International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:8, 2008

Novel Approach for Promoting the Generalization Ability of Neural Networks

Naiqin Feng, Fang Wang, and Yuhui Qiu

***Abstract*—**A new approach to promote the generalization ability of neural networks is presented. It is based on the point of view of fuzzy theory. This approach is implemented through shrinking or magnifying the input vector, thereby reducing the difference between training set and testing set. It is called "shrinking-magnifying approach" (SMA). At the same time, a new algorithm; α -algorithm is presented to find out the appropriate shrinking-magnifying-factor (SMF) α and obtain better generalization ability of neural networks. Quite a few simulation experiments serve to study the effect of SMA and α -algorithm. The experiment results are discussed in detail, and the function principle of SMA is analyzed in theory. The results of experiments and analyses show that the new approach is not only simpler and easier, but also is very effective to many neural networks and many classification problems. In our experiments, the proportions promoting the generalization ability of neural networks have even reached 90%.

***Keywords*—**Fuzzy theory, generalization, misclassification rate, neural network.

I. Introduction

THE generalization ability of neural networks (NNs) is an important performance criterion of NNs [1]. Researchers of this domain have been making an effort to promote the generalization ability of NNs. For solving this problem, people had presented several methods, for example, early stopping [2], regularization [3,4], result-feedback [5], fuzzification of input vector [6], neural network ensembles [7,8], etc. Although these methods can improve the generalization ability of NNs to some extent, but in general, the problem of NNs' generalization is still not solved or not completely solved. Because the essential character of artificial neural networks is of instance-based learning, it is impossible that NNs can solve all the problems by learning from limited examples. Therefore, developing some new methods for improving NNs' generalization ability will be greatly needed in a long time hereafter.

In these methods for improving NNs' generalization ability, the research of H. Ishibuchi and M. Nii is very important, and breaking a new path. They fuzzed input vector to avoid overfitting of NNs, thereby improving the generalization ability of NNs [6]. Recently, a new algorithm [5] to improve the learning performance of neural network through results-feedback, called FBBP algorithm, presented by Yan Wu and Shoujue Wang, can improve NNs' generalization ability too. This FBBP-based algorithm is an inner-and-outer layer learning method in which weight value renewing plays the dominating role with the assistance of input renewing. It minimizes the error function of neural network through the dual functioning of weight value and input vector value tuning, where tuning of the input vector is similar to fuzz the input vector. This idea brings us new inspiration. People had previously devoted large amounts of time to tuning weights of NNs for improving the NNs' performance (including the generalization ability), but lacked new ideas. In that case, improving generalization ability of NNs through fuzzification of input vector can be called a new angle of thinking and solving the problem. However, we think that the way to fuzz input vector is not unique, and there may be another different way. Through new explorations, we find a approach to promote the generalization ability of neural networks. This approach appropriately shrinks or magnifies input vector, thereby makes the generalization ability of NNs improved. We called the approach "Shrinking-Magnifying Approach" (SMA). The α -algorithm presented by us described the basic process of SMA. Through α -algorithm, we can find the appropriate shrinking-magnifying factor (SMF) and obtain a new neural network having better generalization ability.

II. Definitions, Principle and Approach

In this section, firstly, we will define the basic concepts used in computing and analyzing. Secondly, we will represent the basic idea and principle of the approach presented in this paper. Finely, the SMA and α -algorithm will be given.

A. Definitions

Definition 1. Assume that E_1, $E_{\leqslant 1}$, and $E_{\geqslant 1}$ represent the misclassification rate as SMF $\alpha = 1$, $\alpha \leqslant 1$, and $\alpha \geqslant 1$, respectively, and Ra and Rr represent the absolute rate of correcting mistakes and the relative rate of correcting mistakes, respectively, then we define that

Manuscript received July 2, 2005. This research is supported by the Science Fund of Henan Province, China (0511012500) and key project of Information and Industry Department of Chongqing City, China (200311014).

