Nonlinear Fitting by Using a Neural Net Algorithm

Zheng Li*

Laboratory of Computer Chemistry, Shanghai Institute of Organic Chemistry, Academia Sinica, Shanghai, People's Republic of China, 200032

Zhaonian Cheng and Li Xu

Shanghai Institute of Metallurgy, Shanghai, People's Republic of China, 200050

Tonghua Li

Department of Chemistry, Tongji University, Shanghai, People's Republic of China, 200092

A novel transfer function which is very suitable for normalized data set and a modified conjugate gradient algorithm which converges much faster are proposed to improve the performance of the neural network training procedure. The overfitting problem is discussed in detail. The optimal fitting model can be obtained by adjusting the number of hidden nodes. A data set of furnace lining durability was used as an example to demonstrate the method. The predictive results were better then that of principal component regression and partial least square regression.

INTRODUCTION

In the past few years, artificial neural network (ANN) has generated wide interests and is now gaining in popularity. These models are composed of many linear or nonlinear computational elements (nodes) operating in parallel and arranged in biological analogy patterns, and the nodes are connected by weights that are gradually adapted during training to improve the performance. Some successful applications have been reported in the field of speech and image recognition, signal processing, and other aspects of artificial intelligence research.[1] Recently, the ANN method has been used for spectroscopic calibration and quantitation[2] and structure–activity relationships.[3-5]

In this paper, the neural network algorithm used as fitting models is studied. (1) A novel transfer function is suggested for different kinds of purposes. (2) A conjugate gradient algorithm is described in detail to replace the back-propagation (BP) algorithm. (3) The overfitting problem is discussed and some tactics are proposed to get optimal fitting models.

THEORY

There are several networks such as the feed forward type, feedback type, Boltzmann machine type, self-organize type, and so on.[1] This paper deals only with feed forward type. This kind of network can be used as a linear or nonlinear multidimensional space projection. It is reported[1] that the feed forward network with no more than two hidden layers can arbitrarily form complex decision regions and fit for continuous functions. A lot of problems (e.g. pattern recognition, motion detection, process fitting, and so on) can be considered as such types of projection.

The network learning is an iterative calculation in which a set of inputs and corresponding expected output values are presented and the connection weights are gradually adjusted until the desired level of precision is achieved between the expected and the actual outputs. The weights can be determined by the back-propagation training algorithm,[1] but one of the shortcomings of this algorithm is the low convergence speed. Several modifications have been reported but scarcely used in the range of chemistry. In this work, the BP algorithm is modified by conjugate gradient principle[6] to get a much faster convergence speed and better fitting models. The calculation steps, which are quite suitable for chemistry problems, are explained in detail.

In order to make the calculation easier, it is convenient to normalize the data set before training. In this way the sample set is mean-centered and the variance for every variable is unity.

Transfer Functions. A lot of transfer functions (e.g. sin (x), x^2, tanh (x), sigmoid function, and so on) have been tested[2,3] by some authors, but it is rather difficult to distinguish which one is better, so it is necessary to look for some functions which are suitable for different kinds of data sets and different purposes.

Some calculation results demonstrated that convergence can be accelerated by using a data set with symmetrical structure. If the data set is normalized, all the samples dispersed from negative to positive with a center of zero have some degree of symmetry. In order to match the symmetry of the normalized data, a new transfer function ($f(x) = 2/(1 + e^{-x}) - 1$, as shown in Figure 1) is suggested. Comparing to the sigmoid function, this function expands the output range from 0 to +1 to −1 to +1 with a center of zero.

Comparison calculations showed that the performance can be improved by using such a function, especially for normalized data. Because the output of the network could not be confined within −1 to +1, it is convenient to use the linear function as the transfer function in the output layer. The following combinations of functions for hidden layers ($f_{hid}(x)$) and for output layer ($f_{out}(x)$) are suggested.

(1) For pattern recognition:

$$f_{hid}(x) = f_{out}(x) = 2/(1 + e^{-x}) - 1$$

* Corresponding author.

(1) Lippmann, R. P. *IEEE ASSP Mag.* 1987, April 4, 22.
(2) Long, J. R.; Gegoriou, V. G.; Gemperline, P. J. *Anal. Chem.* 1990, *62*, 1791-1797.
(3) Tomoo, Aoyama; Hiroshi, Ichikawa. *Chem. Pharm. Bull.* 1991, *39* (2), 358-366.
(4) Tomoo, Aoyama; Hiroshi, Ichikawa. *Chem. Pharm. Bull.* 1991, *39* (20), 372-378.
(5) Tomoo, Aoyama; Yuji, Suzuki; Hiroshi, Ichikawa. *J. Med. Chem.* 1990, *33* (9), 2583-2590.
(6) Flectcher, R.; Rceves, C. M. *Comput. J.* 1964, *7*, 149.

