however, is guaranteed to be a member of the Pareto-optimal set if this type of error function is employed. By using an NP-GA to train pattern classifiers, we are directly addressing the multiobjective nature of classification problem.

If the density functions of the normal and abnormal classes ($f_n(\vec{x})$ and $f_a(\vec{x})$, respectively) are known, then the ROC curve that is produced using the likelihood ratio $\mathrm{LR}(\vec{x}) = f_a(\vec{x})/f_n(\vec{x})$ or any monotonic transformation of the likelihood ratio as the decision variable will be the optimal ROC curve [20], [31]. It will exhibit the best classification performances that can be achieved with the given density functions. It is often very difficult with limited datasets to estimate the density functions of the two classes of data. Thus many classifiers, including those used in this paper, make no attempt to accurately estimate these distributions. The optimal ROC curves that have been discussed in this work are quite different. Within the limitations of the classifier employed and the dataset used for training, the ROC curves produced using the NP-GA are optimal, i.e., there is no better ROC curve that can be produced with the same training data and classifier.

There are sacrifices that are made when the NP-GA is used for classifier optimization. GA's are population-based stochastic optimization algorithms. Thus, they are typically more time consuming than are deterministic algorithms. The time to optimize the linear classifier on a 400-MHz Pentium II system was on the order of 3 min. The time to optimize the ANN on this system was about 20 min. In fact, for very complex systems, an NP-GA optimization may be impractical with current computer technology. For ANN's with a large number of inputs and hidden nodes, the NP-GA may not be suitable for training with current computer technology, because of the large number of parameters. In these situations, the techniques for sweeping out ANN ROC curves proposed by Woods and Bowyer [23] may be better suited. The NP-GA, however, can readily be made to run in parallel, which would substantially decrease the execution time.

This paper has dealt with binary classifiers. It is often important, however, to classify observations into more than two classes (benign, malignant, and normal, for example). For a three-class system, aggregating the multiple objective functions into a single scalar function suffers from the same problems as the two-class problem, but to a greater degree. Here, it is even more difficult to adequately incorporate the class preferences in the aggregated objective function. The ability of the NP-GA to circumvent this difficulty is very attractive. Because the nondominated set of solutions will be larger, care must be taken in determining the NP-GA parameter settings to ensure that the Pareto-optimal set is adequately sampled.

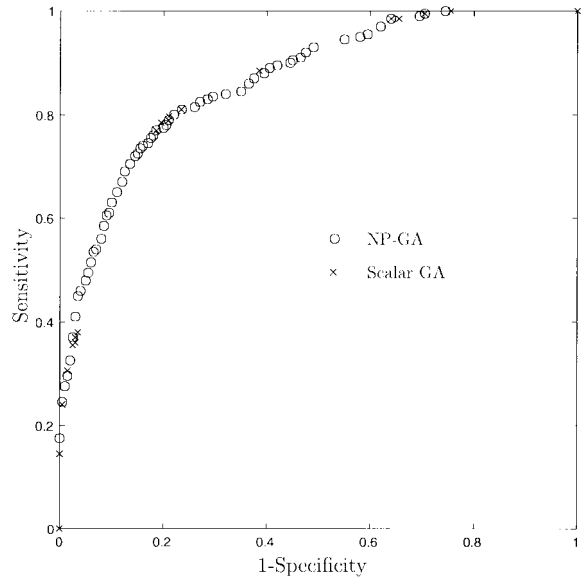

Fig. 11. A comparison of the solutions returned by the NP-GA and the solutions returned by 20 scalar optimizations employing a weighted sum of sensitivity and specificity as the scalar cost function. The two methods returned many similar solutions, but the solutions returned from multiple scalar optimizations tended to clump together in certain areas, whereas the NP-GA solutions were uniformly distributed in ROC space. Note that only 18 of the 20 scalar solutions were distinct.

Complexity and overtraining are issues of great importance in diagnostic classifier research and, in particular, in ANN training [28], [32]. In practice, there is typically a limited amount of training data available, and some sort of regularization is imposed during the classifier training to ensure that it performs well on other (unknown) data sets. It is well known that large ANN weights correspond to complex separation functions [28], [32], which may be indicative of overtraining. To avoid this, we have imposed limitations on the magnitudes of the ANN weights when using the NP-GA to determine the weight values. More systematic methods of regularizing the NP-GA-based training may be possible, however. One such method is to add a third component to the vector objective function that measures complexity. In this way, one can maximize the sensitivity and specificity while minimizing the complexity of the classifier. Depending on the amount and quality of the available training data, a nondominated solution returned by the NP-GA can be chosen such that the classifier performance and generalizability of the result are appropriate for the classification task. We are currently investigating this approach to classifier training.