Naiqin Feng was with the College of Computer & Information Technology, Henan Normal University, Xinxiang, CO 453007 China. He is now with the Faculty of Computer & information Science, South West-China Normal University, Chongqing, 400715 China. (e-mail: fengnq@swnu.edu.cn).

Fang Wang is now with the Faculty of Computer & information Science, South West-China Normal University, Chongqing, 400715 China (e-mail: teresa@ swnu.edu.cn).

Yuhui Qiu is now with the Faculty of Computer & information Science, South West-China Normal University, Chongqing, 400715 China (e-mail: yhqiu@ swnu.edu.cn).

$Ra=(E_1- E_{\leqslant 1})\times 100\%$, if $\alpha \leqslant 1$ (1)
$Ra=(E_1- E_{\geqslant 1})\times 100\%$, if $\alpha \geqslant 1$ (2)
$Rr=(E_1- E_{\leqslant 1})/ E_1\times 100\%$, if $\alpha \leqslant 1$ (3)
$Rr=(E_1- E_{\geqslant 1})/ E_1\times 100\%$, if $\alpha \geqslant 1$ (4)

Definition 2. For vector $\boldsymbol{p}$, mode m and shrinking magnifying factor α, the α-fuzzy-mode-shrinking operation FmodS of $\boldsymbol{p}$ is defined as:

$$F \bmod S(\alpha p)=\begin{cases} f(\alpha \boldsymbol{p}), & \alpha \leq 1 \\ g((\alpha - m)\boldsymbol{p}), & \alpha > 1 \end{cases} \quad (5)$$

where, $m \in N$ (natural number set), f and g are the real function, for example, It can be $f(\alpha p)=\alpha p$, $g((\alpha -m)p)=(\alpha -m)p$, etc.. In general, f and g are different.

B. Basic Idea and Principle

Let's see a simple phenomenon in life.

Screen Effect Open the computer, tuning the screen, the picture can be shrunk or magnified. With the tuning of shrinking or magnifying, the picture can be clear on the whole but fuzzy on the local (when shrinking). Sometimes, it can become clear on the local but fuzzy on the whole (when magnifying). When shrinking, the object might become a fuzzy dot, and when magnifying, it might become losing its original appearance because the picture overflows out of the screen. In brief, in the two conditions, the object is fuzzed on the local or on the whole, respectively.

Shrinking-Magnifying Principle Fuzz-Based In practice, the approach of magnifying or shrinking things is often adopted by people. The former magnified the detail of the thing, in order to observe and research the thing's parts more accurately. However, with the parts being more accurate, the thing is fuzzed relatively on the whole, so belonging to the fuzzed approach on the whole; and the latter, shrinking approach, through shrinking the object and ignoring its detail, summaries the object in order to grasp the whole. However, with the whole of the object being clearer, the parts of the object are fuzzed relatively; thereby it belongs to the fuzzed approach on the local. So the two approaches, magnifying and shrinking, seem to like a contradiction, but in fact they have the inner consistency, and can be viewed as two different fuzz approaches. Along with that two objects are fuzzed on the part or on the whole, their differences must be reduced on the local or on the whole, making two objects that are very different more approach, even being merged into the same kind.

For example, assume that training example $\boldsymbol{p}=[x_1,x_2,x_3,x_4]$ $=[0.3,0.8,0.2,0.5]$, test example $\boldsymbol{p}_1=[x_1,x_2,x_3,x_4]$ $=[0.6,0.3,0.7,0.2]$, taking the stand on fuzzy theory [9~10], we can view $\boldsymbol{p}$ and $\boldsymbol{p}_1$ as two fuzzy sets. If using the Hamming distance formula,

$$d(p, p_1)=\frac{1}{n}\times\sum_{i=1}^{n}|\mu_{p(xi)}-\mu_{p1(xi)}| \quad (6)$$

and calculating semantic distance between $\boldsymbol{p}$ and $\boldsymbol{p}_1$, then the semantic distance under normal condition and the semantic distance when adopting SMA(α=0.02) respectively are:

$$d(p, p_1)=\frac{1}{4}\times\sum_{i=1}^{4}|\mu_{p(xi)}-\mu_{p1(xi)}|$$
$$=\frac{1}{4}(0.3+0.5+0.5+0.3)=0.4$$
$$d(\alpha.p,\alpha.p_1)=\frac{\alpha}{4}\times\sum_{i=1}^{4}|\mu_{p(xi)}-\mu_{p1(xi)}|$$
$$=\frac{1}{4}\times 0.02\times(0.3+0.5+0.5+0.3)=0.008$$

