Pareto-Based Multiobjective Machine Learning: An Overview and Case Studies

Yaochu Jin, *Senior Member, IEEE*, and Bernhard Sendhoff, *Senior Member, IEEE*

***Abstract*—Machine learning is inherently a multiobjective task. Traditionally, however, either only one of the objectives is adopted as the cost function or multiple objectives are aggregated to a scalar cost function. This can be mainly attributed to the fact that most conventional learning algorithms can only deal with a scalar cost function. Over the last decade, efforts on solving machine learning problems using the Pareto-based multiobjective optimization methodology have gained increasing impetus, particularly due to the great success of multiobjective optimization using evolutionary algorithms and other population-based stochastic search methods. It has been shown that Pareto-based multiobjective learning approaches are more powerful compared to learning algorithms with a scalar cost function in addressing various topics of machine learning, such as clustering, feature selection, improvement of generalization ability, knowledge extraction, and ensemble generation. One common benefit of the different multiobjective learning approaches is that a deeper insight into the learning problem can be gained by analyzing the Pareto front composed of multiple Pareto-optimal solutions. This paper provides an overview of the existing research on multiobjective machine learning, focusing on supervised learning. In addition, a number of case studies are provided to illustrate the major benefits of the Pareto-based approach to machine learning, e.g., how to identify interpretable models and models that can generalize on unseen data from the obtained Pareto-optimal solutions. Three approaches to Pareto-based multiobjective ensemble generation are compared and discussed in detail. Finally, potentially interesting topics in multiobjective machine learning are suggested.**

***Index Terms*—Ensemble, evolutionary multiobjective optimization, generalization, machine learning, multiobjective learning, multiobjective optimization, neural networks, Pareto optimization.**

I. Introduction

MACHINE learning is concerned with the development of computer algorithms and techniques that are able to learn, i.e., to improve automatically through experience [1], [2]. Any machine learning method consists of two steps, i.e., selecting a candidate model, and then, estimating the parameters of the model using a learning algorithm and available data. Very often, model selection and parameter estimation are combined in an iterative process, and in many cases, model selection has been done only once intuitively and empirically. In other words, the user chooses a model empirically, and then, employs a learning algorithm to estimate the parameters of the model.

Machine learning algorithms can largely be divided into three categories. One large category is supervised learning, where the model should approximate the mapping between the input and output of the given data, typically known as regression or classification. Unsupervised learning belongs to the second category of learning algorithms. Data clustering is a typical unsupervised learning method, where a given set of data is to be assigned to different subsets (clusters) so that the data in each subset share some common trait (similarity) defined by a distance measure. The third category is reinforcement learning, which aims to find a policy for an agent to take actions that maximize the cumulated rewards in a given environment.

All learning algorithms perform model selection and parameter estimation based on one or multiple criteria. In supervised learning, the common criterion is an error function that reflects the approximation quality, whereas in clustering, the similarity between the elements in the same cluster (intercluster similarity) should be maximized and the similarity of the elements in different clusters (intracluster similarity) should be minimized. In reinforcement learning, the criterion is a value function that predicts the reward to perform a given action in a given state. Therefore, all learning problems can be considered as an optimization problem. Hereafter, we restrict our discussions mainly to supervised learning and data clustering, since little work has been reported on multicriterion reinforcement learning with few exceptions [3]. In addition, we term any learning criterion an *objective* because we are going to discuss learning problems from the optimization point of view.

A categorization of the existing supervised learning algorithms from the optimization point of view is provided in Section II according to how many objectives are considered in the learning algorithms and whether a scalarized or Pareto-based multiobjective optimization approach is adopted. A brief overview of representative research on Pareto-based multiobjective supervised and unsupervised learning is given in Sections III and Section IV, respectively. To illustrate the benefits of the Pareto-based approach to machine learning, a few illustrative examples are presented in the next sections. The experimental setup of the case studies, including the neural network model, the multiobjective evolutionary algorithm (MOEA), and three benchmark problems are outlined in Section V. Case studies on how to identify interpretable models from the achieved Pareto front, how to select models that are most likely to generalize on unseen data, and how to generate ensembles using the Pareto-based approach are described in Section VI. A summary and outlook of the paper is provided in Section VII.

II. Single- and Multiobjective Learning

We divide learning algorithms into three categories, namely, single-objective learning, scalarized multiobjective learning, and Pareto-based multiobjective learning.

Manuscript received February 27, 2006; revised January 6, 2007. This paper was recommended by Editor L. Mark.

The authors are with Honda Research Institute Europe, Offenbach 63073, Germany (e-mail: yaochu.jin@honda-ri.de; bernhard.sendhoff@honda-ride).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCC.2008.919172

A. Single-Objective Learning

By single-objective learning, we mean learning algorithms in which only one objective function is optimized. Take supervised learning as an example, a single-objective learning algorithm often minimizes the mean squared error (MSE) on the training data

$$f = \frac{1}{N}\sum_{i=1}^{N}(y(i) - y^d(i))^2 \tag{1}$$

where $y(i)$ and $y^d(i)$ are the model output and the desired output, respectively, and N is the number of data pairs in the training data. Several other error measures can also be used as the objective function.

The most often used data clustering algorithm is the k-means clustering algorithm, where the following objective function is minimized:

$$f = \sum_{j=1}^{K}\sum_{x\in C_j} ||x - c_j||^2 \tag{2}$$

where $||\cdot||$ is a chosen distance measure between a data point x and the center (c_j) of cluster C_j, K is the number of clusters.

B. Scalarized Multiobjective Learning

Learning is inherently multiobjective. In supervised learning, memorizing the training data is not the only target. Several other objectives have often to be taken into consideration. In regression and classification, a learning model should not only have good approximation performance on the training data, but also on unseen data from the same problem. But this target cannot be achieved by minimizing the single objective in (1) or any other similar error measures. In fact, only minimizing the approximation error on the training data can result in overfitting the training data, which means that the model is likely to perform poorly on unseen data. In other words, the model is not able to generalize to unseen data. To prevent the model from overfitting the training data, the complexity of the model must be controlled. Another common objective that often needs to be taken into account is the comprehensibility or interpretability of the learned model, which is particularly important when supervised learning is used for knowledge discovery from data. As suggested in [4], interpretability of machine learning models depends strongly on the complexity of the model, and in general, the lower the complexity, the easier it is to understand the model. In both cases, a second objective reflecting the complexity of the model must be considered too. To control the complexity, the two objectives can be aggregated into a scalar objective function

$$f = E + \lambda\Omega \tag{3}$$

where E is a common error function such as the one defined in (1), Ω is a measure for the model complexity, such as the number of free parameters in the model, and $\lambda > 0$ is a positive hyperparameter to be defined by the user. In this way, the learning algorithm is able to optimize two objectives, though the objective function is still a scalar function.

The scalarized multiobjective learning approach has been widely adopted in machine learning, such as regularizing neural networks [5], creating interpretable fuzzy rules [6], [7], and generating negatively correlated ensemble members [8]. Unlike neural networks and fuzzy systems for regression and classification, where complexity control is not a must, some learning models, like support vector machines [9], sparse coding [10], or learning tasks, such as receiver operating characteristics (ROC) analysis [11], explicitly consider more than one objective, which naturally fall into the category of scalarized multiobjective learning.

Similar to supervised learning, multiple objectives can be considered in data clustering as well. On the one hand, it is well recognized that the objective function defined in (2) is strongly biased toward spherically shaped clusters. For data with different types of cluster structures, other objective functions may be more appropriate [12]. On the other hand, it is also suggested that stability, which reflects the variation in the clustering solutions under perturbations should be considered in developing clustering algorithms [13].

There are two main weaknesses if a scalarized objective function is used for multiobjective optimization. First, the determination of an appropriate hyperparameter λ that properly reflects the purpose of the user is not trivial. Second, only a single solution can be obtained, from which little insight into the problem can be gained. This is particularly important if the multiple objectives conflict with each other, and consequently, no single optimal solution exists that optimizes all the objectives simultaneously. This is particularly true for multiobjective learning, e.g., reducing the approximation error often leads to an increase of the complexity of the model. In addition to the aforementioned two drawbacks, it has been pointed out from the optimization point of view that a desired solution may not be achieved using a scalar objective function even if the hyperparameter is specified properly [14]. Note, however, that this weakness can be addressed in part if the hyperparameter is changed dynamically during optimization [15].

An additional, potential advantage of the Pareto-based learning approach is that multiobjectivization may help the learning algorithm from getting out of local optima, thus improving the accuracy of the learning model. Some empirical evidence has been reported in [16] and [17]. However, a rigorous proof of the favorable change to the learning curve by multiobjectivization remains to be shown.

C. Pareto-Based Multiobjective Learning

Using the Pareto approach to address multiple objectives in machine learning is actually a natural idea. However, this approach has not been adopted until a decade ago and has become popular only very recently. The reason is, in our opinion, that traditional learning algorithms, and most traditional optimization algorithms are inefficient in solving multiobjective problems using the Pareto-based approach. In a Pareto-based approach to multiobjective optimization, the objective function is no longer a scalar value, but a vector. As a consequence, a number of Pareto-optimal solutions should be achieved instead of one single solution.

Pareto-optimality is the most important concept in Pareto-based multiobjective optimization. Consider the following m-objective minimization problem:

$$\min F(X),\quad F = \{f_1(X), f_2(X), \dots, f_m(X)\}.$$

A solution X is said to dominate a solution Y if $\forall j = 1, 2, \dots, m, f_j(X) \le f_j(Y)$, and there exists $k \in \{1, 2, \dots, m\}$ such that $f_k(X) < f_k(Y)$. Solution X is called Pareto-optimal if it is not dominated by any other feasible solutions. As previously mentioned, there often exists more than one Pareto-optimal solution if the objectives are conflicting with each other. The curve or surface composed of the Pareto-optimal solutions is known as the Pareto front. In practice, we often do not know where the global Pareto front of a real-world optimization problem lies, and therefore, nondominated solutions achieved by an MOEA are not necessarily Pareto-optimal. However, nondominated solutions achieved by multiobjective optimization algorithms are loosely called Pareto-optimal solutions.

Pareto-based multiobjective learning follows the Pareto-based multiobjective optimization approach to handle learning problems. For example, the scalarized biobjective learning problem in (3) can be formulated as a Pareto-based multiobjective optimization as follows:

$$\min\{f_1, f_2\} \tag{4}$$

$$f_1 = E \tag{5}$$

$$f_2 = \Omega. \tag{6}$$

The most popular error measure is the MSE defined in (1). The complexity of a neural network model can, among others, either be the sum of the squared weights

$$\Omega = \sum_{i=1}^{M} w_i^2 \tag{7}$$

or the sum of the absolute weights

$$\Omega = \sum_{i=1}^{M} |w_i| \tag{8}$$

where $w_i, i = 1, \dots, M$ is a weight in the neural model, and M is the number of weights in total. The aforementioned two complexity measures are often used for neural network regularization and (7) is known as the Gaussian regularizer and (8) the Laplacian regularizer.

Comparing the scalarized multiobjective learning described by (3) and the Pareto-based multiobjective learning described by (4), we find that we no longer need to specify the hyperparameter in the Pareto-based multiobjective learning. On one hand, this spares the user the burden to determine the hyperparameter before learning, on the other hand, the user needs to pick out one or a number of solutions from the achieved Pareto-optimal solutions according to the user's preference after learning. One question may arise: Where is then the difference between the scalarized multiobjective learning and the Pareto-based multiobjective learning? As we will show in the next sections, Pareto-based multiobjective learning algorithms are able to achieve a number of Pareto-optimal solutions, from which the user is able to extract knowledge about the problem and make a better decision when choosing the final solution.

In the following sections, selected existing research on Pareto-based supervised and unsupervised learning algorithms will be briefly reviewed. For an updated and more detailed account of the existing research on multiobjective learning, the reader is referred to [18].

III. Multiobjective Supervised Learning

A. Earlier Ideas

The first ideas to formulate supervised learning as a Pareto-based multiobjective optimization were reported in the mid of 1990s. One of the earliest work in which the neural learning problem was formulated as a multiobjective optimization problem was reported in [19], where two error measures (L_2-norm and L_∞-norm) and one complexity measure (the number of nonzero elements) of a Volterra polynomial basis function network and a Gaussian radial basis function network were minimized using the min–max approach

$$f_1(W) = ||y(W) - y^d(W)||_2 \tag{9}$$

$$f_2(W) = ||y(W) - y^d(W)||_\infty \tag{10}$$

$$f_3(W) = C \tag{11}$$

$$F(W) = \min_W \{\max\{f_1'(W), f_2'(W), f_3'(W)\}\} \tag{12}$$

where C is the number of nonzero weights, $f_1'(W)$, $f_2'(W), f_3'(W)$ are the normalized values of $f_1(W), f_2(W)$, $f_3(W)$, W is the weight matrix of the neural network. Unfortunately, a single-objective genetic algorithm has been employed to implement the learning process, and as a result, only one solution has been achieved.