VI. Conclusion

We have studied the use of a niched Pareto GA in training two popular diagnostic classifiers. Unlike conventional classifier training techniques that formulate the problem as

the solution to a scalar optimization, the NP-GA explicitly addresses the multiobjective nature of the training task. It has been demonstrated that the multiobjective approach removes the ambiguity associated with defining a scalar measure of classifier performance and that it returns a set of optimal solutions that are equivalent in the absence of any information regarding the preference of the objectives (sensitivity, specificity). The performances of these solutions can be interpreted as operating points on an optimal ROC curve, describing the limiting tradeoffs between sensitivity and specificity that are achievable by that classifier, given the available training data. The task of classifier optimization and ROC curve generation are combined into a single task. It was demonstrated that constructing the ROC curve in this way may result in a better ROC curve than is produced by conventional methods of ROC-curve generation. The NP-GA optimization typically requires more computation time than do conventional nonstochastic optimization methods, which may limit its application to certain problems. The advantages of the NP-GA approach to classifier training become more pronounced when the number of classes to be classified increases beyond two.

APPENDIX

In this work, we have investigated the use of a multiobjective optimization algorithm to train diagnostic classifiers and generate ROC curves. In fact, scalar optimization methods can theoretically arrive at the same ROC curves as a multiobjective optimization. Consider the following scalar optimization problem:

$$\text{Maximize} \sum_{i=1}^{p} \lambda_i f_i(\vec{w}) \tag{A1}$$

where $\vec{w}$ is an element of the space of possible parameter vectors $\mathcal{W}$, $\lambda_i > 0$ are fixed, and $\sum_{i=1}^{p} \lambda_i = 1$. Geoffrion [33] proved the following lemma.

Lemma 1

(a) If $\vec{w}_0$ maximizes (A1), then $\vec{w}_0$ is also Pareto-optimal in the vector objective space $[f_1(\vec{w}), f_2(\vec{w}), \ldots, f_p(\vec{w})]$.

(b) Let $\mathcal{W}$ be a convex set and let the f_i be convex on $\mathcal{W}$. Then $\vec{w}_0$ is Pareto-optimal if and only if $\vec{w}_0$ maximizes (A1) for some $\lambda_i > 0$ and $\sum_{i=1}^{p} \lambda_i = 1$.

Because the multiobjective training problem, as we have formulated it, satisfies the convexity conditions used in the Lemma, it must be true that the optimal ROC operating points can be obtained by performing multiple scalar optimizations with varying λ_i's.

It is clear from Fig. 4 that the solutions returned by the NP-GA are Pareto-optimal because, for this problem, we can plot the performances of all possible solutions (the shaded region in Fig. 4). However, in Fig. 7, we cannot plot the performances of all possible solutions due to the large dimensionality of the parameter space. We can, however, make a comparison between the solutions returned by the NP-GA and the solutions returned by multiple scalar optimizations which maximize

$$\lambda \mathit{Sens}(\vec{w}) + (1-\lambda)\mathit{Spec}(\vec{w}) \tag{A2}$$

with λ varying between zero and one. We implemented a scalar GA, using the same GA parameters and parameter restrictions as imposed on the NP-GA to optimize (A2). As described above, the solutions to both of these problems should be Pareto-optimal in ROC space assuming the optimizations are complete. Fig. 11 compares the NP-GA solutions and the solution achieved through multiple runs of a scalar optimization with varying λ. The points returned by the multiple scalar optimizations are similar to certain points returned by the NP-GA. Note that the multiple scalar optimized solutions are clumped together in certain areas of the ROC space. It is unknown, *a-priori*, how to vary λ to evenly sample the Pareto-front, whereas the NP-GA employs niching to ensure an even sampling of the Pareto-front or optimal ROC curve. One also cannot employ gradient-based techniques to optimize discrete performance measures such as sensitivity and specificity. Because of this, we performed 20 separate stochastic scalar optimizations to get the 20 ROC operating points. On the other hand, a more complete sampling of the ROC curve was obtained by a single run of the NP-GA, which required approximately the same CPU time as one run of the scalar optimizer. Thus, despite the theoretical equivalence of the two methods, there are practical advantages to performing a single multiobjective optimization over multiple scalar optimizations.

ACKNOWLEDGMENT

The authors thank Dr. Charles E. Metz and Darrin Edwards for their many helpful suggestions. The authors also thank Dr. Xiaochuan Pan and Dr. Maryellen L. Giger for their frequent encouragement.

REFERENCES

[1] M. L. Giger, "Computer-aided diagnosis," *RSNA Categorical Course Phys.*, pp. 283–298, 1993.

[2] K. Doi, M. L. Giger, R. M. Nishikawa, K. R. Hoffmann, H. MacMahon, R. A. Schmidt, and K.-G. Chua, "Digital radiography: A useful clinical tool for computer-aided diagnosis by quantitative analysis of radiographic images," *Acta Radiologica*, vol. 34, pp. 426–439, 1993.

[3] R. M. Nishikawa, M. L. Giger, K. Doi, C. J. Vyborny, and R. A. Schmidt, "Computer-aided detection of clustered microcalcifications on digital mammograms," *Med. Biol. Eng. Comput.*, vol. 33, pp. 174–178, 1995.

[4] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*. New York: Springer-Verlag, 1996.

[5] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford, U.K.: Oxford Univ. Press, 1995.

[6] C. E. Metz, "Basic principles of ROC analysis," *Seminars Nucl. Med.*, vol. VIII, no. 4, pp. 283–298, 1978.