0003-2700/93/0365-0393$04.00/0

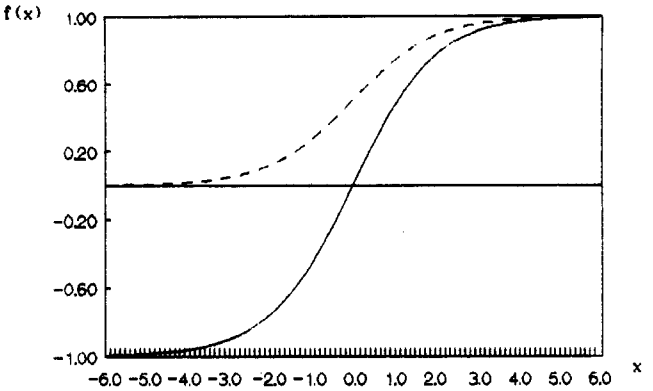

Figure 1. (- - -) Sigmoid function. (—) New transfer function.

(2) For nonlinear fitting:

$$f_{hid}(x) = 2/(1 + e^{-x}) - 1, f_{out}(x) = x$$

(3) For linear fitting (such a network is the same as the linear multiregression):

$$f_{hid}(x) = f_{out}(x) = x$$

CG Algorithm. *Step 1.* Set up the topological structure of the network. The input nodes correspond to the variables and the output nodes to the target. Multitarget problems can be decomposed to several monotarget ones. Then the hidden layers and nodes should be selected. Usually one hidden layer is enough.

Step 2. Calculate the number of total weights and offsets, N_G, e.g. for one hidden layer:

$$N_G = (N_{in}+1)N_{hid\ 1} + (N_{hid\ 1}+1)N_{out}$$

for two hidden layers:

$$N_G = (N_{in}+1)N_{hid\ 1} + (N_{hid\ 1}+1)N_{hid\ 2} + (N_{hid\ 2}+1)N_{out}$$

where N_{in} is the number of input nodes, N_{out} is the output nodes, $N_{hid,1}$ is the first hidden layer node, $N_{hid,2}$ is the second hidden layer node.

Step 3. Initialize weights and offsets. Set all weights and offsets to random values from −1 to +1.

Step 4. Present training data set. Present the input vector as $x_1, x_2, \ldots x_n$ and the corresponding desired outputs $d_1, d_2, \ldots d_m$ as training data set.

Step 5. Calculate actual output of network and first optimal direction $z_{ij}(1)$.

$$z_{ij}(1) = \delta_j(1)O_i(1)$$

To the output layer:

$$\delta_j(1) = d_j - O_j(1) \text{ (if } f_{out}(x) = x)$$

$$\delta_j(1) = 0.5(1-O_j^2(1))(d_j - O_j(1)) \text{ (if } f_{out}(x) = 2/(1 + e^{-x}) - 1)$$

To the hidden layers:

$$\delta_j(1) = \sum_k \delta_k(1)w_{jk}(1) \text{ (if } f_{hid}(x) = x)$$

$$\delta_j(1) = 0.5(1-O_j^2(1))\sum_k \delta_k(1)w_{j,k}(1) \text{ (if } f_{hid}(x) = 2/(1 + e^{-x}) - 1)$$

where δ_j is an error term for node j, O_j is the output of node j in this layer, O_i is the output of node i in the lwoer layer, δ_k is an error term for node k in the upper layer, and w_{ij} the weight from node i to node j.

Step 6. Adapt the weights $w_{ij}(t)$ and calculate the conjugate direction $z_{ij}(t)$

$$w_{ij}(t) = w_{ij}(t-1) + \alpha z_{ij}(t-1)$$

where α is a learning rate; usually 0.01 is enough for normalized data.

$$z_{ij}(t) = -G_{ij}(t) + \mu(t)z_{ij}(t-1)$$

$$-G_{ij}(t) = \delta_j(t)O_i(t)$$

where μ is conjugate coefficient, $\mu(t) = [G(t)]^T[G(t)]/[G(t-1)]^T[G(t-1)]$, and $G(t)^T = [G_{11}(t), G_{12}(t), \ldots G_{ij}(t)]$.

Step 7. Calculate the general error $\sum_p E_p(t)$:

$$\sum_p E_p(t) = \sum_p \sum_j (d_j - O_j(t))^2$$

Here p is the variable for the number of samples and O_j is the output of network. If $E_p(t)$ is not small enough, repeat the calculation by going back to step 6. Because the number of conjugate direction equals to the number of variables, so if the iterations t equal to N_G, repeat the calculation by going back to step 5 and let $z_{ij}(1) = z_{ij}(t+1)$. The offsets can be adapted in a similar way by assuming they are connection weights on links from auxiliary constant-valued inputs.