The weakness of the scalarized approach to handling competitive objectives in learning and the necessity to consider the tradeoff using the Pareto-based approach has been discussed in [20]. An important step forward was made in [21], where the training of a multilayer perceptron network was formulated as a biobjective optimization problem. The MSE and the number of hidden nodes of the network were taken into account. A branch and bound algorithm was employed to solve the mixed integer multiobjective problem. Due to the limited ability of the branch-and-bound algorithm, the advantage of the Pareto-based approach to machine learning was not fully demonstrated in the paper.

With the increasing popularity of MOEAs [22], the idea of employing MOEAs to learning problems became more and more practical. Existing research on Pareto-based approaches to supervised learning can roughly be divided into three categories according to their motivations.

B. Generalization Improvement

One major concern in supervised learning is to generate learning models that not only have good approximation performance on training data, but can also generalize on unseen data. To achieve this, several objectives in addition to the training error can be taken into account. Inspired from neural network regularization, the training error and the sum of the absolute weights were minimized using an ϵ-constraint-based multiobjective optimization method [17]. The Tikhonov regularization term was used as a second objective for a parameter identification problem in [23] and the biobjective problem was solved by a multiobjective real-coded evolutionary algorithm. Similar to [21], the training error and the number of hidden nodes of a feedforward neural network are minimized using a Pareto-based differential evolution algorithm [24]. The influence of three different regularization terms on complexity minimization has been discussed in [25] using an multiobjective optimization approach. Different to the conclusion drawn from gradient-based regularization algorithms, it is shown that the Gaussian regularizer is also able to efficiently reduce the network complexity like the Laplacian regularizer when an evolutionary method is used [26].

Another idea to improve the generalization performance of neural networks is to minimize different, potentially conflicting error measures [27], such as the Euclidian error, and the robust error, which can be defined by

$$E_r = \exp(\lambda|\vec{y} - \vec{y}^d|^p) \tag{13}$$

where λ and p are two parameters to be defined. In [28], two different methods for determining nondominated solutions were investigated, one using a validation dataset rather than the training set, and the other using a boosting approach.

Cooperative coevolution of neural networks based on multiple objectives has been studied in [29]. Two populations coevolve in the algorithm, the module (subnetwork) population and the network population. The module population consists again of a number of subpopulations, each of which evolves both the structure and weights of a subnetwork (a subcomponent of a neural network). The chromosome of the network population encodes which subcomponents should be picked out to construct the whole neural network. A steady-state genetic algorithm is used for the network population. For coevolutionary algorithms, it is not straightforward to determine the fitness value of the individuals in the module population. In [29], several criteria for evaluating the fitness of the modules are discussed. The first criterion is concerned with the performance of the modules, which can again be determined in different ways. For example, the performance of a module can be the mean fitness value of a number of best neural networks in which the model participates. Alternatively, the performance of a module can be determined by the average fitness change of the best neural networks when the module is replaced or removed. The second criterion is the number of neural networks the module is present in, which is to be maximized during the optimization. The third criterion is the complexity of the module, including the number of connections (NC), the number of nodes, and the sum of the absolute value of the weights. Two objectives are considered for the network population, namely, the performance and the fitness of each module.

In addition to feedforward neural networks, tradeoff between accuracy and complexity using the Pareto-based approach has also been considered for generation of radial-basis neural networks [30], [31], support vector machines [32]–[34], decision trees [35], and classifier systems [36]. Interesting applications of Pareto-based multiobjective learning to face detection [37], feature extraction [38], robotics [39], and text retrieval [40] have been reported.

C. Interpretability Enhancement in Rule Extraction

Extraction of logic or fuzzy rules from data or from trained neural networks is an important approach to knowledge discovery. One critical issue here is the interpretability, also known as understandability or transparency of the generated rules. Several aspects can be highly related to the interpretability of rules [41], such as the compactness (number of rules, number of premises) and the consistency of the rules. For fuzzy rules, the partition of the fuzzy subsets should be well distinguishable so that a meaningful term can be attached to the fuzzy subsets. Different aspects of interpretability have been coped with using the scalarized multiobjective optimization [6], [7].

The first idea to improve understandability of rule systems is to select a small subset from a large number of rules generated from data. A Pareto-based multiobjective genetic algorithm (MOGA) was used to generate fuzzy rules by trading off the classification error against the number of rules [42]. Similar work has also been reported in [43] and [44]. A step further is to include a third objective that minimizes the rule length (number of premises) [45], or the number of selected input variables [46]. To improve the distinguishability of the fuzzy partition, the maximum similarity between the fuzzy subsets has also been minimized in addition to accuracy and compactness [47]. To further improve the distinguishability of the fuzzy partition, similar subsets are merged, singletons are removed, and overlapped subsets are separated in multiobjective optimization of fuzzy rules considering accuracy and compactness with application to both classification and regression problems [48], [49].

Several objectives have to be optimized in extracting logic rules from trained neural networks, such as coverage, i.e., the number of patterns correctly classified by a rule set, error, i.e., the number of the patterns that are misclassified, and compactness [50].

The main advantage of the Pareto-based approach to generating interpretable fuzzy rules is that the user is able to choose a preferred solution from a number of Pareto-optimal solutions.

D. Diverse Ensemble Generation

An ensemble of learning models performs much better than a single learning model, if the members of the ensemble are sufficiently different [51]. However, there is a tradeoff between accuracy and diversity and it is essential that the ensemble members are highly diverse and sufficiently accurate [52], [53]. Previously, the diversity of the ensemble members has been promoted through the use of different data, different learning algorithms or different learning models [54]. An alternative approach is to develop a learning algorithm that reduces the training error and minimizes the correlation among the outputs of the ensemble members. Traditionally, the approximation error and the output correlation between the ensemble members are summed up to a scalar objective function [8], [55]. In [52], the Pareto-based approach is adopted to generate diverse and accurate ensembles, where the following two objectives are minimized,

$$f_1 = \frac{1}{N}\sum_{i=1}^{N}(y(i) - y^d(i))^2 \tag{14}$$

$$f_2 = \sum_{i=1}^{N}(y_k(i) - y(i))\left[\sum_{j\neq k, j=1}^{M}(y_j(i) - y(i))\right] \tag{15}$$

where $y_k(i)$ is the output of the kth ensemble member, $y(i)$ is the output of the ensemble for the ith training sample, N is the number of training samples, and M is the number of members in the ensemble. This research has been extended to a framework for evolving ensembles that is composed of three levels of evolution [56]. On the first level, a mixture of learning models, such as multilayer perceptrons, radial basis function networks, and support vector machines are evolved. On the second level, different training datasets are used for evolving the hybrid ensembles produced on the first level. On the third level, all subsets of homogenous learning models of the hybrid ensembles generated on the second level are evolved separately to minimize training error and correlation between the ensemble members. In each iteration, the current ensemble, which consists of each of the different types of models, is archived if it dominates the previous best ensemble based on training error and test error. The ensemble in the archive serves as the final hybrid ensemble.

A different idea to take advantage of Pareto-based learning for ensemble generation has been presented in [57], where the training data is divided into two sets and the errors on the two datasets are used as two objectives for learning

$$f_1 = \sum_{i=1}^{N_1}\left(y(i) - y_1^d(i)\right)^2 \tag{16}$$

$$f_2 = \sum_{i=1}^{N_2}\left(y(i) - y_2^d(i)\right)^2 \tag{17}$$

where y_j^d are the training data in dataset $j, j = 1, 2$, N_1 and N_2 are the size of the datasets. One should take care that the neural network model used should be sufficiently small in order not to overfit both datasets.

Another idea suggested for generating neural network ensembles is to include the complexity measure as the second objective [25], [26]

$$f_1 = \sum_{i=1}^{N}(y(i) - y^d(i))^2 \tag{18}$$

$$f_2 = C \tag{19}$$

where C is the NC in the neural network. In this way, the diversity of the networks is achieved in terms of different network structures, which is ensured by the fact that ensemble members always have different NC. Simulation results on both regression and classification problems show that the approach is effective in generating neural network ensembles. It should be noticed, however, that very simple Pareto-optimal neural networks will be generated whose error on the training data can be very large. These networks should not be included in the ensemble if models of high accuracy are targeted. One question that has not been answered in [25] and [26] is how to choose ensemble members from the nondominated solutions. We will come back to this issue again in the case studies.

The method for multiobjective cooperative coevolution of the neural networks in [29] has also been applied to generating neural network ensembles [58]. In case of ensemble generation, one population evolves single neural networks and the other evolves neural network ensembles. For the population evolving single networks, objectives with respect to the performance of the single network, the performance on difficult patterns (measured, e.g., by the number of ensembles misclassifying it), and the average performance of the ensembles in which the network is present can be taken into account for evaluating the performance of the single networks. In addition, network complexity, ability to cooperate, and diversity are other objectives to consider. In addition to the correlation measure used in [52], functional diversity, which measures the average Euclidean distances among the outputs of two neural networks, mutual information between the output of two networks, and the Yule's Q statistics [59], which measures the correlation of the errors made by two models, are also considered. For the ensemble population, performance and ambiguity are two objectives to optimize. It has been shown that the generalization performance of the ensembles generated using the multiobjective approach is significantly better than that of the ensembles generated by classical approaches.

Pareto-based generation of ensembles for radial basis function networks [60] and fuzzy rule systems [61] have also been reported.

E. Miscellaneous

Much early work on Pareto-based multiobjective learning has been motivated by specific applications, where multiple objectives have to be considered even without thinking about generalization. For example, in generating the ROC curve for classifiers, both the true positive rate (TPR) and the false positive rate (FPR) are to be minimized. In [62], the Niched Pareto GA [63] was employed to generate the ROC curves of neural network classifiers [62]. It has been shown that better results can be obtained by using the Pareto-based approach compared to the traditional method for generating ROC curve usually by changing the threshold of the neural classifier after training. Notice that traditionally, ROC analysis is just a method for evaluating a given classifier, but in the Pareto-based approach, the classifiers on the ROC curve are different. Most recently, the generalization ability of neural classifiers using the Pareto-based approach to ROC curve generation has been studied in [64], and Pareto-based multiobjective multiclass ROC analysis has been investigated in [65].

Systems control is another area in which multiple objectives need to be satisfied. In [66], Pareto-based evolutionary programming was used to minimize the undershooting and overall tracking error of a neural-network-based controller. A number of Pareto-optimal solutions are obtained and the control performance of some typical Pareto-solutions is analyzed.

Supervised feature selection is one of the machine learning tasks where a tradeoff between the number of selected features and the performance of the learning model using the features must be considered. As a result, the Pareto-based multiobjective learning has been investigated [67]–[69].

IV. Multiobjective Unsupervised Learning

In this section, we discuss existing research work on Pareto-based multiobjective unsupervised learning, mainly multiobjective data clustering. In [70], four objectives are considered in Pareto-based evolutionary data clustering. The first objective is concerned with the cluster cohesiveness, which favors dense clusters, the second objective is to maximize the separateness between the clusters measured by their distance from the global centroid, the third objective is meant to reduce the number of clusters, and the fourth one minimizes the number of selected features. Rather than combining the objectives, a Pareto-based evolutionary algorithm has been employed to achieve multiple Pareto-optimal solutions. Through analyzing the individual Pareto-optimal solutions, significant features and an appropriate number of clusters can be identified.

The advantage of Pareto-based data clustering has been convincingly demonstrated in [71], where the number of clusters can be determined automatically by analyzing the Pareto front. In that paper, two objectives are minimized to reflect the compactness of clusters and the connectedness of data points. The cluster compactness is described by the overall deviation of a partitioning and the connectedness checks the degree to which data points in a neighborhood are assigned to the same cluster

$$f_1 = \sum_{C_k \in C} \sum_{x_i \in C_k} ||x_i - c_k||_2 \tag{20}$$

$$f_2 = \sum_{i=1}^{N} \sum_{j=1}^{L} \gamma_{ij} \tag{21}$$

where $C = \{C_1, C_2, \ldots, C_K\}$ is a union of all clusters, c_k is the center of cluster C_k, $k = 1, 2, \ldots, K$, x_i is a data point assigned to cluster C_k, K is the number of clusters, L is the number of data points in a predefined neighborhood, and γ_{ij} is defined by

$$\gamma_{ij} = \begin{cases} \frac{1}{j}, & \text{if } x_i \text{ and } \mathrm{NN}_j(x_i) \text{ are not in the same cluster} \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

where $\mathrm{NN}_j(x_i)$ is the jth nearest neighbor of data point x_i.

The Pareto-optimal solutions trading off between deviation and connectivity are plotted in such a way that the number of clusters contained in the Pareto-optimal solutions increases from left to right. It is argued that the overall deviation decreases with the increasing number of clusters and when the cluster number is larger than the "true" number of clusters, the gain in deviation minimization will be minor while the cost in connectivity increases rapidly. Thus, the Pareto-optimal solution that delivers the maximal gain in performance against the increase in the number of clusters provides the correct number of clusters, as suggested in [72].