Obviously, the fuzzy functioning of SMA makes the semantic distance between the training example and the testing example decrease many times, thereby there may be the possibility of correctly classifying to $\boldsymbol{p}_1$.

To magnifying method (α>1), through FmodS operation, the semantic distance between $\boldsymbol{p}$ and $\boldsymbol{p}_1$ can also be decreased, so it provides the possibility to improve the generalization ability of neural network too.

C. Shrinking-Magnifying Approach

Based on the point of view of fuzzy theory given above, Shrinking-Magnifying Approach (SMA) is presented. This approach shrinks or magnifies input vector of a NN appropriately in advanced, thereby reducing the differences between training set and testing set, and improving the generalization ability of the NN. The appropriate shrinking-magnifying factor (SMF) α is determined by α-algorithm.

D. α-algorithm

Step 1. Divide training set P_1 and testing P_2, corresponding target set t_1 and t_2. Determine the search range [a, b] and search step δ; j=0;
Step 2. for r=a: δ :b
Step 3. j=j+1; α [j]=r;
Step 4. $p_1=\alpha[j]\cdot P_1$; $p_2=\alpha[j]\cdot P_2$;
Step 5. Train or generate the new net by using p_1, t_1;
Step 6. Simulate and test the net by using p_2, deriving the result y_2;
Step 7. Compare y_2 with t_2, calculate error $E[j]=t_2-y_2$;
Step 8. endfor;
Step 9. Find out the error $E_{\alpha=1}$ while α [j]=1;
Step10. if $\min(E[j])<E_{\alpha=1}$ then find out corresponding α [j], $\alpha_0=\alpha$ [j];
else $\alpha_0=1$;endif;
Step11. $p_0=\alpha_0\cdot P_1$;
Step12. Train or generate the target neural network NET by p_0;
Step13. Return

III. Experiments

A. Wine Classification using PNN

Firstly, we select Probabilistic Neural Network (PNN) to make the experiment to wine classification. PNN is simple and easy, because the scale of PNN is determined depending on the scale of input vector, but not needed to determine the number of neurons by people [1]. On the other hand, PNN is appropriate

for classification problems. Assume that search range of α -algorithm is [0.0001, 2], step δ =0.0001. Wine data set have 178 examples, and every one includes 13 attributes of wine. The 178 examples are divided into three categories, where the first category includes 59 examples; the second category and the third category include 71 examples and 48 examples, respectively. In the experiment, wine dataset is divided into two subsets. In the first subset, the example numbers of each of three categories are 30, 36 and 24, respectively; in the second subset, the example numbers of each of three categories are 29, 35 and 24, respectively. The two subsets are used to train and test respectively. The main results are listed in Table I.

From Table I, we can obtain results as follows:

(1) Shrinking method can reduce the Misclassification Rate (MR) of PNN to wine classification, i.e. SMA is able to reduce or correct misclassification of PNN. The appropriate SMF α is about 0.05~0.1;

(2) Magnifying can not reduce the MR of PNN to wine classification, on the contrary, the MR increases following increasing of α ;

TABLE I
RESULTS OF WINE CLASSIFICATION USING PNN

Shrinking Factor	Misclassification Rate	Shrinking Factor	Misclassification Rate
0.0001	53/88	**0.07**	**25/88**
0.0003	42/88	**0.09**	**25/88**
0.0005	30/88	**0.10**	**25/88**
0.0007	30/88	0.30	26/88
0.0009	29/88	0.50	29/88
0.001	28/88	0.70	34/88
0.003	25/88	0.90	42/88
0.005	27/88	**1.00**	**44/88**
0.007	25/88	1.10	46/88
0.009	28/88	1.30	53/88
0.01	28/88	1.50	54/88
0.03	26/88	1.70	56/88
0.05	**25/88**	2.00	57/88

