Published in: The 3rd Int. Conf. on Evolutionary Multi-Criterion Optimization. EMO 2005, LNCS 3410, pp.752-766, Springer

Evolutionary Multi-objective Optimization for Simultaneous Generation of Signal-type and Symbol-type Representations

Yaochu Jin, Bernhard Sendhoff, and Edgar Körner

Honda Research Institute Europe
63073 Offenbach/Main, Germany
yaochu.jin@honda-ri.de

Abstract. It has been a controversial issue in the research of cognitive science and artificial intelligence whether signal-type representations (typically connectionist networks) or symbol-type representations (e.g., semantic networks, production systems) should be used. Meanwhile, it has also been recognized that both types of information representations might exist in the human brain. In addition, symbol-type representations are often very helpful in gaining insights into unknown systems. For these reasons, comprehensible symbolic rules need to be extracted from trained neural networks. In this paper, an evolutionary multi-objective algorithm is employed to generate multiple models that facilitate the generation of signal-type and symbol-type representations simultaneously. It is argued that one main difference between signal-type and symbol-type representations lies in the fact that the signal-type representations are models of a higher complexity (fine representation), whereas symbol-type representations are models of a lower complexity (coarse representation). Thus, by generating models with a spectrum of model complexity, we are able to obtain a population of models of both signal-type and symbol-type quality, although certain post-processing is needed to get a fully symbol-type representation. An illustrative example is given on generating neural networks for the breast cancer diagnosis benchmark problem.

1 Introduction

Artificial neural networks are one of the most well known signal-type representations in cognitive and vision research. Neural networks are linear or nonlinear systems that encode information with connections and units in a distributed manner, which are more or less of biological plausibility. In contrast, symbol-type representations use meaningful symbols, and information is encoded by defining the relationship among various symbols. Several symbolic representation models have been developed, such as semantic networks, production systems (symbolic rules) and finite-state automata.

Symbolic representations and symbolic processing are believed to have several desirable features that are closely related to mental representations and cognition [12], namely, productivity, systematicity, compositionality and inferential

coherence. Besides, increasing evidence has been found in cognitive neuroscience that the human brain does have mechanisms that are responsible for symbolic processing [13, 21, 4]. It is widely believed that symbolic systems are more transparent to human users than connectionist networks, which plays an important role when neural networks are employed in critical engineering applications.

For the above reasons, it is often necessary to extract symbolic or fuzzy rules from trained neural networks [3, 20, 11]. A common drawback of most existing rule extraction method is that the rules are extracted after a neural network has been trained, which incurs additional computational costs.

This paper attempts to generate signal-type and symbol-type models simultaneously using the multi-objective optimization approach. Multiple neural networks of various model complexities, instead of either a single signal-type or symbol-type model, will be generated using a multi-objective evolutionary algorithm combined with a local search, where accuracy and complexity serve as two conflicting objectives. It has been shown that evolutionary multi-objective algorithms are well suited and very powerful in obtaining a set of Pareto-optimal solutions in one single run of optimization [8, 7].

Training neural networks using evolutionary multi-objective optimization is not new in itself [30, 1, 2, 19]. However, the existing work focuses on improving the accuracy of a single network or an ensemble of networks. Generating an ensemble of fuzzy classifiers using evolutionary algorithms has also been studied in [16]. Objectives in training neural networks include accuracy on training data, accuracy on test data, number of hidden neurons, and number of connections. Note that a trade-off between the accuracy on the training data and the accuracy on test data does not necessarily mean a trade-off between accuracy and complexity.

Section 2 discusses very briefly the existing methods for controlling model complexity in the context of model selections in machine learning. Methods for converting signal-type neural networks to symbol-type rules in the area of neural networks will also be introduced. Section 3 shows that any formal neural network regularization methods can be treated as multi-objective optimization problems. The details of the evolutionary multi-objective algorithm, together with the local search method will be provided in Section 4. An illustrative example is given in Section 5, where a population of Pareto-optimal neural networks are generated for the breast diagnosis problem. It will be shown that among the models generated by the multi-objective evolutionary algorithm, those with a higher complexity are of more signal quality and those of a lower complexity are of more symbol quality.