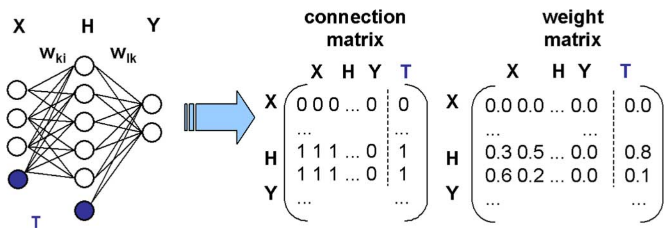


Fig. 1. Coding of the structure and parameters of neural networks using a connection matrix and a weight matrix.

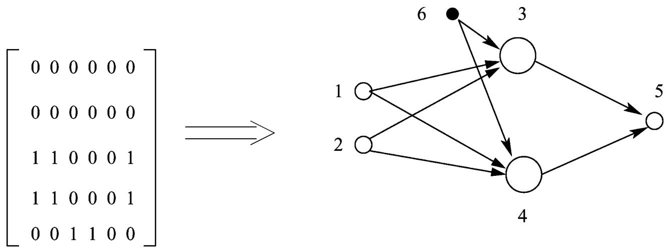


Fig. 2. Example of a connection matrix and its corresponding neural network structure.

V. Case Studies: Experimental Setup

A. Neural Network Model

Feedforward neural networks with one hidden layer are used in the case studies. The hidden neurons are nonlinear and the output neurons are linear. The activation function used for the hidden neurons is as follows:

$$g(z) = \frac{x}{1 + |x|}. \tag{23}$$

In the optimization, the maximum of hidden nodes is set to 10. Weights are initialized between -0.2 and 0.2.

B. Evolutionary Algorithms for Pareto-Based Learning

1) Coding of Neural Networks: A connection matrix and a weight matrix are employed to describe the structure and the weights of the neural networks, see Fig. 1. The connection matrix specifies the structure of the network, whereas the weight matrix determines the strength of each connection. Assuming that a neural network consists of M neurons in total, including the input and output neurons, then the size of the connection matrix is $M \times (M + 1)$, where an element in the last column indicates whether a neuron is connected to a bias value. In the connection matrix, if element $c_{ij}, i = 1, \ldots, M, j = 1, \ldots, M$ equals 1, it means that there is a connection between the ith and jth neuron and the signal flows from neuron j to neuron i. If $j = M + 1$, it indicates that there is a bias in the ith neuron. Fig. 2 illustrates a connection matrix and the corresponding network structure. It can be seen from the figure that the network

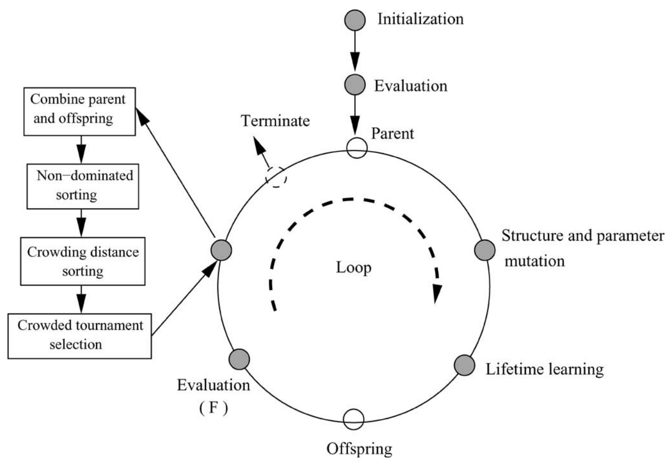

Initialization
Evaluation
Parent
Terminate
Combine parent and offspring
Non−dominated sorting
Crowding distance sorting
Crowded tournament selection
Loop
Structure and parameter mutation
Lifetime learning
Evaluation (F)
Offspring

Fig. 3. Framework for evolutionary multiobjective optimization of neural networks.

has two input neurons, two hidden neurons, and one output neuron. Besides, both hidden neurons have a bias.

2) Mutations of Structure and Weights: Evolutionary algorithms have widely been employed to optimize both the structure and parameters of neural networks, often combined with a gradient-based local search method [73]. The framework for evolutionary multiobjective optimization of neural networks employed in our case studies is shown in Fig. 3. In comparison to conventional evolutionary optimization, we note that only mutation operations are used in the framework for varying the structure and parameters of neural networks, which are specific to neural networks, including inserting a new neuron or deleting an existing neuron, adding or removing a connection between two neurons. A Gaussian mutation is applied to the weights

$$\Delta w_{ij} = N(0, \sigma_w) \tag{24}$$

where w_{ij} denotes the weight connecting neuron j and neuron i, σ_w is the standard deviation of the Gaussian distribution.

3) Lifetime Learning: After mutation, lifetime learning using an improved version of the Rprop algorithm [74] has been employed to fine tune the weights. After lifetime learning, the fitness of each individual regarding the approximation error (f_1) is updated. In addition, the weights modified during the lifetime learning are encoded back to the chromosome, which is known as the Lamarckian type of inheritance.

The Rprop learning algorithm [75] is believed to be a fast and robust learning algorithm. In each iteration, the weights are modified in the following manner

$$\Delta w_{ij}^{(t)} = -\text{sign}\left(\frac{\partial E^{(t)}}{\partial w_{ij}}\right)\Delta_{ij}^{(t)} \tag{25}$$

where sign$(\cdot)$ is the sign function, $\Delta_{ij}^{(t)} \geq 0$ is the step size, which is initialized to Δ_0 for all weights. The step size for each weight is adjusted as

$$\Delta_{ij}^{(t)} = \begin{cases} \xi^{+}\Delta_{ij}^{(t-1)}, & \text{if } \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} \times \dfrac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ \xi^{-}\Delta_{ij}^{(t-1)}, & \text{if } \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} \times \dfrac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ \Delta_{ij}^{(t-1)}, & \text{otherwise} \end{cases} \tag{26}$$

where $0 < \xi^{-} < 1 < \xi^{+}$. To prevent the step sizes from becoming too large or too small, they are bounded by $\Delta_{\min} \leq \Delta_{ij} \leq \Delta_{\max}$.

After the weights are updated, it is necessary to check if the partial derivative changes sign, which indicates that the previous step might be too large, and thus, a minimum has been missed. In this case, the previous weight change should be retracted

$$\Delta w_{ij}^{(t)} = -\Delta_{ij}^{(t-1)}, \qquad \text{if} \frac{\partial E^{(t-1)}}{\partial w_{ij}} \times \frac{\partial E^{(t)}}{\partial w_{ij}} < 0. \tag{27}$$

Recall that if the weight change is retracted in the tth iteration, the $\partial E^{(t)}/\partial w_{ij}$ should be set to 0.

In reference [74], it is argued that the condition for weight retraction in (27) is not always reasonable. The weight change should be retracted only if the partial derivative changes sign and if the approximation error increases. Thus, the weight retraction condition in (27) is modified as follows:

$$\Delta w^{(t)} = -\Delta_{ij}^{(t-1)}, \quad \text{if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} \times \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \quad \text{and} \quad E^{(t)} > E^{(t-1)}. \tag{28}$$

It has been shown on several benchmark problems that the modified Rprop (termed as Rprop^{+}) exhibits consistently better performance than the Rprop algorithm [74].

4) Selection: The most significant difference of multiobjective optimization to scalar optimization is the selection method. In our research, the selection method from NSGA-II [76] is adopted, which consists of four major steps. First, the parent and offspring populations are combined. This implies that NSGA-II is an elitism. Second, the combined population is sorted according to the nondominance ranks. During the ranking, nondominated solutions in the combined population are assigned a rank 1, which belongs to the first nondominated front. These individuals are removed temporally from the population, and the nondominated individuals in the rest of the population are identified, which consists of the second nondominated front of the population and are assigned a rank 2. This procedure repeats until all individuals in the combined population are assigned with a rank from 1 to R, assuming that R nondominated fronts can be identified in total. Third, a crowding distance reflecting the crowdedness in the neighborhood of a particular solution is calculated. The crowding distance of solution i in the nondominated front j, $(j = 1, \ldots, R)$ is the distance between the two neighbors of solution s_i^j in the objective space

$$d_i^j = \sum_{k=1}^{m} |f_k(s_{i-1}^j) - f_k(s_{i+1}^j)| \tag{29}$$

TABLE I
PARAMETER SETTINGS OF THE ALGORITHMS

Neural Network Initialization	
maximum number of hidden neurons	10
initial weights	-0.2 $\sim$ 0.2
Evolutionary Algorithm	
population size	100
mutation rate	0.20
σ_w	0.1
Rprop$^+$ Algorithm	
ξ^+	1.2
ξ^-	0.5
Δ_0	0.01
$\Delta_{\max}$	50
$\Delta_{\min}$	10^{-6}

where m is the number of objectives in the multiobjective optimization problem and solutions s_{i-1}^j and s_{i+1}^j are the two neighboring solutions of solution s_i^j. A large distance is assigned to the boundary solutions in each nondominated front. Here, the larger the crowding distance, the less crowded around the solution s_i^j it is. Fourth, a tournament selection that leverages between nondominated ranking and crowdedness is conducted. Given two randomly chosen individuals, the solution with the better (lower) rank wins the tournament. If the two solutions have the same rank, the one with the larger crowding distance wins. If the two solutions have the same rank and the same crowding distance, choose a winner randomly. This procedure continues until the required number of offspring is generated.

The parameter settings used in the simulations are summarized in Table I.

C. Benchmark Problems

1) Wisconsin Breast Cancer Data: The Wisconsin breast cancer diagnosis problem in the University of California at Irvine (UCI) repository of machine learning database was collected by Dr. W. H. Wolberg at the University of Wisconsin-Madison Hospitalics [77]. The benchmark problem contains 699 examples, each of which has nine inputs and two outputs. The inputs are: clump thickness (x_1), uniformity of cell size (x_2), uniformity of cell shape (x_3), marginal adhesion (x_4), single epithelial cell size (x_5), bare nuclei (x_6), bland chromatin (x_7), normal nucleoli (x_8), and mitosis (x_9). All inputs are normalized, to be more exact, $x_1, \ldots, x_9 \in \{0.1,\ 0.2, \ldots, 0.8,\ 0.9,\ 1.0\}$. The two outputs are a complementary binary value, i.e., if the first output is 1, which means "benign," then the second output is 0. Otherwise, the first output is 0, which means "malignant," and the second output is 1. Therefore, only the first output is used.

2) Diabetes Data: The Pima Indians Diabetes Data consists of 768 data pairs with eight attributes normalized between 0 and 1 [77]. The eight attributes are number of pregnant (x_1), plasma glucose concentration (x_2), blood pressure (x_3), triceps skin fold thickness (x_4), 2 h serum insulin (x_5), body mass index (x_6), diabetes pedigree function (x_7), and age (x_8). In this database, 268 instances are positive (output equals 1) and 500 instances are negative (output equals 0).

3) Iris Data: The third dataset we looked at is the Iris data [77]. The dataset contains three classes of 40 instances each, where each class refers to a type of Iris plant. The three classes are: Iris setosa (class 1, represented by -1), Iris versicolor (class 2, represented by 0), and Iris virginica (class 3, represented by 1). Four attributes are used to predict the Iris class, i.e., sepal length (x_1), sepal width (x_2), petal length (x_3), and petal width (x_4), all in centimeters. Among the three classes, class 1 is linearly separable from the other two classes, and classes 2 and 3 are not linearly separable from each other. To ease knowledge extraction, we reformulate the data with three outputs, where class 1 is represented by $\{1, 0, 0\}$, class 2 by $\{0, 1, 0\}$, and class 3 by $\{0, 0, 1\}$.

VI. CASE STUDIES: RESULTS

Based on the MOEA described in the previous section, we show in this section how one can benefit from Pareto-based multiobjective learning. We generate a number of Pareto-optimal neural network models that trade the accuracy on training data off against the network complexity. We show on the three benchmark problems how to identify interpretable neural networks from which understandable logic rules can be extracted, and networks that are most likely to generalize on unseen data, from the achieved Pareto-optimal solutions. Afterwards, we compare three methods for generating neural network ensembles using the Pareto-based multiobjective learning, which are suggested by Abbass [57], Chandra and Yao [52], and Jin *et al.* [26].

A. Identifying Interpretable Models

As suggested in [4], interpretability of neural networks is mainly determined by their complexity. The simpler a network, the easier it is to understand the knowledge embedded in the neural network. This is also true if we look at the definition of interpretability of fuzzy systems [6], [41].