(3) By calculating from Table I and formula 1 and 3, the maximum of Ra and Rr respectively are:

$$\max(Ra)=(E_1- E_{0.05})\times 100\%=19/88\times 100\%=22\%$$

$$\max(Rr)=(E_1- E_{0.05})/ E_1\times 100\%=19/44\times 100\%=43\%$$

From the above we can see that, when SMA is applied to PNN to wine classification, Ra and Rr of SMA are good.

B. Iris Classification using RBFN

Iris database is a known data system, widely used to pattern classification problems. Fisher's Iris database includes 150 records or patterns, where every one includes four attributes of Iris: Sepal Length, Sepal Width, Petal Length, and Petal Width, and can be expressed as a vector of four dimensions. The 150 records are divided into three species categories: Iris Setosa, Iris Versicolor, and Iris Virginica, where each of three categories includes 50 records. In our experiment, Iris data set is divided into two parts P1 and P2, each of them includes 75 examples, respectively, where P1 used to establish BPNN, and P2 used to simulate or test. The Radial Basis Function Network (RBFN) is established by function "newrbe" included in MATLAB neural network toolbox. The experiment results are given in Figure 1.

From Figure 1, we can obtain results as follows:

(1) Shrinking method can reduce the Misclassification Rate (MR) of RBFN to Iris classification, i.e. SMA is able to reduce or correct misclassification of RBFN. The appropriate SMF α is about 0.2;

(2) Magnifying can also reduce the MR of RBFN to Iris classification, but the reducing is less than shrinking method. The appropriate SMF α is about 1.2~1.6,

(3) From Figure 1, the maximum of Ra and Rr respectively are:

$$\max(Ra)=(E_1- E_{0.2})\times 100\%>50\%$$

$$\max(Rr)=(E_1- E_{0.2})/ E_1\times 100\%>90\%$$

From the above, we can see that the Ra and Rr of SMA are very good when SMA is applied to RBFN to Iris classification. This is encouraging because of obviously improving the generalization ability of RBFN.

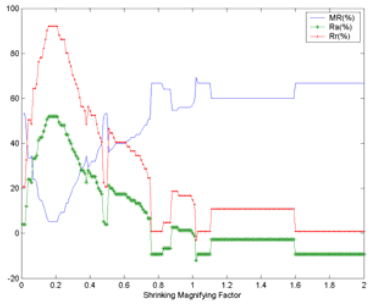

Fig. 1 Results of iris classification using RBFN

C. Iris Classification using BPNN

The application of neural networks using back propagation algorithm (BPNN) is very much wide at present, so the effect of SMA to BPNN should be studied. In this experiment, the architecture of BPNN is 4-6-3, and its transfer functions are "tansig" and "purelin", respectively. The learning algorithm to BPNN adopts the sized conjugate gradient back propagation algorithm. The BPNN is established by function "newff" included in MATLAB neural network toolbox. The Iris database used in this experiment is the same as the Iris database used in the above experiment. The main results are given in Table II.

The experiment results of BPNN are very interesting, and worth research.

(1) The results are concerned with initialization of BPNN. The experiment results show that SMA is not effective to BPNN's generalization ability to Iris classification problem, when all values of weights and biases of BPNN are initialized as zero through function "initzero". Whatever what value SMF α takes, the MR is always 18/75 (BPNN is established by function "newcf"). However, when the NN is initialized by Nguyen-Widrow method (by adopting this method, the activated area of every neuron will be well-distributed in whole input space, thereby avoiding wastage of neurons), SMA is

effective. At this time, the maximum of values of Ra and Rr respectively are:

max(Ra)=(E_1- $E_{0.28}$)×100%=9.55/75×100%=13%

max(Rr)=(E_1- $E_{0.28}$)/ E_1×100%=9.55/10.85×100%=88%

(2) Random initialization makes the work results of BPNN correspondingly change. Table 2 lists the mean of MR value for the 20 runs.