2 Complexity Control and Rule Extraction

2.1 Model Selection and Complexity Control

The task of model selection is to choose the best model for a set of given data, assuming that a number of models is available. Several criteria have been proposed based on the Kullback-Leibler Information Criterion [6]. The most popular

criteria are Akaike's Information Criterion (AIC) and Bayesian Information Criterion (BIC). For example, model selection according to the AIC is to minimize the following criterion:

$$AIC = -2\ log(\mathcal{L}(\theta|y,\ g) + 2\ K, \tag{1}$$

where, $\mathcal{L}(\theta|y,\ g)$ is the maximized likelihood for data y given a model g with model parameter θ, K is the number of effective parameters of g.

The first term of Equation (1) reflects how good the model approximates the data, while the second term is the complexity of the model. Usually, the higher the model complexity is, the more accurate the approximation will be. Obviously, a trade-off between accuracy and model complexity has to be taken into account in model selection.

On the other hand, model selection criteria have often been used to control the complexity of models to a desired degree in model generation. This approach is usually known as regularization in the neural network community [5]. The main purpose of neural network regularization is to improve the generalization capability of a neural network by control its complexity. By generalization, it is meant that a trained neural network should perform well not only on training data, but also on unseen data.

2.2 Complexity Reduction in Rule Extraction

Generally, neural networks are signal-type representations that are difficult to understand for human users. Due to this reason, many efforts have been made to extract symbolic or fuzzy rules from trained neural network [3, 18, 11]. Two assumptions are often made during rule extraction from trained neural networks. First, units in the neural network are either maximally active or inactive. To meet this requirement, regularization techniques such as structural regularization [17], weight sharing [20] or network pruning [28] are usually implemented before rule extraction, which in effect reduces the complexity of neural networks. In fact, this assumption is also essential for the interpretability of the extracted rules. The complexity reduction procedure prior to rule extraction is also termed skeletonization [10]. The second assumption is that a label, or in other words, a meaning needs to be associated with each unit.

Rule extraction from trained neural networks can be seen as a model selection process that trades off between accuracy and interpretability, where preference is put on the interpretability of the model. Thus, the trade-off between accuracy and complexity in model selection also reflects a trade-off between accuracy and interpretability, in other words, a trade-off between signal-type and symbol-type of representations.

Existing methods for extracting symbolic rules from trained neural network can largely be divided into three steps: neural network training, network skeletonization, and rule extraction [11].

3 Multi-objective Optimization Approach to Complexity Reduction

3.1 Neural Network Regularization

Neural network regularization can be realized by including an additional term that reflects the model complexity in the cost function of the training algorithm:

$$J = E + \lambda\Omega, \tag{2}$$

where E is an error function, Ω is the regularization term representing the complexity of the network model, and λ is a hyperparameter that controls the strength of the regularization. The most common error function in training or evolving neural networks is the mean squared error (MSE):

$$E = \frac{1}{N}\sum_{i=1}^{N}(y^d(i) - y(i))^2, \tag{3}$$

where N is the number of training samples, $y^d(i)$ is the desired output of the i-th sample, and $y(i)$ is the network output for the i-th sample. For the sake of clarity, we assume that the neural network has only one output. Refer to [5] for other error functions, such as the Minkowski error or cross-entropy.

Several measures have also been suggested for denoting the model complexity Ω. A most popular regularization term is the squared sum of all weights of the network:

$$\Omega = \frac{1}{2}\sum_{k} w_k^2, \tag{4}$$

where k is an index summing up all weights. This regularization method has been termed *weight decay*.

One weakness of the weight decay method is that it is not able to drive small irrelevant weights to zero, when gradient-based learning algorithms are employed, which may result in many small weights [25]. An alternative is to replace the squared sum of the weights with the sum of absolute value of the weights:

$$\Omega = \sum_{i} |w_i|. \tag{5}$$

It has been shown that this regularization term it is able to drive irrelevant weights to zero [22].

Both regularization terms in equations (4) and (5) have also been studied from the Bayesian learning point of view, which are known as the Gaussian regularizer and the Laplace regularizer, respectively.

A more direct measure for model complexity of neural networks is the number of weights contained in the neural network:

$$\Omega = \sum_{i}\sum_{j} c_{ij}, \tag{6}$$

where c_{ij} equals 1 if there is connection from neuron j to neuron i, and 0 if not. It should be noticed that the above complexity measure is not generally applicable to gradient-based learning methods.

A comparison of the three regularization terms using multi-objective evolutionary algorithms has been implemented [19]. Different to the conclusions reported [22] where gradient-based learning method has been used, it has been shown that regularization using the sum of squared weights is able to change (reduce) the structure of neural networks as efficiently as using the sum of absolute weights.