When we minimize both accuracy and complexity of the networks in a Pareto-based approach, we are able to achieve a number of Pareto-optimal solutions with a complexity ranging from very simple networks to highly complex ones. We argue that the simple Pareto-optimal neural networks on the Pareto front are actually the interpretable models from which understandable logic rules can be extracted. Before providing examples on the benchmark problems, we first briefly describe the rule extraction method we adopted in this case study, which is similar to the one used in [78]. Consider a simple neural network with one single input, one hidden neuron, and one output neuron, refer to Fig. 4. For binary classification problems, we usually assume that an instance is labeled as class 1 if the output is smaller than 0.5. Otherwise, it is labeled as class 2. To have more confidence in decision making, we can also define a stronger criterion, for instance:

$$\begin{array}{lll} \text{If} & y \geq 0.75, & \text{then class 1} \\ \text{If} & y \leq 0.25, & \text{then class 2} \\ \text{If} & 0.25 < y < 0.75, & \text{undecided.} \end{array} \tag{30}$$

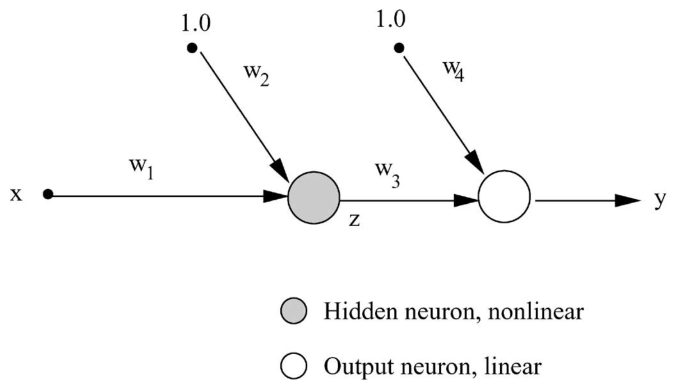


Fig. 4. Typical simple network for extracting logic rules.

In the following, we will show how to derive rules from neural networks using the defined thresholds. Let the output of the hidden neuron be z, then a rule that defines class 1 should satisfy

$$w_3\, z + w_4 \geq 0.75. \tag{31}$$

Then, we get

$$z \geq \frac{(0.75 - w_4)}{w_3}, \quad \text{if } w_3 > 0$$
$$z \leq \frac{(0.75 - w_4)}{w_3}, \quad \text{if } w_3 < 0.$$

Consider the first case and define $(0.75 - w_4)/w_3 = \theta_1 > 0$, we have

$$\frac{w_1\, x + w_2}{1 + |w_1\, x + w_2|} \geq \theta_1. \tag{32}$$

Since $\theta_1 > 0$, $w_1\, x + w_2$ must also be larger than zero to satisfy the conditions for class 1. Consequently,

$$\frac{w_1\, x + w_2}{1 + w_1\, x + w_2} \geq \theta_1 \tag{33}$$

and

$$x \geq \frac{\theta_1 - w_2(1-\theta_1)}{w_1(1-\theta_1)}, \quad \text{if } w_1(1-\theta_1) > 0 \tag{34}$$

$$x \leq \frac{\theta_1 - w_2(1-\theta_1)}{w_1(1-\theta_1)}, \quad \text{if } w_1(1-\theta_1) < 0. \tag{35}$$

Let $[\theta_1 - w_2(1-\theta_1)]/[w_1(1-\theta_1)] = \theta_2$, either of the following two rules can be extracted that defines the condition for class 1:

$$\text{If } x \geq \theta_2, \quad \text{then class 1}, \quad \text{if } w_1(1-\theta_1) > 0$$
$$\text{If } x \leq \theta_2, \quad \text{then class 1}, \quad \text{if } w_1(1-\theta_1) < 0.$$

Note, however, that it can happen that no rule can be extracted from the neural network. For instance, if $\forall z, w_3\, z + w_4 < 0.75$. In this case, the neural network is not able to separate the two classes.

1) Wisconsin Breast Cancer Data: For rule extraction, all available data are used for training the neural network. The Pareto-optimal solutions from a typical run are plotted in Fig. 5.

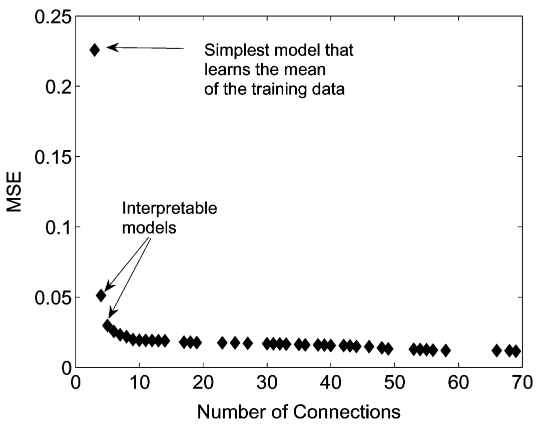


Fig. 5. Typical Pareto-front obtained for the breast cancer data composed of 41 solutions.

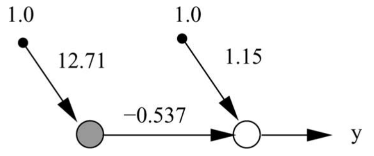


Fig. 6. Simplest Pareto-optimal network model for the breast cancer data, which exactly learns the mean of the training data.

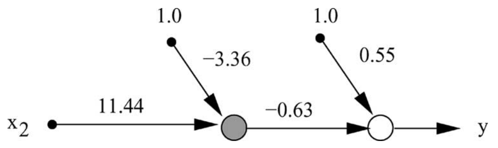


Fig. 7. Pareto-optimal network model with four connections for the breast cancer data.

As we will show later on, the simplest Pareto-optimal neural networks achieved from different runs are almost identical.

Let us now look at the simplest Pareto-optimal neural networks. The simplest neural network has three connections in total, in which no input is selected. In other words, the input of the neural network is constant, refer to Fig. 6. Interestingly, this neural network learns exactly the mean output of the training data.

The second simplest network is presented in Fig. 7, which has four connections. Of the nine input attributes, only x_2 (uniformity of cell sizes) is selected, which implies that x_2 might be the most important feature for determining whether an instance is benign or malignant. The MSE of the network is 0.051. From the network, the following two rules can be extracted using the previously described rule extraction method:

$$\text{If } x_2 \leq 0.2, \quad \text{then benign}$$
$$\text{If } x_2 \geq 0.4, \quad \text{then malignant}.$$

With these two simple rules, the correct classification rate is 97.0% on 602 instances with the rest 97 instances undetermined, recalling that the thresholds are set to 0.75 and 0.25 to make sure that the decision is confident enough. However, if we set the

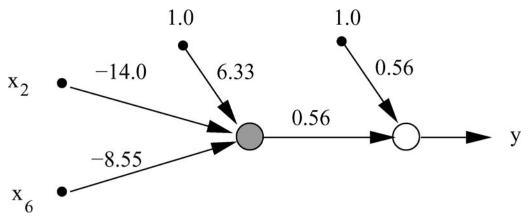


Fig. 8. Pareto-optimal network model with five connections for the breast cancer data.

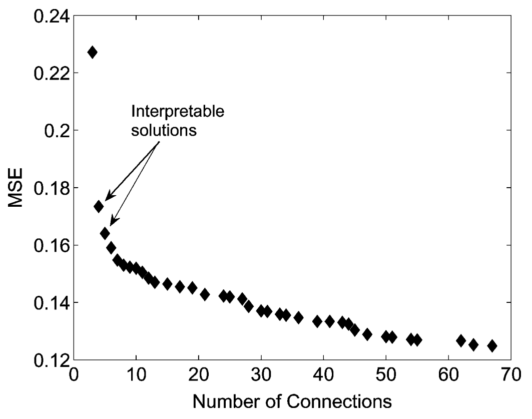


Fig. 9. Typical Pareto-front obtained for the diabetes data composed of 37 solutions.

classification threshold to 0.5, the following rule can be obtained with a correct classification rate of 92.4% on all instances.

$$\text{If } x_2 \le 0.3, \quad \text{then benign}$$
$$\text{otherwise malignant.}$$

The next simple Pareto-optimal neural network has five connections, in which both x_2 and x_6 are chosen as input features (see Fig. 8). The MSE of the model is 0.029. From this neural network, the following two rules can be extracted:

$$\text{If } 14\,x_2 + 8.55\,x_6 \le 5.81, \quad \text{then benign}$$
$$\text{If } 14\,x_2 + 8.55\,x_6 \ge 7.55, \quad \text{then malignant.}$$

Using these two rules, the correct classification rate is 97.2% on 680 instances with the rest 19 instances undetermined. If the threshold is set to 0.5, the following rule can be obtained with a correct classification rate of 96.4% on all instances:

$$\text{If } 14\,x_2 + 8.55\,x_6 \le 6.45, \quad \text{then benign}$$
$$\text{otherwise malignant.}$$

2) Diabetes Data: The same empirical study is conducted on the diabetes data. The achieved Pareto front is shown in Fig 9.

Same as the breast cancer data, the simplest Pareto-optimal solution contains three connections and learns the mean of the output value. The two simple Pareto solutions with at least one attribute chosen are plotted in Figs. 10 and 11, respectively. The MSEs of the two simple network models are 0.17 and 0.16.

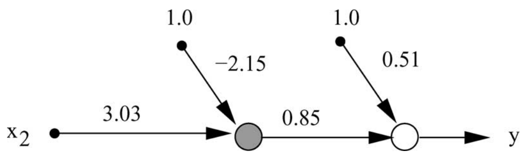


Fig. 10. Pareto-optimal network model with four connections for the diabetes data.

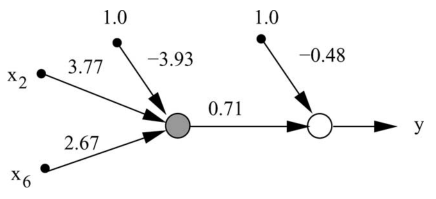


Fig. 11. Pareto-optimal network model with five connections for the diabetes data.

From the neural network with four connections (see Fig. 10), the following two rules can be extracted:

$$\text{If } x_2 \le 0.83, \quad \text{then positive}$$
$$\text{If } x_2 \ge 0.56, \quad \text{then negative.}$$

By applying the aforementioned two rules, we are able to make a decision on 413 instances with a correct classification rate of 85.4%. The rest 355 instances cannot be determined with these two rules.

If we set the threshold to 0.5, the following rule is obtained:

$$\text{If } x_2 \le 0.72, \quad \text{then positive} \quad \text{otherwise negative.} \tag{37}$$

The correct classification rate using the aforementioned rule is 75.0% on all 768 instances.

The following rules can be obtained for the neural network in Fig. 11, when the threshold is set to 0.75 and 0.25:

$$\text{If } 3.77\,x_2 + 2.67\,x_6 \le 4.54, \quad \text{then positive}$$
$$\text{If } 3.77\,x_2 + 2.67\,x_6 \ge 3.46, \quad \text{then positive.}$$

With these two rules, the correct classification rate is 85.4% with the rest 308 instances undecided. If the threshold is set to 0.5, we then have the following rule:

$$\text{If } 3.77\,x_2 + 2.67\,x_6 \le 3.97, \quad \text{then positive} \quad \text{otherwise negative.} \tag{38}$$

From the aforementioned rule, the correct classification rate on all 768 instances is 77.0%.

3) Iris Data: The Pareto front from the Iris data is presented in Fig. 12, which consists of 20 solutions (two Pareto optimal solutions have the same MSE and complexity). Again, the simplest network with seven connections approximates the mean value of the output.

The two Pareto-optimal networks with eight connections are plotted in Figs. 13 and 14, respectively. From the figures, we

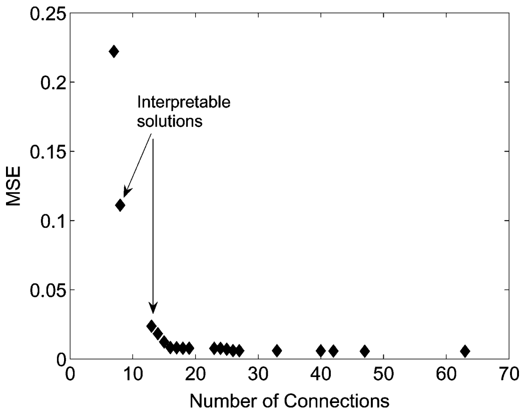


Fig. 12. Typical Pareto-front obtained for the Iris data composed of 20 solutions.

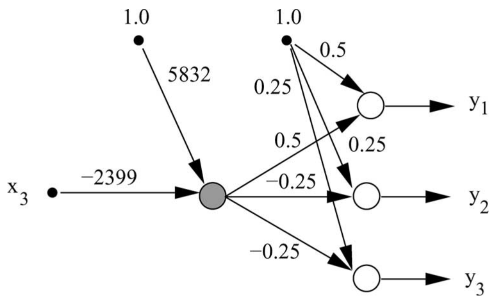


Fig. 13. Pareto-optimal network model with eight connections for the Iris data. In this model, x_3 is chosen as the input.