(3) Magnifying approach is effective to BPNN, i.e. it can improve the BPNN's generalization ability. To Iris classification problem, the maximum of Ra and Rr respectively are:

max(Ra)=(E_1- $E_{1.08.}$)×100%=7.7/75×100%=10%

max(Rr)=(E_1- $E_{1.08}$)/ E_1×100%

=7.7/10.85×100%=71%

It is clear that magnifying approach can also increase the generalization ability of BPNN. Of cause, shrinking is more effective than magnifying. The above experiment results may be not the optimal, but the conclusions are still effective.

TABLE II
RESULTS OF IRIS CLASSIFICATION USING BPNN

Shrinking Factor	Misclassification Rate	Shrinking Factor	Misclassification Rate
0.24	1.70/75	0.68	3.25/75
0.28	**1.30/75**	0.70	4.90/75
0.30	3.45/75	0.74	5.45/75
0.34	1.60/75	0.78	9.30/75
0.38	4.50/75	0.80	3.45/75
0.40	3.30/75	0.84	1.55/75
0.44	1.90/75	0.88	2.75/75
0.48	2.85/75	0.90	1.35/75
0.50	2.00/75	0.94	5.30/75
0.54	8.20/75	0.98	3.00/75
0.58	3.30/75	**1.00**	**10.85/75**
0.60	2.10/75	1.04	10.45/75
0.64	5.30/75	**1.08**	**3.15/75**

In addition, we have made the experiment to Wine classification using Radial Basis Function Network (RBFN), the experiment to Iris classification using PNN and the experiment to wine classification using Learning Vector Quantization Network (LVQN), with expectant effects. The experiment results of the former are as follows:

(1) Appropriate SMF α is about 0.02;

(2) In the experiment range, magnifying is not effective;

(3) At this time, the maximum of Ra and Rr respectively are:

max(Ra)=(E_1- $E_{0.02}$)×100%=39/88×100%=%44

max(Rr)=(E_1- $E_{0.02}$)/ E_1×100%=39/59×100%=66%

The experiment results of wine classification using LVQN are weaker,

max(Ra)=(E_1- $E_{2.0}$)×100%=2/88×100%=2.3%

max(Rr)=(E_1- $E_{2.0}$)/ E_1×100%=2/27×100%=7.4%

However, we need to point out that it is not invalid, although here the functioning of SMA improving generalization ability of LVQN is less effective. In fact sometimes, improving 1% can be crucial. In addition, LVQN's results of learning and simulation are concerned with its initialization.

IV. DISCUSSIONS AND ANALYSES

Why can such simple SMA and α-algorithm improve the NN's generalization ability and be very much obvious sometimes? Analyzing the results of experiments and exploring its causes, we think that there are several aspects as follows:

(1) Neural network belongs to the soft computation or not classical computation, having uncertainty and fuzziness in essence. This uncertainty and fuzziness are the application foundation of SMA.

(2) Shrinking approach is similar to a shrinking glass, in which the differences between two objects are shrunk or fuzzed. To neural networks, it shrinks or fuzzes the difference between training set and testing set (or new patterns), thereby promoting the adaptability of data which is similar to training example, i.e. improving the generalization ability of NNs. For instance, in Iris database, there is such a data $\boldsymbol{p}_1$=[x_1,x_2,x_3,x_4]=[60,22,50,15], which is of the third species. But when using the created PNN to simulate and test the testing set $\boldsymbol{p}_1$, it is misclassified as the second species, being one of three misclassified examples. However, when using SMA, taking SMF α=0.04~0.08, and recreating PNN and testing to $\boldsymbol{p}_1$, the previous mistake disappears, correcting one of total of three mistakes. To understand its reasons, we can do such analysis: randomly take a example from training set, which is of the third species, for example, $\boldsymbol{p}$=[x_1,x_2,x_3,x_4]=[63,28,51,15]. Standing on fuzzy theory, we might as well can regard $\boldsymbol{p}$ and $\boldsymbol{p}_1$ as another kind of fuzzy set, however, their membership function's vale range is not in normal [0, 1], but in [0,100]. If using Hamming distance formula directly to calculate the semantic distance between $\boldsymbol{p}$ and $\boldsymbol{p}_1$, then it is