It is verified by comparison that the convergence speed of the CG algorithm is much faster than that of BP. Furthermore, the CG algorithm can get convergence result with fewer hidden nodes which the BP algorithm could not do within practical time. This is very important in nonlinear fitting.

EXPERIMENTAL SECTION

The value of furnace lining durability mainly depends on the chemical composition and technological conditions. A listing of data set of furnace lining durability in a certain steel plant appears in Table I. y is the durability data (x_1, quantity of adding lining material; x_2, blowing time; x_3, duration of heating; x_4, content of Mn; x_5, content of iron in slag; x_6, operation rate).

If the calibration model between y and x_1–x_6 can be set up, it is quite useful in adjusting technological conditions to prolong the furnace lining durability.

RESULTS AND DISCUSSION

Transfer Function and Algorithm. The data set in Table I was used to make a comparison among different algorithms. The structure of network was set up: Six input nodes were corresponded to the six technological parameters separately. One output node corresponded to the value of furnace lining durability. One hidden layer with five hidden nodes was included. The data set was normalized as the training sample set.

Some results representing the training procedure are listed in Table II. K is the iteration setp; E is the general error. Case a: BP algorithm with sigmoid function as the transfer function. Case b: BP algorithm with a new transfer function. Case c: CG algorithm with a new transfer function.

It is worth noting that the convergence is sped up significantly when the new transfer function is used. The BP algorithm is as good as the CG algorithm at early training steps but much worse than GC at late training steps.

Overfitting Problem. One of the merits of neural network is its strong nonlinear characteristic, but it will cause an unfavorable phenomenon—overfitting.[2] Most of the data contain some degree of noise, so a good fitting model should fit for the general tendency of the data set, not fit for every sample exactly. But some networks with too strong

Table I. Data Set of Furnace Lining Durabilities

no.	y	x_1	x_2	x_3	x_4	x_5	x_6
1	1030	0.2922	18.5	41.4	58	18	83.3
2	1006	0.2672	18.4	41	51	18	91.7
3	1000	0.2685	17.7	38.6	52	17.3	78.9
4	702	0.1835	18.9	41.8	18	12.8	47.2
5	1087	0.2348	18	39.4	51	17.4	57.4
6	900	0.1386	18.9	40.5	39	12.8	22.5
7	708	0.2083	18.3	39.8	64	17.1	52.6
8	1223	0.418	18.8	41	64	16.4	26.7
9	803	0.103	18.4	39.2	20	12.3	35
10	715	0.4893	19.3	41.4	49	19.1	31.3
11	784	0.2058	19	40	40	18.8	41.2
12	535	0.0925	17.9	38.7	50	14.3	66.7
13	949	0.1854	19	40.8	44	21	28.6
14	1012	0.1963	18.1	37.2	46	15.3	63
15	716	0.1008	18.2	37	46	16.8	33.9
16	858	0.2702	18.9	39.5	48	20.2	31.3
17	826	0.1465	19.1	38.6	45	17.8	28.1
18	1015	0.1353	19	38.6	42	16.7	39.7
19	861	0.2244	18.8	37.7	40	17.4	49
20	1098	0.2155	20.2	40.2	52	16.8	41.7
21	580	0.0316	20.9	41.2	48	17.4	52.6
22	573	0.0491	20.3	40.6	56	19.7	35
23	832	0.1487	19.4	39.5	42	18.3	33.3
24	1076	0.2445	18.2	36.6	41	15.2	37.9
25	1376	0.2222	18.4	37	40	13.7	42.9
26	914	0.1298	18.4	37.2	45	17.2	44.3
27	861	0.23	18.4	37.1	47	22.9	21.6
28	1105	0.2436	17.7	37.2	45	16.2	37.9
29	1013	0.2804	18.3	37.5	46	17.3	20.3
30	1249	0.197	17.3	35.9	46	13.8	57.4
31	1039	0.184	16.2	35.3	43	16.6	44.8
32	1502	0.1679	17.1	34.6	43	20.3	37.3
33	1128	0.1524	17.6	36	51	14.2	36.7

Table II. Comparison among Different Algorithms

K	100	200	300	400	500	600	700	800	900	1000
E^a	11.84	4.36	2.81	2.01	1.35	1.05	0.86	0.77	0.71	0.66
E^b	3.40	1.21	0.84	0.17	0.091	0.078	0.062	0.052	0.045	0.038
E^c	3.41	1.13	0.24	0.11	0.058	0.042	0.023	0.011	0.0087	0.0041

Table III. Simulated Data for the Function $y = x^a$

	training set		test set	
no.	y'	x'	y''	x''
1	−0.969	−0.926		
2			−0.777	−0.777
3	−0.572	−0.513		
4			−0.333	−0.333
5	−0.121	−0.102		
6			0.111	0.111
7	0.338	0.358		
8			0.555	0.555
9	0.764	0.812		
10	0.992	0.909		

[a] Function $y = x$: y' and x' are the samples with 10% noise as training samples, y'' and x'' are the noise free samples as prediction samples. The network with four hidden nodes was trained by using these data.