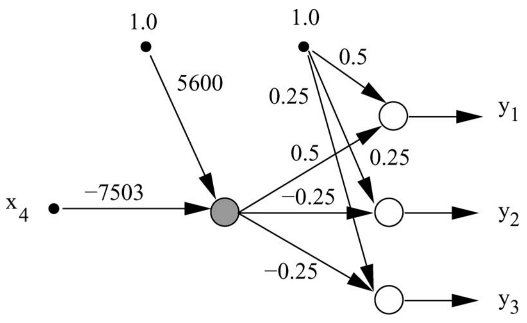


Fig. 14. Pareto-optimal network model with eight connections for the Iris data. In this model, x_4 is chosen as the input.

notice that only one of the attribute (either x_3 or x_4) is chosen. From the network in Fig. 13, the following rule can be extracted:

$$\text{If } x_3 \le 2.4, \qquad \text{then Iris setosa.} \tag{39}$$

Similarly, the following rule can be extracted form the network in Fig. 13:

$$\text{If } x_4 \le 0.80, \qquad \text{then Iris setosa.} \tag{40}$$

It can be easily verified that both rules are able to separate Iris sesota from the other two classes correctly.

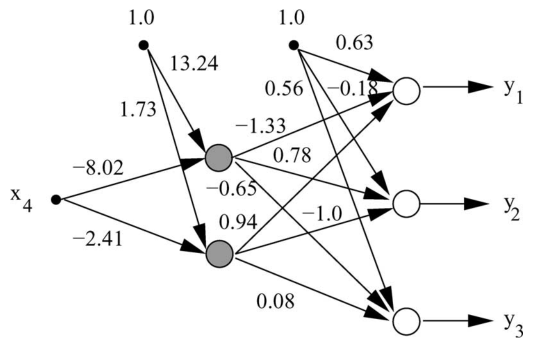


Fig. 15. Pareto-optimal network model with 13 connections for the Iris data. x_4 is chosen as the input.

The neural network model with 13 connections is shown in Fig. 15. Interestingly, only x_4 is used for classification. From this neural network, we can extract the following three rules:

$$\begin{array}{rl} \text{If } x_4 \le 0.6, & \text{then Iris setosa} \\ \text{If } 1.1 \le x_4 \le 1.6, & \text{then Iris versicolor} \\ \text{If } x_4 \ge 1.7, & \text{then Iris virginica.} \end{array}$$

The correct classification rate is 91.3% on all instances. Note that the classification rate is almost the same when the threshold is set to 0.5 on the Iris data.

From the three benchmark problems, we can conclude that by trading off accuracy against complexity, the Pareto-based multiobjective optimization algorithm is able to find the simplest structures that solve the problem best. Besides, the simple Pareto-optimal networks are able to capture the main knowledge embedded in the data so that interpretable logic rules can be extracted. Compared to other methods used in extracting rules from trained neural network [79], [80], the Pareto-based approach is very straightforward and efficient. Besides, the multiple interpretable yet Pareto-optimal solutions provide additional knowledge that can help the user understand the problem, as we have shown on the three benchmark problems.

B. Model Selection by Analyzing the Pareto Front

Model selection is a well-studied topic in machine learning [81], [82]. If sufficient data are available, the best approach to model selection is to split the data into three subsets, where the first subset (training data) is for constructing models, the second one (validation data) is used to estimate prediction error for selecting a model, and the third one (test data) for accessing the generalization error of the selected model. In case of insufficient data, which is often the case in real-world applications, either analytical methods such as the information-theoretic criteria [81], [82], e.g., the Akaike's information criterion (AIC) and the Bayesian information criterion (BIC), or resampling techniques like k-fold cross-validation [82], are used.

In this section, we show that the Pareto approach to handling the accuracy–complexity tradeoff provides an empirical, yet interesting alternative to selecting models that have good

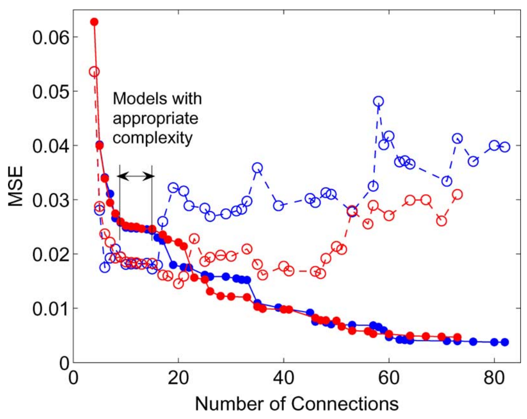


Fig. 16. Accuracy versus complexity of the Pareto-optimal solutions from two independent runs: breast cancer data. Dots denote training data and circles test data.

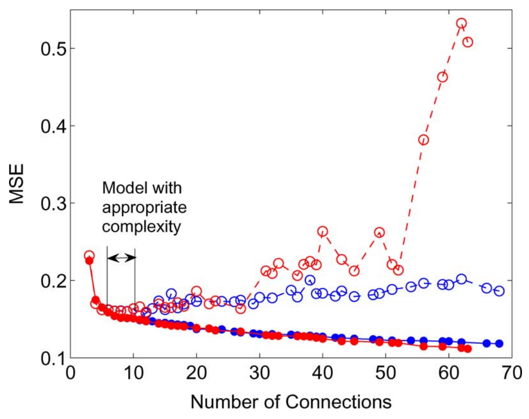


Fig. 17. Accuracy versus complexity of the Pareto-optimal solutions from two independent runs: diabetes data. Dots denote training data and circles test data.

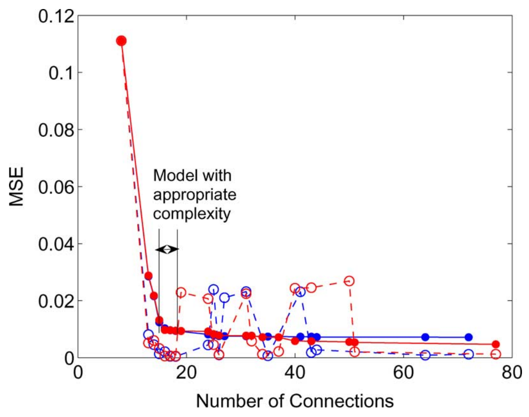


Fig. 18. Accuracy versus complexity of the Pareto-optimal solutions from two independent runs: Iris data. Dots denote training data and circles test data.

generalization on unseen data. The basic argument is that the complexity of the model should match that of the data to be learned and the ability of the learning algorithm. When the complexity of the model is overly large, learning becomes sensitive to stochastic influences, and results on unseen data will be unpredictable, i.e., overfitting can happen. Inspired by the work on determining the correct number of clusters in multiobjective data clustering [71], the appropriate complexity of the data can be determined by the *normalized performance gain* (NPG)

$$\mathrm{NPG} = \frac{\mathrm{MSE}_j - \mathrm{MSE}_i}{C_i - C_j} \qquad (41)$$

where MSE_i, MSE_j, and C_i, C_j are the MSE on training data, and the NC of the ith and jth Pareto optimal solutions. When the solutions are ranked in the order of increasing complexity, the following relationships hold:

$$\begin{aligned} C_{i+1} &> C_i \\ \mathrm{MSE}_{i+1} &\leq \mathrm{MSE}_i. \end{aligned}$$

We hypothesize that if the model complexity is lower than that of the data, an increase in complexity will result in significant increase in performance (NPG). As the complexity continues to increase, the NPG decreases gradually to zero. At this point, the complexity of the model matches that of the data. Further increase in complexity will probably bring about further enhancement in performance on the training data, but with the increasing risk of overfitting the training data.

We are now going to verify empirically the suggested method for model selection on the three benchmark problems. In this part of the simulations, available data are split into a training dataset and a test dataset. For the breast cancer data, 525 instances are used for training and 174 instances for test. The training set of the diabetes data contains 576 samples and the test set 192 samples. Finally, 120 instances are used for training and the rest 30 instances for test for the Iris data.

The Pareto fronts generated from two independent runs on the three benchmark problems are presented in Figs. 16, 17, and 18, respectively. The dots denote the results on the training dataset, while the circles the results on test data. The NPG from the two independent runs for the three problems are plotted in Figs. 19, 20, and 21, respectively.

We first analyze the results on the breast cancer data. From Fig. 19, we notice that the NPG decreases to 0 after the first peak in performance gain when the NC is between 12 and 14. Meanwhile, it can be seen from Fig. 16 that the learning performance on the training data from different runs begins to fluctuate when the NC is larger than 17. These two facts suggest that the appropriate complexity of the neural network for this problem is between 12 and 17. We can see from Fig. 16 that the error on the test data is well controlled when the complexity is in the suggested range.

Similar observations can be made on the diabetes data and the Iris data. For the diabetes data, the NPG first drops to 0 when the NC of the neural networks is around 10. In addition, a discrepancy between the two runs becomes large after the NC reaches 13. From these two observations, we conclude that the complexity of the neural network on the diabetes data should be around 8–10. For the same reasons, the NC of the neural network should be between 16 and 18 for the Iris data.

The proposed method for model selection is empirical and needs to be verified on more problems. For clarity, we only plot results from two independent runs in the aforementioned

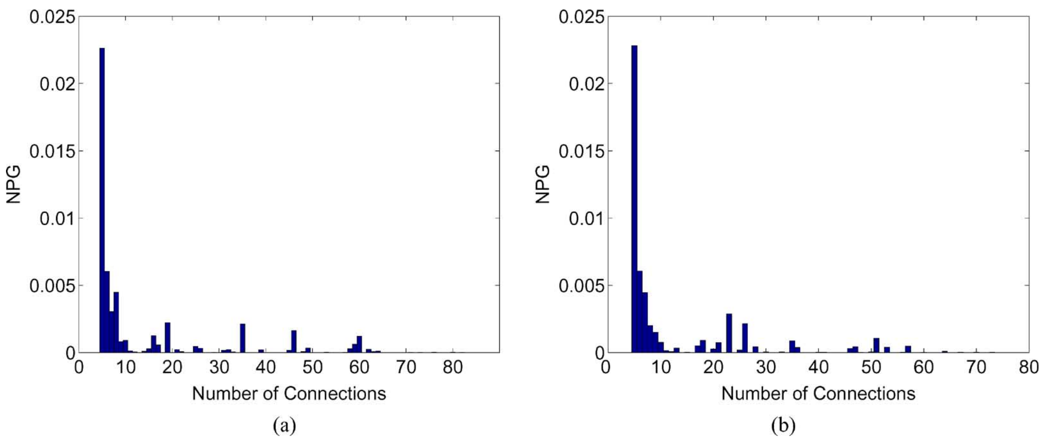


Fig. 19. NPG from two independent runs for the breast cancer data.

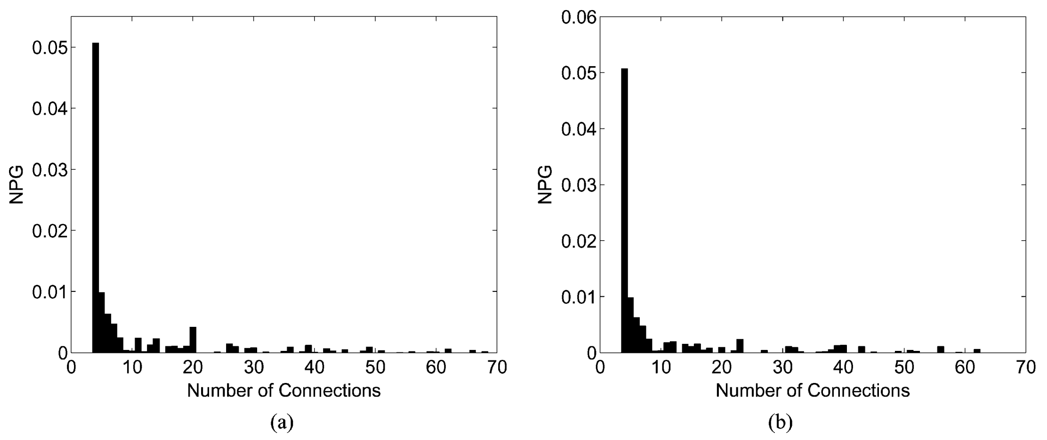


Fig. 20. NPG from two independent runs for the diabetes data.

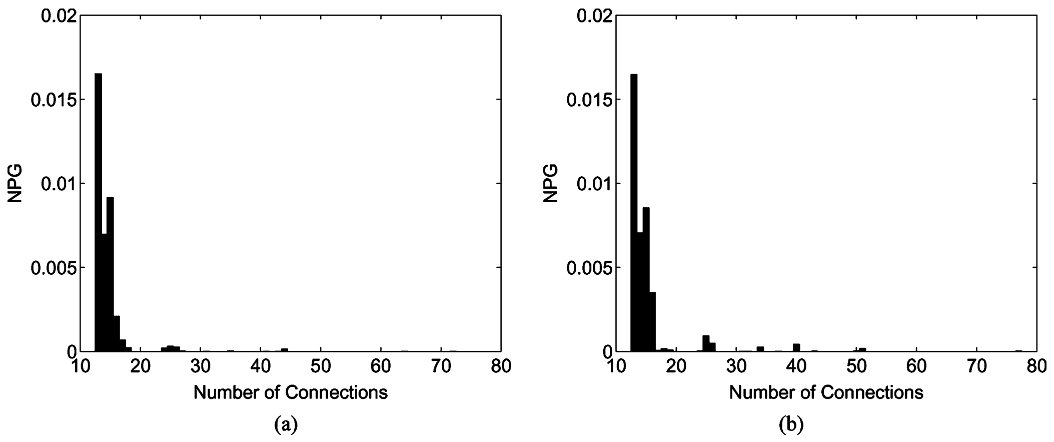


Fig. 21 NPG from two independent runs for the Iris data.

analyses. The results of ten independent runs are plotted in Figs. 28–30. From these results, we can confirm that the generalization performance of the neural network is good when the learning performance on the training data is stable in different runs.