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

[8] M. A. Anastasio, H. Yoshida, R. Nagel, R. M. Nishikawa, and K. Doi, "A genetic algorithm-based method for optimizing the performance of a computer-aided diagnosis scheme for detection of clustered microcalcifications in mammograms," *Med. Phys.*, vol. 25, no. 9, p. 1613, 1998.

[9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[10] D. B. Fogel, E. C. Wasson III, and E. M. Boughton, "Evolving neural networks for detecting breast cancer," *Cancer Lett.*, vol. 96, pp. 49–53, 1995.

[11] J. D. Shaffer, D. Whitley, and L. J. Eshelman, "Combinations of genetic algorithms and neural networks: A survey of the state of the art," in *Proc. COGANN-92: Int. Workshop Combinations Genetic Algorithms Neural Networks*, Los Alamitos, CA, 1992.

[12] B. Sahiner, H.-P. Chan, D. Wei, N. Petrick, M. A. Helvie, D. D. Adler, and M. M. Gootsitt, "Image feature selection by a genetic algorithm:

Application to classification of mass an normal breast tissue," *Med. Phys.*, vol. 23, no. 10, p. 1671, 1996.

[13] Y. Yuan and H. Zhuang, "A genetic algorithm for generating fuzzy classification rules," *Fuzzy Sets Syst.*, vol. 84, no. 1, 1996.

[14] R. Srikanth, R. George, N. Warsi, D. Parbhu, F. E. Petry, and B. P. Buckles, "A variable-length genetic algorithm for clustering and classification," *Pattern Recognit. Lett.*, vol. 16, no. 8, p. 789, 1995.

[15] D. White and P. Ligomenides, "GANNet: A genetic algorithm for optimizing topology and weights in neural network design," *Lecture Notes Comput. Sci.*, no. 686, pp. 322–327, 1993.

[16] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization," in *Genetic Algorithms: Proc. Fifth Int. Conf.*, San Mateo, CA, July 1993.

[17] J. D. Schaffer and J. J. Grefenstette, "Multi-objective learning via genetic algorithms," in *Proc. Ninth Int. Joint Conf. Artificial Intelligence*, 1985, pp. 593–595.

[18] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computation*, vol. 3, no. 1, pp. 1–16, 1995.

[19] J. Horn and N. Nafpliotis, "Multiobjective optimization using the niched pareto genetic algorithm," in *Proc. First IEEE Conf. Evolutionary Computation, IEEE World Congress Computational Intelligence*, Piscataway, NJ, 1994, vol. 1, pp. 82–87.

[20] J. Egan, *Signal Detection Theory and ROC Analysis*. New York: Academic, 1975.

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

[22] Y. Wu, M. L. Giger, K. Doi, C. J. Vyborny, R. A. Schmidt, and C. E. Metz, "Artificial neural networks in mammography: Application to decision making in the diagnosis of breast cancer," *Radiology*, vol. 187, no. 1, pp. 81–87, 1993.

[23] K. Woods and K. W. Bowyer, "Generating ROC curves for artificial neural networks," *IEEE Trans. Med. Imag.*, vol. 16, pp. 329–337, June 1997.

[24] C. E. Metz, "Evaluation of digital mammography by ROC analysis," in *Digital Mammography* (International Congress Series), K. Doi, Ed. Amsterdam, The Netherlands: Elsevier, 1996, pp. 61–68.

[25] M. A. Anastasio, M. A. Kupinski, and R. M. Nishikawa, "Optimization and FROC analysis of rule-based detection schemes using a multiobjective approach," *IEEE Trans. Med. Imag.*, vol. 17, pp. 1089–1093, Dec. 1998.

[26] J. Horn and N. Nafpliotis, "Multiobjective optimization using the niched parto genetic algorithm," Univ. Illinois, Urbana-Champaign, IlliGAL Rep. 93005, July 1993.

[27] Y.-H. Pao, *Adaptive Pattern Recognition and Neural Networks* Reading, MA: Addison-Wesley, 1989.

[28] D. J. S. MacKay, "Bayesian methods for adaptive models," Ph.D. dissertation, California Inst. Technol., Pasadena, 1992.

[29] J. Hertz, A. Krogh, and R. Palmer, *Introduction to the Theory of Neural Computation* Redwood City, CA: Addison-Wesley, 1991.

[30] S. Haykin, *Neural Networks, A Comprehensive Foundation* New York: Macmillan, 1994.

[31] H. L. Van Trees, *Detection, Estimation, and Modulation Theory Part I* New York: Academic, 1968.

[32] M. A. Kupinski and M. L. Giger, "Investigation of regularized neural networks for the computerized detection of mass lesions in digital mammograms," in *Proc 19th Int. Conf. Engineering Medicine Biology*, Chicago, IL, Oct. 30–Nov. 2, 1997, pp. 1336–1339.

[33] A. M. Geoffrion, "Proper efficiency and the theory of vector maximization," *J. Math. Anal. Appl.*, vol. 22, pp. 618–630, 1968.