3.2 Multi-objective Optimization Approach to Regularization

It is quite straightforward to notice that neural network regularization in equation (2) can be reformulated as a bi-objective optimization problem:

$$\min \{f_1, f_2\} \tag{7}$$

$$f_1 = E, \tag{8}$$

$$f_2 = \Omega, \tag{9}$$

where E is defined in equation (3), and Ω is one of the regularization terms defined in equation (4), (5), or (6).

It is noticed that regularization is traditionally formulated as a single objective optimization problem as in Equation (2) rather than a multi-objective optimization problem as in equation (7). In our opinion, this tradition can be mainly attributed to the fact that traditional gradient-based learning algorithms are not able to solve multi-objective optimization problems.

3.3 Simultaneous Generation of Signal-type and Symbol-type Models

If evolutionary algorithms are used to solve the multi-objective optimization problem in Equation (7), multiple solutions with a spectrum of model complexity can be obtained in a single optimization run. Thus, if we choose a model of a higher complexity, the model will be of signal quality. On the contrary, if we choose a model of a lower complexity, the model will be of more symbol quality, assuming that a proper physical meaning can be associated with each neuron. In this sense, signal-type and symbol-type models can be generated simultaneously in one step using the multi-objective optimization approach to model selection.

4 Evolutionary Multi-objective Model Generation

4.1 Parameter and Structure Representation of the Network

A connection matrix and a weight matrix are employed to describe the structure and the weights of the neural networks. The connection matrix specifies the

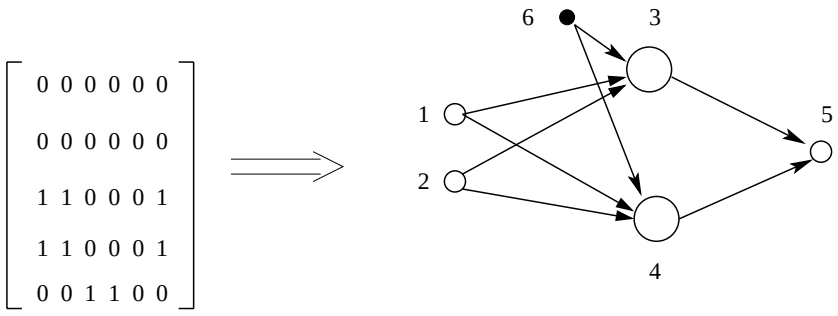

Fig. 1. A connection matrix and the corresponding network structure.

structure of the network, whereas the weight matrix determines the strength of each connection. Assume 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 matrix, if element $c_{ij}, i = 1, ..., M, j = 1, ..., M$ equals 1, it means that there is a connection between the i-th and j-th neuron and the signal flows from neuron j to neuron i. If $j = M + 1$, it indicates that there is a bias in the i-th neuron. Fig. 1 illustrates a connection matrix and the corresponding network structure. It can be seen from the figure that the network has two input neurons, two hidden neurons, and one output neuron. Besides, both hidden neurons have a bias.

The strength (weight) of the connections is defined in the weight matrix. Accordingly, if the c_{ij} in the connection matrix equals zero, the corresponding element in the weight matrix must be zero too.

4.2 Global and Local Search

A genetic algorithm has been used for optimizing the structure and weights of the neural networks. Binary coding has been used representing the neural network structure and real-valued coding for encoding the weights. Five genetic operations have been introduced in the global search, four of which mutate the connection matrix (neural network structure) and one of which mutates the weights. The four mutation operators are insertion of a hidden neuron, deletion of a hidden neuron, insertion of a connection and deletion of a connection [14]. A Gaussian-type mutation is applied to mutate the weight matrix. No crossover has been employed in this algorithm.

After mutation, an improved version of the Rprop algorithm [15] has been employed to train the weights. This can be seen as a kind of life-time learning (the first objective only) within a generation. After learning, the fitness of each individual with regard to the approximation error (f_1) is updated. In addition, the weights modified during the life-time learning are encoded back to the chromosome, which is known as the Lamarkian type of inheritance.