It is difficult to select one single model using our empirical method. Instead, it will be more reliable if multiple models of potentially good generalization performance are chosen to construct an ensemble. This topic will be discussed in the next section.

C. Generating Diverse and Accurate Ensemble Members

In this section, we compare three Pareto-based multiobjective approaches to ensemble generation. The first approach is

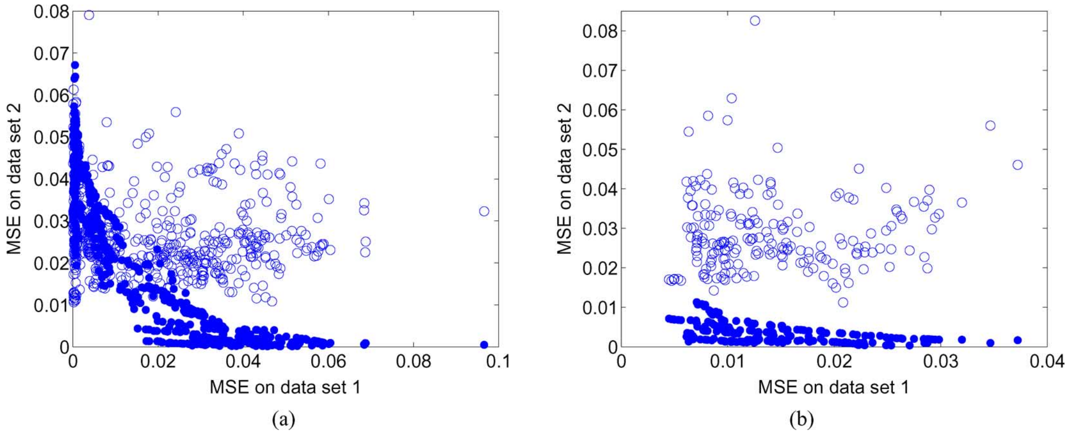


Fig. 22. Achieved nondominated solutions using Abbass's approach: breast cancer data. (a) Lifetime learning is switched between two datasets. (b) Lifetime learning applied on the combination of the data.

presented in Abbass [57], where the accuracies on two datasets serve as two objectives. The second one is described in Chandra and Yao [52], where a tradeoff between accuracy and diversity is taken into account to generate ensembles. The final approach studied in the section is suggested in Jin *et al.* [25], [26], in which the accuracy and the NC of the neural network are adopted as two conflicting objectives. The experimental setup is the same as in the previous studies, except that in the Abbass' approach, the training data of the three benchmark problems are equally divided into two datasets so that the approximation errors on the two datasets can be computed as the two objectives.

Another issue, which has not been explicitly addressed in Abbass [57], is the lifetime learning under the context of multiobjective learning. Note that RProp is adopted as the lifetime learning algorithm, which works for single-objective learning only. This is not a problem in Chandra's as well as in Jin's approach in that the lifetime learning is applied to one of the objective only. However, when both objectives are approximation errors, multiobjective lifetime learning should be applied, which is not straightforward for gradient-based learning algorithms. In Jin *et al.* [83], it is suggested that the lifetime learning should be switched randomly between the two objectives to achieve diverse Pareto-optimal solutions. In this study, lifetime learning is switched between the two objectives at an equal probability. For comparison, simulations are also conducted where the lifetime learning is of single-objective nature, i.e., the RProp is applied on the combination of the two datasets.

The Pareto-optimal solutions from ten runs on the breast cancer data are plotted in Fig. 22(a), where the lifetime learning is switched between the two datasets, and Fig. 22(b), where the lifetime learning is applied on the combination of the two datasets. In the figures, the dots denote the results on the training data and the circles the results on test data. From these results, we can make the following observations. First, by switching the lifetime learning between the two datasets, more diverse solutions can be achieved. Second, good performance on the training data does not ensure good performance on the test data. As suggested in [53], ensemble members should be both accurate and diverse. In other words, ensembles whose members are of poor accuracy cannot perform well. This indicates that if the Pareto-optimal solutions are used as ensemble members, the quality of the ensemble will be poor. Third, lifetime learning on the combination of the data results in serious overfitting.

Very similar results are obtained for the diabetes data and the Iris data, which are plotted in Figs. 23 and 24, respectively. Again, it is difficult to choose proper ensemble members from the Pareto-optimal solutions.

The simulation results using Chandra and Yao's approach for the three benchmark problems are presented in Figs. 25, 26, and 27, respectively. Again, the results on the training and test data are denoted by dots and circles. From the figures, we find that the achieved Pareto-optimal network models tend to overfit the data regardless the diversity, particularly on the diabetes data and the Iris data.

Finally, we take a look at Jin *et al.*'s approach, which ensures the diversity of the ensemble members by generating neural networks with different complexities. The results are presented in Figs. 28, 29, and 30, respectively. From the figures, we can see that in all the three examples, the MSE on the test data is well constrained when the complexity of the neural network is appropriately low. As suggested in the previous section, the required complexity that matches the data can be estimated using the NPG. By choosing these networks as ensemble members, we are able to have a neural network ensemble whose members are both accurate and diverse. The diversity of the networks is guaranteed by the difference in the complexity of the neural networks.

Comparing the three Pareto-based approaches to ensemble generation, we conclude that it is not straightforward to choose ensemble members from the achieved Pareto-optimal solutions for constructing ensembles in Abbass' as well as in Chandra and Yao's approaches. To ensure good performance on unseen data of the ensemble, additional methods such as cross-validation must be employed. In contrast, it is rather easy to identify neural networks that can be used as ensemble members when Pareto-optimal solutions are generated using Jin *et al.*' approach. Another important point is that in Abbass' and Chandra and Yao's approaches, the achieved solutions from the ten independent

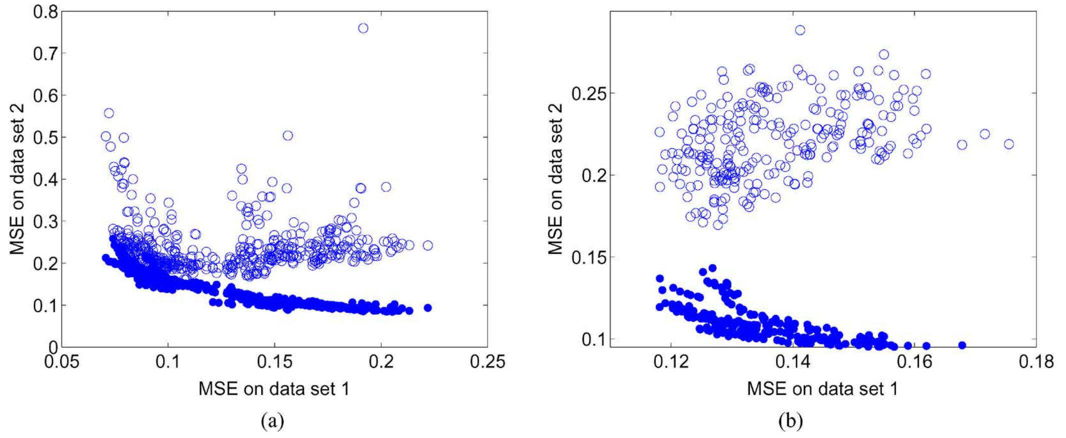


Fig. 23. Achieved nondominated solutions using Abbass's approach: diabetes data. (a) Lifetime learning is switched between two datasets. (b) Lifetime learning applied on the combination of the data.

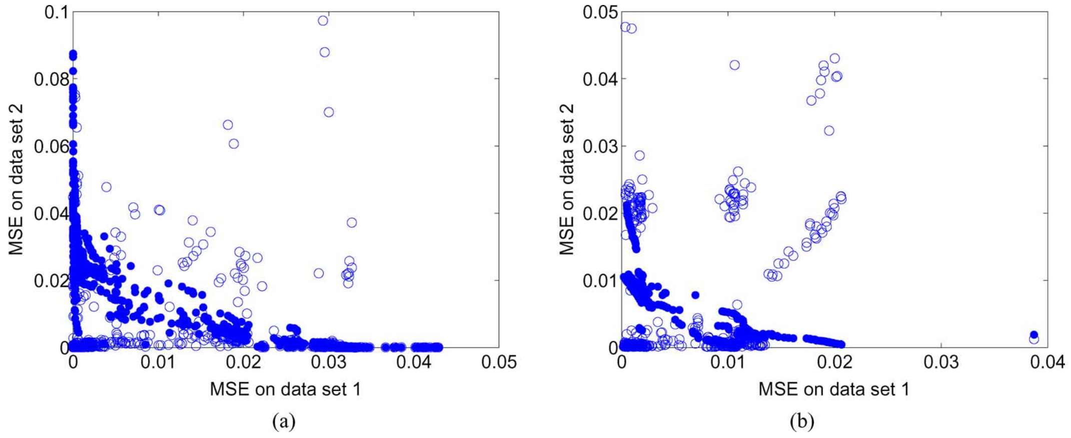


Fig. 24. Achieved nondominated solutions using Abbass's approach: Iris data. (a) Lifetime learning is switched between two datasets. (b) Lifetime learning applied on the combination of the data.

runs are rather different along the whole Pareto front, which means that these two methods are quite sensitive to stochastic influences. Opposite to that, the results from the ten independent runs are quite stable in Jin *et al.*'s approach when the complexity is low.

VII. Summary and Outlook

Pareto-based approach to machine learning provides us a new point of view for studying machine learning problems. By means of Pareto-based optimization, we are able to gain a deeper insight into different aspects of machine learning, and thus, develop new learning algorithms. The power of the Pareto-based approach is made more attractive due to the successful application of evolutionary algorithms to Pareto-based multiobjective optimization.

This paper provides an up-to-date yet not necessarily complete review of the existing research on Pareto-based multiobjective learning algorithms. We illustrate, on three benchmark problems, how we can address important topics in machine learning, such as generating interpretable models, model selection for generalization, and ensemble generation, using the Pareto-based multiobjective approach. We show that the simplest Pareto-optimal model without any input selected approximates the mean of the training data, while the simple Pareto-optimal models with one or two most important features selected capture the essential knowledge in the data. In addition, we demonstrate empirically that by analyzing the Pareto-optimal solutions in terms of performance and complexity, and the learning performance w.r.t. model complexity in independent runs, we are able to choose models that are most likely to exhibit good performance on unseen data. Finally, we compare three Pareto-based approaches to the generation of neural ensembles and indicate that the method by trading off accuracy and complexity can provide reliable results.

Many issues remain to be resolved and new areas could be opened up in the field of Pareto-based multiobjective machine learning. One interesting question is how the Pareto-based approach to machine learning influences the learning behavior, e.g., the property of the learning curve [84], [85]. It has

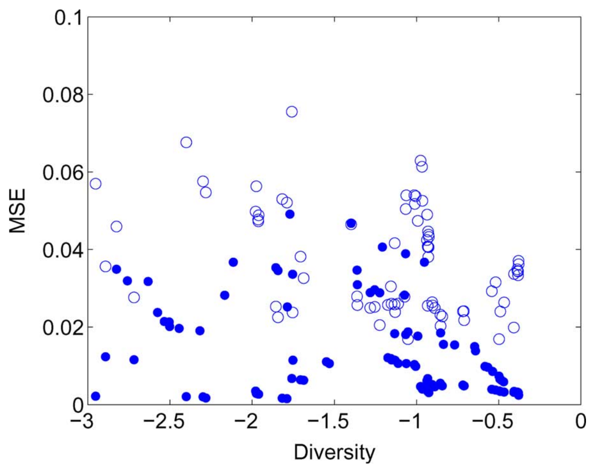


Fig. 25. Achieved nondominated solutions using Chandra and Yao's approach: breast cancer data.

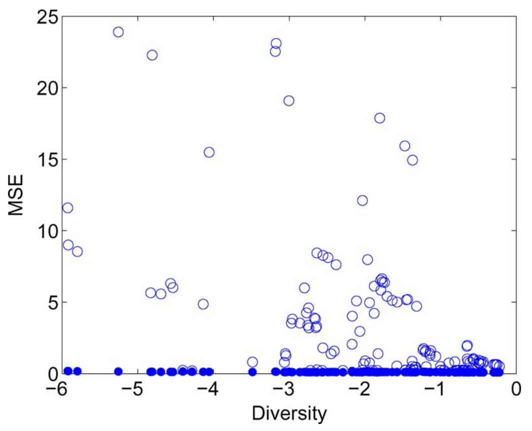


Fig. 26. Achieved nondominated solutions using Chandra and Yao's approach: diabetes data.