$$d(p, p_1)=\frac{1}{4}\times\sum_{i=1}^{4}|\mu_{p(xi)}-\mu_{p1(xi)}|=\frac{1}{4}(3+6+1+0)=2.5$$

While adopting SMA (α=0.05), it becomes

$$d(\alpha.p,\alpha.p_1)=\frac{\alpha}{4}\times\sum_{i=1}^{4}|\mu_{p(xi)}-\mu_{p1(xi)}|$$

$$=\frac{1}{4}\times 0.05\times(3+6+1+0)=0.125$$

Obviously, the fuzz functioning of SMA makes semantic distance between training data and test data reduce many times, PNN therefore correctly classifies $\boldsymbol{p}_1$ as the third species.

(3) The regularization method can improve generalization ability of NNs, at the same time here we can take advantage of the point to interpret the functioning of SMA.

Under the condition that size of training set is kept, the NN's generalization ability inseparable from NN's scale. If NN's scale is much less than the training set's size, then the chance of overfitting is very little. On the contrary, to the same training example set, if the NN is on a large scale, then there is much probability of overfitting. But to a specific problem, it is very difficult to determine the NN's scale. The regularization method develops a new style. It modifies the NN's performance function of training in order to improve NN's generalization ability. In general NN's performance function of training is the mean squared error function, i.e.

$$mse = \frac{1}{N}\sum_{i=1}^{N}(e_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(t_i - a_i)^2 \quad (7)$$

The regularization method modified the performance function of NN as follows:

$$msereg = \gamma.mse + (1-\gamma)msw \quad (8)$$

where, γ is the proportional coefficient, msw is mean of the total of NN squared weight, i.e.

$$msw = \frac{1}{n}\sum_{j=1}^{n} W_j^2 \quad (9)$$

Through adopting the new performance criterion function, and under the condition of guaranteeing the value of performance function as little as possible, we can make the NN having less weight, i.e. make the NN's effective value of weights is as little as possible. This is equivalent to automatically shrinking of NN's scale.

The point of view that SMA is advantageous to reduce NN's weights is obvious to PNN (also to RBFN). The PNN is actually a parallel implementation of a standard Bayesian classifier. It is a three-layer network that can perform pattern classification. In its standard form, the probabilistic network is not trained. The training vectors simply become the weight vectors in the first layer of the network. The advantage of the PNN is that it does not require training. Consequently as SMA shrinks the training vectors, the PNN's weights are shrunk in the wake of it, being equivalent to automatically shrinking the NN's scale. It is clear that, from the formula 8 and 9, this is advantageous to reduce the value of performance function, i.e. is advantageous to reduce the PNN's error and improve NN's generalization ability. Of cause, because of the limit of the thing, shrinking boundlessly is not possible, but with a appropriate SMF α .

(4) In general, shrinking and magnifying is contradictory and opposite, but on the specific condition, both of them are also the unity of opposites. To PNN and RBFN, shrinking can effectively improve their generalization ability, but magnifying can't do that. However, under the specific conditions of LVQN or BPNN, both shrinking and magnifying are simultaneously effective, i.e. SMA is simultaneously effective to α in both range $(-\infty, 1]$ and $[1, +\infty)$. According to authors' comprehension, magnifying is also a fuzz approach: it magnifies and stresses a object's details, but fuzzes the whole of the object, i.e. shrinks and fuzzes the difference in its entirety between training examples and new patterns, equally, it can improve the NN's generalization ability.