The Rprop learning algorithm [26] is believed to be a fast and robust learning algorithm. Let w_{ij} denotes the weight connecting neuron j and neuron i, then the change of the weight (Δw_{ij}) in each iteration is as follows:

$$\Delta w_{ij}^{(t)} = -\text{sign}\left(\frac{\partial E^{(t)}}{\partial w_{ij}}\right) \cdot \Delta_{ij}^{(t)}, \tag{10}$$

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 follows:

$$\Delta_{ij}^{(t)} = \begin{cases} \xi^+ \cdot \Delta_{ij}^{(t-1)} & , \text{ if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} \cdot \frac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ \xi^- \cdot \Delta_{ij}^{(t-1)} & , \text{ if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} \cdot \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ \Delta_{ij}^{(t-1)} & , \text{ otherwise} \end{cases}, \tag{11}$$

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

One exception must be considered. 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:

$$\Delta w^{(t)} = -\Delta_{ij}^{(t-1)}, \text{ if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} \cdot \frac{\partial E^{(t)}}{\partial w_{ij}} < 0. \tag{12}$$

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

In reference [15], it is argued that the condition for weight retraction in equation (12) 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 equation (12) is modified as follows:

$$\Delta w^{(t)} = -\Delta_{ij}^{(t-1)}, \text{ if } \frac{\partial E^{(t-1)}}{\partial w_{ij}} \cdot \frac{\partial E^{(t)}}{\partial w_{ij}} < 0 \textbf{ and } E^{(t)} > E^{(t-1)}. \tag{13}$$

It has been shown on several benchmark problems in [15] that the modified Rprop (termed as Rprop$^+$ in [15]) exhibits consistently better performance than the Rprop algorithm.

4.3 Elitist Non-dominated Sorting

In this algorithm, the elitist non-dominated sorting method proposed in the NSGA-II algorithm [9] has been adopted. Assume the population size is N. At first, the offspring and the parent populations are combined. Then, a non-domination rank and a local crowding distance are assigned to each individual in the combined population. In selection, all non-dominated individuals (say there are N_1 non-dominated solutions) in the combined population are passed to the

offspring population, and are removed from the combined population. Now the combined population has $2N - N_1$ individuals. If $N_1 < N$, the non-dominated solutions in the current combined population (say there are N_2 non-dominated solutions) will be passed to the offspring population. This procedure is repeated until the offspring population is filled. It could happen that the number of non-dominated solutions in the current combined population (N_i) is larger than the left slots ($N - N_1 - N_2 - ... - N_{i-1}$) in the current offspring population. In this case, the $N - N_1 - N_2 - ... - N_{i-1}$ individuals with the largest crowding distance from the N_i non-dominated individuals will be passed to the offspring generation.

5 An Illustrative Example

To illustrate the feasibility of the idea of generating signal-type and symbol-type models simultaneously using evolutionary optimization approach, neural networks have been generated to solve the breast cancer benchmark problem in the UCI repository of machine learning database collected by Dr. W.H. Wolberg at the University of Wisconsin-Madison Hospitals [23]. Studies have been carried out to extract symbolic rules from trained neural network using the three-step procedure for rule extraction on this benchmark problem [29, 27]. The benchmark problem contains 699 examples, each of which has 9 inputs and 2 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, $x_1, ..., x_9 \in \{0.1,\ 0.2,\ ..., 0.8,\ 0.9,\ 1.0\}$. The two outputs are 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 considered in this work. The data samples are divided into two groups: one training data set containing 599 samples and one test data set containing 100 samples. The test data are unavailable to the algorithm during the evolution.

The population size is 100 and the optimization is run for 200 generations. One of the five mutation operations is randomly selected and performed on each individual. The standard deviation of the Gaussian mutations applied on the weight matrix is set to 0.05. The weights of the network are initialized randomly in the interval of $[-0.2, 0.2]$. In the Rprop$^+$ algorithm, the step-sizes are initialized to 0.0125 and bounded between $[0, 50]$ during the adaptation, and $\xi^- = 0.2$, $\xi^+ = 1.2$, which are the default values recommended in [15] and 50 iterations are implemented in each local search.

Although a non-layered neural network can be generated using the coding scheme described in Section 3, a feedforward network with one hidden layer will be generated. The maximum number of hidden nodes is set to 10. The hidden neurons are nonlinear and the output layers are linear. The activation function

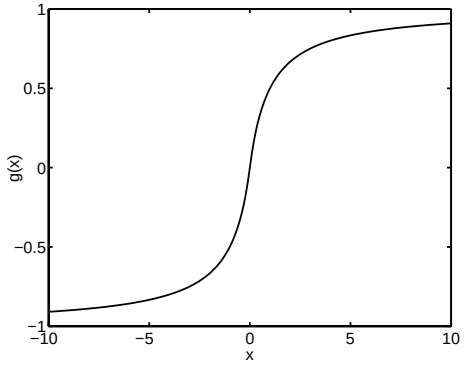

Fig. 2. The activation function of the hidden nodes.

used for the hidden neurons is as follows,

$$g(z) = \frac{x}{1 + |x|}, \tag{14}$$

which is illustrated in Fig. 2.