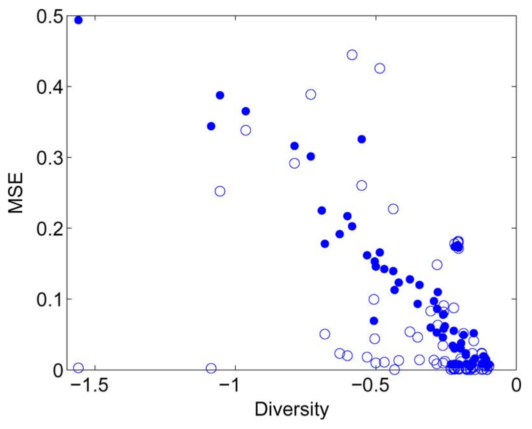


Fig. 27. Achieved nondominated solutions using Chandra and Yao's approach: Iris data.

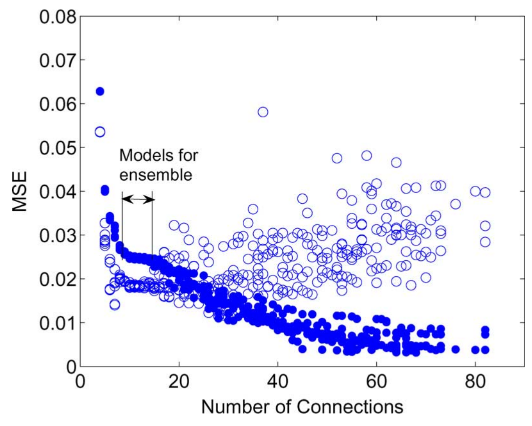


Fig. 28. Achieved nondominated solutions using Jin *et al.*'s approach: breast cancer data.

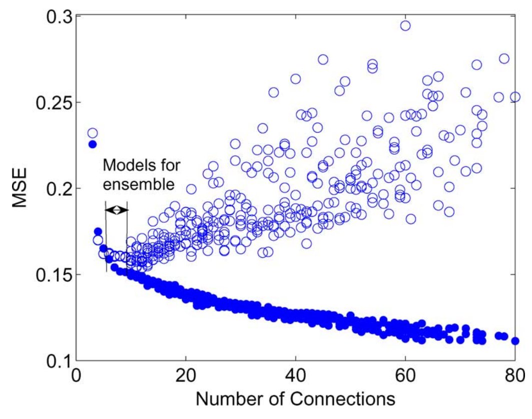


Fig. 29. Achieved nondominated solutions using Jin *et al.*'s approach: diabetes data.

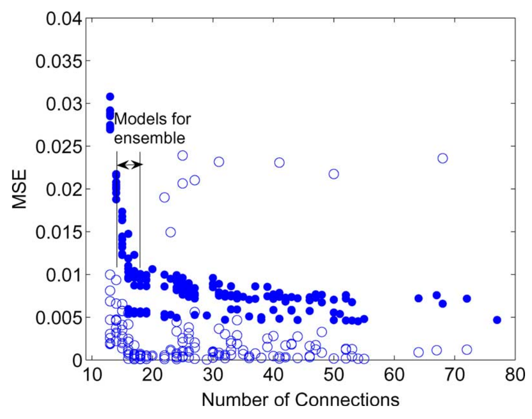


Fig. 30. Achieved nondominated solutions using Jin *et al.*'s approach: Iris data.

been empirically disclosed in general optimization problems that the number of local optima can be reduced by converting multimodal single-objective problems into multiobjective ones [86], [87]. If we are able to show that the same happens in machine learning, it is then more convincing to argue that the Pareto-based multiobjective learning is able to improve learning performance.

Most topics discussed so far are mainly concerned with the bias–variance tradeoff in machine learning. Another important topic in machine learning, as well as in human memory systems, is the plasticity–stability tradeoff, which is also known as online learning [88], incremental learning [89], or catastrophic forgetting [90]. A preliminary attempt has been made in [83] to address catastrophic forgetting using the Pareto-based approach. It has been shown that the multiobjective approach is more promising in alleviating forgetting than its single-objective counterpart. The idea of Pareto-optimality can also be extended to the study on connectivity and complexity [91], [92] of general networks, and to the research on structure and functionality of spiking neural networks [93], [94].

ACKNOWLEDGMENT

The authors are grateful to E. Körner for his kind support.

REFERENCES

[1] E. Alpaydin, *Introduction to Machine Learning*. Cambridge, MA: MIT Press, 2004.

[2] T. M. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

[3] Z. Zoltan, Z. Kalmar, and C. Szepesvari, "Multi-criteria reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 1998, pp. 197–205.

[4] Y. Jin, B. Sendhoff, and E. Körner, "Evolutionary multi-objective optimization for simultaneous generation of signal-type and symbol-type representations," in *Proc. 3rd Int. Conf. Evol. Multi-Criterion Optim.* Lecture Notes in Computer Science, vol. 3410. New York: Springer-Verlag, 2005, pp. 752–766.

[5] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural network architectures," *Neural Comput.*, vol. 7, pp. 219–269, 1995.

[6] Y. Jin, "Fuzzy modeling of high-dimensional systems: Complexity reduction and interpretability improvement," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 2, pp. 212–221, 2000.

[7] Y. Jin, W. V. Seelen, and B. Sendhoff, "On generating FC^3 fuzzy rule systems from data using evolution strategies," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 6, pp. 829–845, Dec. 1999.

[8] Y. Liu and X. Yao, "Ensemble learning via negative correlation," *Neural Netw.*, vol. 12, pp. 1399–1404, 1999.

[9] C. Cortes and V. Vapnik, "Support vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, 1995.

[10] B. Olhausen and D. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?," *Vis. Res.*, vol. 37, pp. 3311–3325, 1997.

[11] T. Fawcett, "ROC graphs: Notes and practical considerations for data mining researchers," HP Labs., Palo Alto, CA, Tech. Rep. HPL-2003-4, 2003.

[12] J. Banfield and A. Raftery, "Model-based Gaussian and non-Gaussian clustering," *Biometrics*, vol. 49, pp. 803–821, 1993.

[13] M. H. C. Law, A. P. Topchy, and A. K. Jain, "Multiobjective data clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* Piscataway, NJ: IEEE Press, 2004, vol. 2, pp. 424–430.

[14] I. Das and J. Dennis, "A closer look at drawbacks of minimizing weighted sum of objectives for Pareto set generation in multicriteria optimization problems," *Struct. Optim.*, vol. 14, no. 1, pp. 63–69, 1997.

[15] Y. Jin, M. Olhofer, and B. Sendhoff, "Dynamic weighted aggregation for evolutionary multi-objective optimization: Why does it work and how?," in *Proc. Genetic Evol. Comput. Conf.* San Mateo, CA: Morgan Kaufmann, 2001, pp. 1042–1049.

[16] H. Abbass, "Speeding up back-propagation using multi-objective evolutionary algorithms," *Neural Comput.*, vol. 15, no. 11, pp. 2705–2726, 2003.

[17] R. de A. Teixeira, A. Braga, R. Takahashi, and R. Saldanha, "Improving generalization of MLP with multi-objective optimization," *Neurocomputing*, vol. 35, pp. 189–194, 2000.

[18] Y. Jin, Ed., *Multi-Objective Machine Learning*. New York: Springer-Verlag, 2006.

[19] G. Liu and V. Kadirkamanathan, "Learning with multi-objective criteria," in *Proc. Inst. Electr. Eng. Conf. Artif. Neural Netw.*, 1995, pp. 53–58.

[20] Y. Matsuyama, "Harmonic competition: A self-organizing multiple criteria optimization," *IEEE Trans. Neural Netw.*, vol. 7, no. 3, pp. 652–668, May 1996.

[21] K. Kottathra and Y. Attikiouzel, "A novel multicriteria optimization algorithm for the structure determination of multilayer feedforward neural networks," *J. Netw. Comput. Appl.*, vol. 19, pp. 135–147, 1996.

[22] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, U.K.: Wiley, 2001.

[23] T. Furukawa, "Parameter identification with weightless regularization," *Int. J. Numer. Methods Eng.*, vol. 52, pp. 219–238, 2001.

[24] H. Abbass, "A memetic Pareto approach to artificial neural networks," in *Proc. 14th Aust. Joint Conf. Artif. Intell.*, 2001, pp. 1–12.

[25] Y. Jin, T. Okabe, and B. Sendhoff, "Evolutionary multi-objective approach to constructing neural network ensembles for regression," in *Applications of Evolutionary Multi-Objective Optimization*, C. Coello Coello, Ed. Singapore: World Scientific, 2004, pp. 653–672.

[26] Y. Jin, T. Okabe, and B. Sendhoff, "Neural network regularization and ensembling using multi-objective evolutionary algorithms," in *Proc. Congr. Evol. Comput.* Piscataway, NJ: IEEE Press, Jun. 2004, pp. 1–8.

[27] J. Fieldsend and S. Singh, "Pareto multi-objective non-linear regression modelling to aid CAPM analogous forecasting," in *Proc. Int. Joint Conf. Neural Netw.*, 2002, vol. 1, pp. 388–393.

[28] J. Fieldsend and S. Singh, "Pareto evolutionary neural networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 338–354, Mar. 2005.

[29] N. Garcia-Pedrajas, C. Hervas-Martinez, and J. Munoz-Perez, "Multi-objective cooperative coevolution of artificial neural networks," *Neural Netw.*, vol. 15, no. 10, pp. 1255–1274, 2002.

[30] G. G. Yen and H. Lu, "Hierarchical rank density genetic algorithm for radial-basis function neural network design," in *Proc. Congr. Evol. Comput.*, 2002, vol. 1, pp. 25–30.

[31] T. Hatanaka, N. Kondo, and K. Uosaki, "Multi-objective structure selection for radial basis function networks based on genetic algorithm," in *Proc. Congr. Evol. Comput.*, 2003, pp. 1095–1100.

[32] J. Bi, "Multi-objective programming in SVMs," in *Proc. 20th Int. Conf. Mach. Learn.*, Washington, DC, 2003, pp. 35–42.

[33] C. Igel, "Multi-objective model selection for support vector machines," in *Evolution Multi-Criterion Optimization* Lecture Notes in Computer Science, vol. 3410. New York: Springer-Verlag, 2005, pp. 534–546.

[34] H. Nakayama and T. Asada, "Support vector machines formulated as multi-objective linear programming," in *Proc. ICOTA*, 2001, pp. 1171–1178.

[35] D. Kim, "Structural risk minimization on decision trees using an evolutionary multiobjective optimization," in *Proc. Eur. Conf. Genetic Program.* Lecture Notes in Computer Science, vol. 3003. New York: Springer-Verlag, 2004, pp. 338–348.

[36] E. Bernado-Manssilla and J. Garrell-Guii, "MOLeCS: Using multi-objective evolutionary algorithms for learning," in *Proc. EMO 2001* Lecture Notes in Computer Science, vol. 1993. New York: Springer-Verlag, pp. 696–710.

[37] S. Wiegand, C. Igel, and U. Handmann, "Evolutionary multiobjective optimization of neural networks fore face detection," *Int. J. Comput. Intell.*, vol. 4, no. 3, pp. 237–253, 2005.

[38] Y. Zhang and P. I. Rockett, "Evolving optimal feature extraction using multi-objective genetic programming: A methodology and preliminary study on edge detection," in *Proc. Genetic Evol. Comput. Conf.*, 2005, pp. 795–802.

[39] J. Teo and H. Abbass, "Automatic generation of controllers for embodied legged organisms: A Pareto evolutionary multi-objective approach," *Evol. Comput.*, vol. 12, no. 3, pp. 355–394, 2004.

[40] M. Luque, O. Cordon, and E. Herrera-Viedma, "A multi-objective genetic algorithm for learning linguistic persistent quries in text retrieval environments," in *Multi-Objective Machine Learning*, Y. Jin, Ed. New York: Springer-Verlag, 2006, ch. 25, pp. 585–600.

[41] Y. Jin, *Advanced Fuzzy Systems Design and Applications*. Heidelberg, Germany: Physica Verlag, 2003.

[42] H. Ishibuchi, T. Murata, and I. Türksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern recognition," *Fuzzy Sets Syst.*, vol. 89, pp. 135–150, 1997.

[43] A. Gomez-Skarleta, F. Jimenez, and J. Ibanez, "Pareto-optimality in fuzzy modeling," in *Proc. 6th Eur. Congr. Int. Tech. Soft Comput.*, 1997, pp. 694–700.

[44] T. Suzuki, T. Furuhashi, S. Matsushima, and H. Tsutsui, "Efficient fuzzy modeling under multiple criteria by using genetic algorithms," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 1999, vol. 5, pp. 314–319.