The functioning mechanism of SMA can be better revealed by the screen effect and the fuzzy mode shrinking operation in section 2. Still using $\boldsymbol{p}$ and $\boldsymbol{p}_1$ as examples, when α =1.08, we can get m=1, g(x)=x, then,

$$F \bmod S(\alpha p) = g((\alpha - m)\boldsymbol{p}) = 0.08\boldsymbol{p},$$

$$F \bmod S(\alpha p_1) = g((\alpha - m)\boldsymbol{p_1}) = 0.08\boldsymbol{p_1}$$

By the Hamming distance formula 6, we obtained

$$d(p, p_1) = \frac{1}{4}\times\sum_{i=1}^{4}|\mu_{p(xi)} - \mu_{p1(xi)}|$$

$$= \frac{1}{4}(3+6+1+0) = 2.5$$

$$d(\alpha.p, \alpha.p_1) = \frac{\alpha}{4}\times\sum_{i=1}^{4}|\mu_{p(xi)} - \mu_{p1(xi)}|$$

$$= \frac{1}{4}\times 0.08\times(3+6+1+0) = 0.2$$

Simultaneously, the semantic distance between $\boldsymbol{p}$ and $\boldsymbol{p}_1$ is reduced, preparing the external condition for correct classification of the NN. Of cause, this is merely an external cause, and the external cause takes effect by the internal cause. The real effect of the NN is determined by the common function of the internal and external cause.

V. Conclusions

Standing on fuzzy theory, this paper presented a new approach and a new algorithm for improving the NNs' generalization ability. They are implemented by shrinking or magnifying the example data, thereby shrink or fuzz the difference between training set and testing set. We make a lot of experiments in order to verify their effectiveness, at the same time, with discussing and analyzing to their principle. The results of experiments and analyses show that SMA and α -algorithm are not only simple and easy, but also obvious effect and having its theory base to some extent. They are applicable to improve the generalization ability of many neural networks. But of cause, NNs is a big family, although we make many experiments and some analyses, wholly enumerating is impossible. Therefore, more practices and researches are needed to promote preferably the theory worth and application worth of SMA and α -algorithm.

References

[1] Martin T Hagan, Howard B Demuth, Mark Beale. *Neural Network Design*. Beijing: China Machine Press, CITIC Publishing House, 2002, 8.

[2] W. S. Sarle. *Stopped training and other remedies for over fitting*, to appear in Proceedings of the 27th Symposium on the Interface, 1995.

[3] G. E. Hinton. *Connectionist learning procedures.* Artificial Intelligence, 1989, 40:185-234.

[4] A. S. Weigand, D. E. Rumelhart, and B. A. Huberman. *Generalization by weight elimination with application to forecasting*. In Advances in Neural Information Processing Systems 3, R. P. Lippman, J. E. Moody and D. J. Touretzky, eds, San Mateo, CA: Morgan Kaufmann, 1991, 575-582.

[5] Yan Wu, Shoujue Wang. *A New Algorithm to Improve the Learning Performance of Neural Network through Result-Feedback.* Journal of Computer Research and Development (in Chinese), 2004, 41(9), 488-492.

[6] H. Ishibuchi, M. Nii. *Fuzzification of input vector for improving the generalization ability of neural networks*. The Int'l Joint Conf. on Neural Networks, Anchorage, Alaska, 1998.

[7] L. K. Hansen, P. Salamon. *Neural Network Ensembles*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1990, 12 (10): 993-1001.

[8] D. Opitz, R. Maclin. *Popular Ensemble Methods: An Empirical Study*. Journal of Artificial Intelligence Research, 1999, 11: 169-198.

[9] Li-Xin Wang. *A Course in Fuzzy Systems and Control. Upper Saddle River*, NJ: Prentice-Hall Inc, A Pearson Education Company, 1997.

[10] J. S. R. Jang, C. T. Sun, E. Mizutani. *Neuro-Fuzzy and Soft Computing*. Upper Saddle River, NJ: Prentice-Hall Inc, Simon & Schuster/A Viacom Company, 1997.