In this study, the complexity measure defined in Equation (6) has been used as the objective describing the complexity of the neural networks. The non-dominated solutions obtained at the 200-th generation are plotted in Fig. 3. Note that many solutions in the final population are the same and finally 41 non-dominated solutions have been generated.

Among the 41 neural networks, the simplest one has only 4 connections: 1 input node, one hidden node and 2 biases, see Fig 4(a). The mean squared error (MSE) of the network on training and test data are 0.0546 and 0.0324, respectively.

Assuming that a case can be decided to be "malignant" if $y < 0.25$, and "benign" if $y > 0.75$, We can then derive that if $x_2 > 0.4$, which means that $x_2 \geq 0.5$, then "malignant" and "benign" if $x_2 < 0.22$, i.e., then $x_2 \leq 0.2$.

From such a simple network, the following two symbol-type rules can be extracted (denoted as MOO_NN1):

$$\text{R1: If } x_2 \text{ (uniformity) } \geq 0.5, \text{ then malignant;} \tag{15}$$

$$\text{R2: If } x_2 \text{ (uniformity) } \leq 0.2, \text{ then benign.} \tag{16}$$

Based on these two simple rules, only 2 out of 100 test samples will be misclassified, and 4 of them cannot be decided with a predicted value of 0.49, which is very ambiguous. The prediction results on the test data are presented in Fig 4(b).

Now let us look at the second simplest network, which has 6 connections in total. The connection and weights of the network are given in Fig. 5(a), and the prediction results are provided in Fig. 5(b). The MSE of the network on training and test data are 0.0312 and 0.0203, respectively.

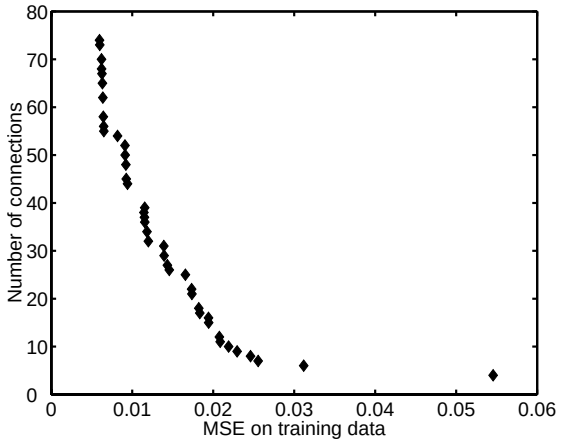

Fig. 3. The Pareto front containing 41 non-dominated solutions representing neural networks of a different model complexity.

In this network, x_2, x_4 and x_6 are present. If the same assumptions are used in deciding whether a case is benign or malignant, then we could extract the following rules: (denoted as MOO_NN2)

$$\begin{aligned}
\text{R1: If} \quad & x_2 \text{ (uniformity)} \geq 0.6 && \text{or} \\
& x_6 \text{ (bare nuclei)} \geq 0.9 && \text{or} \\
& x_2 \text{ (uniformity)} \geq 0.5 \wedge x_6 \text{ (bare nuclei)} \geq 0.2 && \text{or} \\
& x_2 \text{ (uniformity)} \geq 0.4 \wedge x_6 \text{ (bare nuclei)} \geq 0.4 && \text{or} \\
& x_2 \text{ (uniformity)} \geq 0.3 \wedge x_6 \text{ (bare nuclei)} \geq 0.5 && \text{or} \\
& x_2 \text{ (uniformity)} \geq 0.2 \wedge x_6 \text{ (bare nuclei)} \geq 0.7, \text{ then malignant;}
\end{aligned} \tag{17}$$

$$\begin{aligned}
\text{R2: If} \quad & x_2 \text{ (uniformity)} \leq 0.1 \wedge x_6 \text{ (bare nuclei)} \leq 0.4 && \text{or} \\
& x_2 \text{ (uniformity)} \leq 0.2 \wedge x_6 \text{ (bare nuclei)} \leq 0.2, \text{ then benign;}
\end{aligned} \tag{18}$$

Compared to the simplest network, with the introduction of two additional features x_6 and x_4 (although the influence of x_4 is too small to be reflected in the rules), the number of cases that are misclassified has been reduced to 1, whereas the number of cases on which no decision can be made remains to be 4, although the ambiguity of the decision for the four cases did decrease.