[45] H. Ishibuchi, T. Nakashima, and T. Murata, "Three-objective genetics-based machine learning for linguistic rule extraction," *Inf. Sci.*, vol. 136, no. 1–4, pp. 109–133, 2001.

[46] K. Tachibana and T. Furuhashi, "A structure identification method of submodels for hierarchical fuzzy modeling using the multiple objective genetic algorithm," *Int. J. Intell. Syst.*, vol. 17, pp. 495–513, 2001.

[47] F. Jimenez, G. Sanchez, A. F. Gomez-Skarmeta, H. Roubos, and R. Babuska, "Fuzzy modeling with multi-objective neuro-evolutionary algorithms," in *Proc. IEEE Int. Conf. Syst., Man, Cybern*, 2002, vol. 3, pp. 253–258.

[48] H. Wang, S. Kwong, Y. Jin, W. Wei, and K. Man, "Agent-based evolutionary approach to interpretable rule-based knowledge extraction," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 35, no. 2, pp. 143–155, May 2005.

[49] H. Wang, S. Kwong, Y. Jin, W. Wei, and K. Man, "Multi-objective hierarchical genetic algorithm for interpretable fuzzy rule absed knowledge extraction," *Fuzzy Sets Syst.*, vol. 149, no. 1, pp. 149–186, 2005.

[50] U. Markowska-Kaczmar and P. Wnuk-Liplnski, "Rule extraction from neural networks with Pareto optimization," in *Artificial Intelligence and Soft Computing*. Berlin, Germany: Springer-Verlag, 2004, pp. 450–455.

[51] L. Hansen and P. Salammon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 10, pp. 993–1101, Oct. 1990.

[52] A. Chandra and X. Yao, "DIVACE: Diverse and accurate ensemble learning algorithm," in *Proc. 5th Int. Conf. Intell. Data Eng. Autom. Learn.*, 2004, pp. 619–625.

[53] D. Opitz and J. Shavlik, "Generating accurate and diverse members of a neural network ensemble," in *Advances in Neural Information Processing Systems 8*. Cambridge, MA: MIT Press, 1996, pp. 535–541.

[54] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: A survey and categorisation," *J. Inf. Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[55] B. Rosen, "Ensemble learning using decorrelated neural networks," *Connection Sci.*, vol. 8, pp. 373–384, 1996.

[56] A. Chandra and X. Yao, "Evolving hybrid ensembles of learning machines for better generalisation," *Neurocomputing*, vol. 69, pp. 686–700, 2006.

[57] H. Abbass, "Pareto neuro-evolution: Constructing ensemble of neural networks using multi-objective optimization," in *Proc. Congr. Evol. Comput*, Dec. 2003, pp. 2074–2080.

[58] N. Garcia-Pedrajas, C. Hervas-Martinez, and J. Munoz-Perez, "COVNET: A cooperative coevolutionary model for evolving artificial neural networks," *IEEE Trans. Neural Netw.*, vol. 14, no. 3, pp. 575–596, May 2003.

[59] L. Kuncheva and C. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Mach. Learn.*, vol. 51, no. 2, pp. 181–207, 2003.

[60] T. Hatanaka, N. Kondo, and K. Uosaki, "Multiobjective structure selection for RBF networks and its application to nonlinear system design," in *Multi-Objective Machine Learning*, Y. Jin, Ed. New York: Springer-Verlag, 2006, ch. 21, pp. 491–505.

[61] H. Ishibuchi and T. Yamamoto, "Evolutionary multi-objective optimization for generating an ensemble of fuzzy rule-based classifiers," in *Proc. Genetic Evol. Comput. Conf.* Lecture Notes in Computer Science, vol. 2723, 2003, pp. 1077–1088.

[62] M. Kupinski and M. Anastasio, "Multiobjective genetic optimization of diagnostic classifiers with implementations for generating receiver operating characteristic curves," *IEEE Trans. Med. Imag.*, vol. 18, no. 8, pp. 675–685, Aug. 1999.

[63] J. Horn and N. Nafpliotis, "Multiobjective optimization using the niched Pareto genetic algorithms," IlliGAL, Univ. Illinois at Urbana-Champaign, Urbana-Champaign, Tech. Rep. 93005, 1993.

[64] L. Gräning, Y. Jin, and B. Sendhoff, "Generalization improvement in multi-objective learning," in *Proc. Int. Joint Conf. Neural Netw.*, 2006, pp. 9893–9900.

[65] R. Everson and J. Fieldsend, "Multi-class ROC analysis from a multi-objective optimisation perspective," *Pattern Recognit. Lett.*, vol. 27, pp. 918–927, 2006.

[66] S. Park, D. Nam, and C. H. Park, "Design of a neural controller using multi-objective optimization for nonminimum phase systems," in *Proc. IEEE Int. Conf. Fuzzy Sets Syst.*, 1999, vol. I, pp. 533–537.

[67] O. Cordon, F. Herrera, M. del-Jesus, and P. Villar, "A multi-objective genetic algorithm for feature selection and granularity learning in fuzzy-rule based classification systems," in *Proc. Joint 9th IFSA World Congr. 20th NAFIPS Int. Conf.*, 2001, vol. 3, pp. 1253–1258.

[68] C. Emmanouilidis, A. Hunter, J. MacIntyre, and C. Cox, "Selecting features in neurofuzzy modelling by multiobjective genetic algorithms," in *Proc. Int. Joint Conf. Neural Netw.*, 1999, pp. 749–754.

[69] L. Oliveira, R. Sabourin, F. Bortolozzi, and C. Suen, "Feature seelction for ensembles: A hierarchical multi-objective genetic algorithm approach," in *Proc. 7th Int. Conf. Anal. Recognit.*, 2003, pp. 676–680.

[70] Y. Kim, W. Street, and F. Menczer, "Evolutionary model selection in unsupervised learning," *Intell. Data Anal.*, vol. 6, pp. 531–556, 2002.

[71] J. Handl and J. Knowles, "Exploiting the tradeoff—The benefits of multiple objectives in data clustering," in *Evolutionary Multi-Criterion Optimization* Lecture Notes in Computer Science, vol. 3410. New York: Springer-Verlag, 2005, pp. 547–560.

[72] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a dataset via the gap statistics," *J. R. Stat. Soc. B*, vol. 63, pp. 411–423, 2001.

[73] X. Yao, "Evolving artificial neural networks," *Proc. IEEE*, vol. 87, no. 9, pp. 1423–1447, Sep. 1999.

[74] C. Igel and M. Hüsken, "Empirical evaluation of the improved Rprop learning algorithm," *Neurocomputing*, vol. 55, no. C, pp. 105–123, 2003.

[75] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropgation learning: The RPROP algorithm," in *Proc. IEEE Int. Conf. Neural Netw.*, 1993, vol. 1, pp. 586–591.

[76] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Proc. Parallel Problem Solving Nat.*, 2000, vol. VI, pp. 849–858.

[77] L. Prechelt, "PROBEN1—A set of neural network benchmark problems and benchmarking rules," Fakultät Inf., Univ. Karlsruhe, Karlsruhe, Germany, Tech. Rep., 1994.

[78] S. Thrun, "Extracting rules from artificial neural networks with distributed representation," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 1994, pp. 505–512.

[79] S. Kamruzzaman and M. Islam, "Extraction of symbolic rules from artificial neural networks," *Trans. Eng., Comput. Technol.*, vol. 10, pp. 271–277, 2005.

[80] R. Setiono, "Generating concise and accurate classification rules for breast cancer disgnosis," *Artif. Intell. Med.*, vol. 18, pp. 205–219, 2000.

[81] K. Burnham and D. Anderson, *Model Selection and Multimodel Inference*. New York: Springer-Verlag, 2002.

[82] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer-Verlag, 2001.

[83] Y. Jin and B. Sendhoff, "Alleviating catastrophic forgetting via multi-objective learning," in *Proc. Int. Joint Conf. Neural Netw.*, 2006, pp. 6367–6374.

[84] S. Amari, "A universal theorem on learning curves," *Neural Netw.*, vol. 6, no. 2, pp. 161–166, 1993.

[85] T. Fine and S. Mukherjee, "Parameter convergence and learning curves for neural networks," *Neural Comput.*, vol. 11, pp. 747–769, 1999.

[86] S. Louis and G. Rawlins, "Pareto optimality, GA-easiness and deception," in *Proc. Int. Conf. Genetic Algorithms*, 1993, pp. 118–123.

[87] J. Knowles, R. Watson, and D. Corne, "Reducing local optima in single-objective problems by multi-objectivization," in *Proc. 1st Int. Conf. Evol. Multi-Criterion Optim.* Lecture Notes in Computer Science, vol. 1993. New York: Springer-Verlag, 2001, pp. 269–283.

[88] V. de Angulo and C. Torras, "On-line learning with minimal degradation in feedforward networks," *IEEE Trans. Neural Netw.*, vol. 6, no. 3, pp. 657–668, May 1995.

[89] S. Wan and L. Banta, "Parameter incremental learning algorithm for neural networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1424–1438, Nov. 2006.

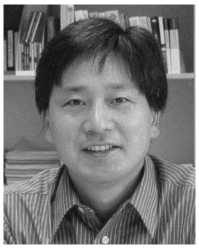
[90] M. Frean and A. Robins, "Catastrophic forgetting in simple networks: An analysis of the pseudo-rehearsal solution," *Netw.: Comput. Neural Syst.*, vol. 10, pp. 227–236, 1999.

[91] R. Adams, L. Calcraft, and N. Davey, "Connectivity in real and evolved associative memories," in *Proc. Brain Inspired Cognit. Syst.*, 2006, pp. 153–159.

[92] O. Sporns and G. Tononi, "Classes of network connectivity and dynamics," *Complexity*, vol. 7, pp. 28–38, 2002.

[93] Y. Jin, R. Wen, and B. Sendhoff, "Evolutionary multi-objective optimization of spiking neural networks," in *Proc. Int. Conf. Artif. Neural Netw. (ICANN)* Part I, Lecture Notes in Computer Science, vol. 4668. New York: Springer-Verlag, 2007, pp. 370–379.

[94] W. Maass and C. Bishop, *Pulsed Neural Networks*. Cambridge, MA: MIT Press, 1999.



Yaochu Jin (M'98–SM'02) received the B.Sc., M.Sc., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1988, 1991, and 1996, respectively, all in automatic control, and the second Ph.D. degree in electrical and computer engineering from Ruhr-Universität Bochum, Bochum, Germany, in 2001.

In 1991, he joined the Electrical Engineering Department, Zhejiang University, where he became an Associate Professor in 1996. From 1996 to 1998, he was with the Institut für Neuroinformatik, Ruhr-Universität Bochum, first as a Visiting Researcher and then as a Research Associate. From 1998 to 1999, he was a Postdoctoral Associate with the Industrial Engineering Department, Rutgers, the State University of New Jersey, Piscataway. In 1999, he joined the Future Technology Research Division, Honda R&D Europe, Offenbach/Main, Germany. Since 2003, he has been a Principal Scientist and a Project Leader at Honda Research Institute Europe, Offenbach. His current research interests include computational approaches to understanding evolution and learning in biology (computational biology), and evolutionary and learning approaches to complex systems design (computational intelligence). He is the coeditor of *Evolutionary Computation in Danamic and Uncertain Environments* (Springer-Verlag, 2007), the editor of *Multi-Objective Machine Learning* (Springer-Verlag, 2006) and *Knowledge Incorporation in Evolutionary Computation* (Springer-Verlag, 2005), and the author of *Advanced Fuzzy Systems Design and Applications* (Springer-Verlag, 2003). He has also authored or coauthored more than 80 journal and conference papers.

Dr. Jin is currently an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS, the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, PART C, and the IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE. He has been a Guest Editor of five journal special issues, including one on Evolutionary Optimization in the Presence of Uncertainties in the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION.

Bernhard Sendhoff (M'99–SM'05) received the Diploma and the Doctorate degree in physics from Ruhr-Universität Bochum, Bochum, Germany, in 1993 and 1998, respectively.

From 1998 to 1999, he was a Research Assistant at the Institute for Neuroinformatics. From 1999 to 2002, he was with Honda R&D Europe GmbH, Offenbach/Main, Germany, where he has last been a Deputy Division Manager. He is currently the Chief Technology Officer of Honda Research Institute Europe GmbH, Offenbach, where he is also the Head of the Evolutionary and Learning Technology Group. He is also an Honorary Professor in the School of Computer Science, University of Birmingham, Birmingham, U.K. His current research interests include topics from systems biology and computational intelligence such as evolutionary system design and structure optimization of adaptive systems. He is the author or coauthor of more than 100 research papers published in journals and refereed conferences.

Dr. Sendhoff is a member of the Association for Computing Machinery, the European Neural Network Society, and the Deutsche Physikalische Gesellschaft (DPG). He is also a member of several advisory boards and scientific committees.