The above two neural networks are very simple in structure. We have shown that for such networks of a low model complexity, rules of symbolic quality can be extracted. In the following, we will take a look at two neural networks obtained in the multi-objective optimization, which are of better accuracy but are of more signal-type quality.

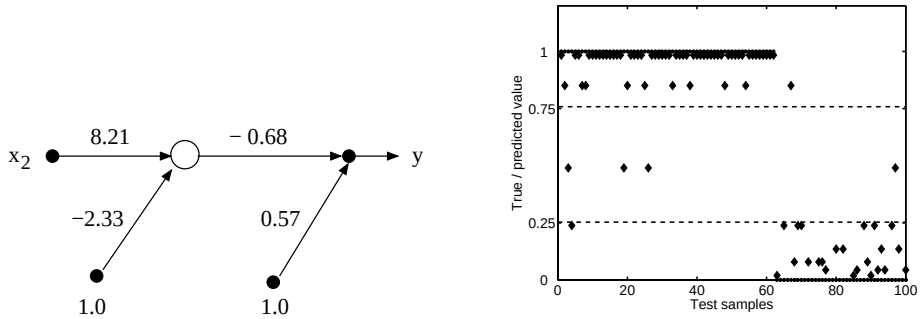

Fig. 4. The simplest neural network. a) The structure; and (b) The prediction results on test data.

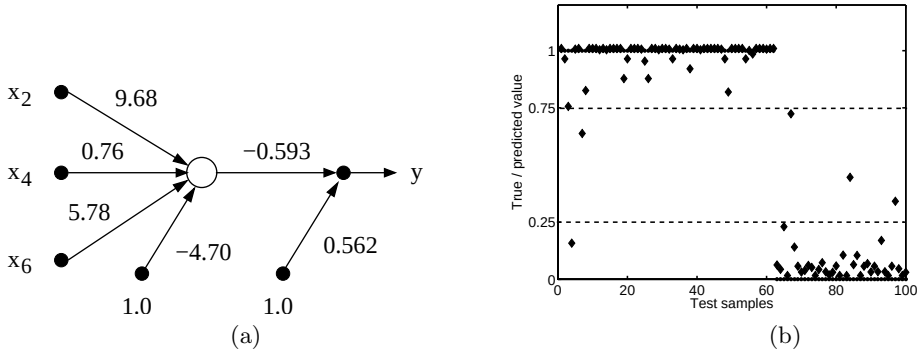

Fig. 5. The next simplest neural network. a) The structure; and (b) The prediction results on test data.

The first network of relatively higher model complexity has 16 connections, whose structure and weights are described in Fig. 6(a). The prediction results are plotted in Fig. 6(b). In this network, only x_3 is absent and there are 2 hidden nodes. The MSE on training and test data sets are 0.019 and 0.014, respectively. From Fig. 6(b), we can see that the classification accuracy is better: only two cases are mis-classified. However, extracting symbolic rules from the network becomes much more difficult. Besides, although the architecture of the two simple networks still exist in the current network, it not longer shows a dominating influence. Thus, the "skeleton" defined by the simple networks has been lost.

The most complex network obtained in the run has 74 connections. All input features are included in the network and the number of hidden nodes is 9. The MSE on the training data set is 0.0060, however, the MSE on the test data set increases to 0.066 with 5 samples misclassified and 1 undetermined. It seems that the network has over-fitted the training data and the understanding of the

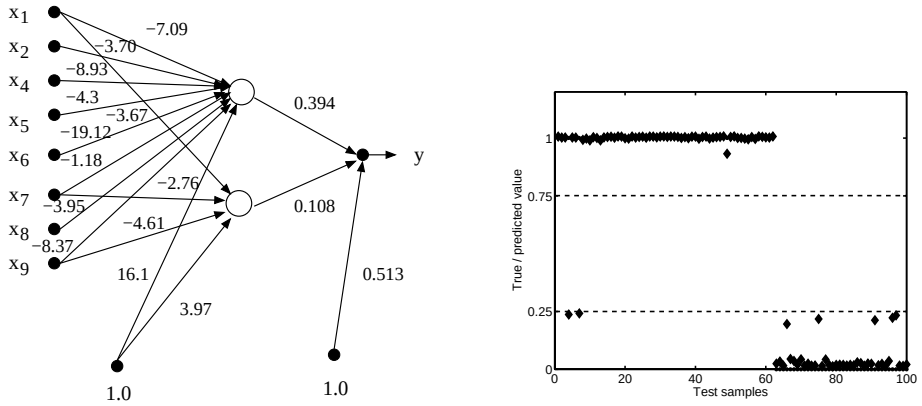

Fig. 6. The neural network with 16 connections. a) The structure; and (b) The prediction results on test data.

network is difficult. The prediction results of this neural network are provided in Fig. 7.

Table 1. Comparison of Performance and Complexity.

	No. Rules	Av. No. of Premises	Correct Classification Rate(%)
MOO_NN1	2	1	94
MOO_NN2	8	1.75	95
C4.5[25]	6	1.5	94.7
Ref.[23], 1e	1	4	97.2
Ref.[23], 2b	6	1.7	98.3
Ref.[25]	5	1.8	96.2

Table 1 lists the number of rules, the average number of premises in each rule and the correct classification rate on the test data of the rules obtained in this paper (MOO_NN1 and MOO_NN2), and those reported in [29] and [27]. It should be pointed out that the comparison is quite rough because the test data used in the three cases are different. Besides, samples with missing attribute values have been discarded in [29] and [27]. In our simulation, the missing values are simply replaced with the mean of the non-missing values of this attribute. Finally, in our work, samples with a lower confidence (those undecidable) are not counted as "correct classification". The purpose of the comparison is to show that the performance of the symbol-type rules generated in our work is comparable to that of rules extracted using popular machine learning (e.g., C4.5 [24]) and rule extraction methods.

Finally, we take a look at the relationship between the accuracy on test data and the complexity of the neural networks, which is shown in Fig. 8. It seems

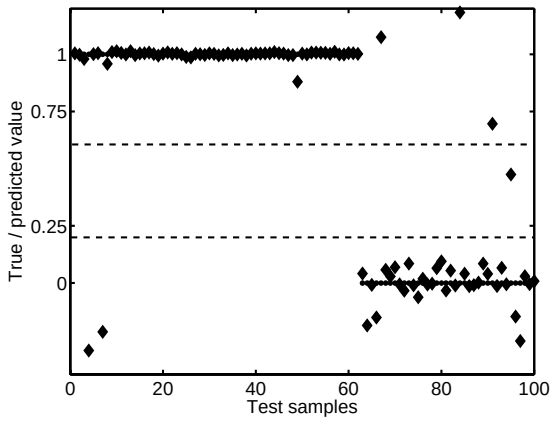

Fig. 7. The prediction results of the most complex neural network obtained in the simulation.

that for this problem, most neural networks having a higher complexity perform poorly on the test data. The reason behind this phenomenon is still unclear.

6 Conclusions

Both signal-type and symbol-type representations are important for cognitive modeling, as well as for critical applications. This paper suggests a method for generating multiple models of both signal-type and symbol-type simultaneously using an evolutionary multi-objective optimization algorithm instead of extracting symbolic rules after a neural network has been trained, like in most existing rule extraction approaches. This idea has been embodied in generating neural network models for the cancer diagnosis benchmark problem.

A problem in the current method is that the "skeleton" of the most complex neural network is not fully similar to the most simple, symbol-type network. To achieve this property, some additional constraints need to be introduced in the multi-objective optimization, which should be further investigated.

References

1. H.A. Abbass. An evolutionary artificial neural networks approach for breast cancer diagnosis. *Artificial Intelligence in Medicine*, 25(3):265–281, 2002.
2. H.A. Abbass. Speeding up back-propagation using multiobjective evolutionary algorithms. *Neural Computation*, 15(11):2705–2726, 2003.
3. R. Andrews, J. Diederich, and A. Tickle. A survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge Based Systems*, 8(6):373–389, 1995.
4. D. Badre and A. Wagner. Semantic retrieval, mnemonic control, and prefrontal cortex. *Behavior and Cognitive Neuroscience Reviews*, 1(3):206–218, 2002.

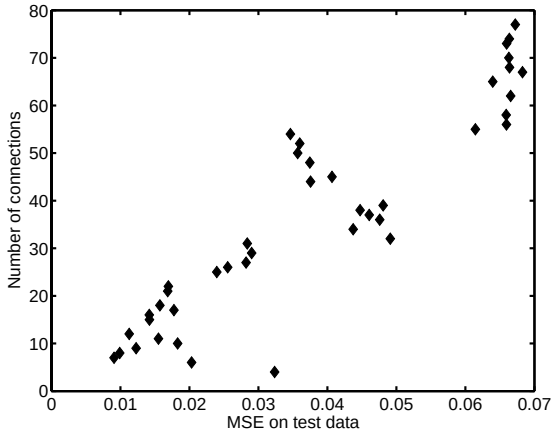

Fig. 8. The relationship between the accuracy on test data and complexity of the Pareto-optimal solutions.

5. C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK, 1995.
6. K.P. Burnham and D.R. Anderson. *Model Selection and Multimodel Inference*. Springer, New York, second edition, 2002.
7. C. Coello Coello, D. Veldhuizen, and G. Lamont. *Evolutionary algorithms for solving multi-objective problems*. Kluwer Academic, New York, 2002.
8. K. Deb. *Multi-objective Optimization Using Evolutionary Algorithms*. Wiley, Chichester, 2001.
9. K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Parallel Problem Solving from Nature*, volume VI, pages 849–858, 2000.
10. W. Duch, R. Adamczak, and K. Grabczewski. Extraction of logical rules from backpropagation networks. *Neural Processing Letters*, 7:1–9, 1998.
11. W. Duch, R. Setiono, and J. Zurada. Computational intelligence methods for rule-based data understanding. *Proceedings of the IEEE*, 92(5):771–805, 2004.
12. J.A. Fodor and Z.W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(3):3–71, 1988.
13. J. Gabrieli, R. Poldrack, and J. Desmond. The role of left prefrontal cortex in langrange and memory. *Proceedings of the National Academy of Sciences*, 95:906–913, 1998.
14. M. Hüsken, J. E. Gayko, and B. Sendhoff. Optimization for problem classes – Neural networks that learn to learn. In Xin Yao and David B. Fogel, editors, *IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks (ECNN 2000)*, pages 98–109. IEEE Press, 2000.
15. C. Igel and M. Hüsken. Improving the Rprop learning algorithm. In *Proceedings of the 2nd ICSC International Symposium on Neural Computation*, pages 115–121, 2000.
16. H. Ishibuchi, T. Yamamoto. Evolutionary multiobjective optimization for generating an ensemble of fuzzy rule-based classifiers. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1077–188, 2003.

17. M. Ishikawa. Rule extraction by successive regularization. *Neural Networks*, 13:1171–1183, 2000.
18. Y. Jin. *Advanced Fuzzy Systems Design and Applications*. Springer, Heidelberg, 2003.
19. Y. Jin, T. Okabe, and B. Sendhoff. Neural network regularization and ensembling using multi-objective evolutionary algorithms. In *Congress on Evolutionary Computation*, pages 1–8. IEEE, 2004.
20. Y. Jin and B. Sendhoff. Extracting interpretable fuzzy rules from RBF networks. *Neural Processing Letters*, 17(2):149–164, 2003.
21. A. Martin and L. Chao. Semantic memory and the brain: Structure and process. *Current Opinions in Neurobiology*, 11:194–201, 2001.
22. D.A. Miller and J.M. Zurada. A dynamical system perspective of structural learning with forgetting. *IEEE Transactions on Neural Networks*, 9(3):508–515, 1998.
23. L. Prechelt. PROBEN1 - a set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe, 1994.
24. J.R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, 1992.
25. R.D. Reed and R.J. Marks II. *Neural Smithing*. The MIT Press, 1999.
26. M. Riedmiller and H. Braun. A direct adaptive method for faster backpropgation learning: The RPROP algorithm. In *IEEE International Conference on Neural Networks*, volume 1, pages 586–591, New York, 1993. IEEE.
27. R. Setiono. Generating concise and accurate classification rules for breast cancer disgnosis. *Artificial Intelligence in Medicine*, 18:205–219, 2000.
28. R. Setiono and H. Liu. Symbolic representation of neural networks. *IEEE Computer*, 29(3):71–77, 1996.
29. I. Taha and J. Ghosh. Symbolic interpretation of artificial neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 11(3):448–463, 1999.
30. R. de A. Teixeira, A.P. Braga, R. H.C. Takahashi, and R. R. Saldanha. Improving generalization of MLPs with multi-objective optimization. *Neurocomputing*, 35:189–194, 2000.