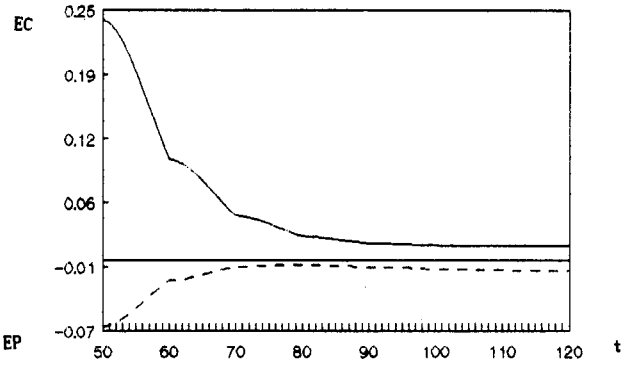

Figure 2. Calibration error (EC) and prediction error (EP) versus iteration.

nonlinear ability may fit for every sample exactly after too many interations, i.e. fit for noise. The more the noise the data contains, the worse the "overfitting" problem will be.

Presented in Table III is a set of simulated data showing the "overfitting".

As shown in Figure 2 the calibration error (EC) decreased with iterations monotonously. The prediction error (EP) decreased to its lowest value at $t = 80$ and then increased with t. It is obvious that the model fit for random noise after 80 iterations. The overfitting results in greater prediction error for the test data. The best model could be obtained by stopping iterations at $t = 80$. The value of EC when the EP arrives at its lowest value indicates the level of noise contained in the data set. The fitting precision should not be higher than the level of noise. The criterion similar to "cross-validation" can be obtained by decomposing data set $\{S\}$ to two parts: calibration set $\{C\}$ for training network, and prediction set $\{P\}$ for detecting overfitting. The iterations must be stopped at a certain step.

The Number of Hidden Nodes. For linear problems, PLS or PCR regression[7] is a safe method without taking the risk of overfitting. If the data set is so accurate that overfitting is not significant, the network may be a good substitute for

(7) Geladi, P.; Kowalski, B. R. *Anal. Chim. Acta* **1986**, *185*, 1–17.

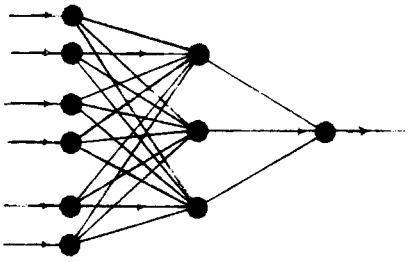

Figure 3. The structure of the network to fit the data set of Table I.

PLS or PCR. As for the nonlinear problem, the nonlinear characteristic of network mainly depends on the number of hidden layers and nodes. The larger the number of hidden nodes, the stronger the nonlinear characteristic of network will be.

The data set of furnace durability (Table I) was used as an example to demonstrate the effects of the hidden nodes. Twenty-nine samples were used as training samples; the other four samples chosen randomly (no. 1, 7, 17, 33) were left out as prediction samples.) Here are four cases when different hidden nodes are used to train this data set.

Case 1. One hidden node, $f_{hid}(x) = f_{out}(x) = x$ (such a network is the same as linear multiregression). The standard calibration error (SEC) decreased to 13.4 as the limit. The average SEC is 0.46; such a large error indicates the strong nonlinear characteristic of the data set.

Case 2. Two hidden nodes, $f_{hid}(x) = 2/(1 + e^{-x}) - 1$, $f_{out}(x) = x$. The SEC decreased to 7.12 (average SEC is 0.25) at first. Then no decreasing tendency was observed. So in this case, there are too few hidden nodes to fit the nonlinear characteristic of data set.

Case 3. Four hidden nodes; the transfer functions are the same as in case 2. The SEC decreased to 0.14 (average SEC is 0.0048) after 3000 iterations and showed the tendency to decrease to a lower level. In this case the nonlinear characteristic of network is superior to that of data set. The ANN's fitting is too precise after too many iterations, i.e. overfitting.

Case 4. Three hidden nodes; the functions are the same as in case 2. The SEC decreased to 1.24 after 4000 iterations and without any tendency to decrease to a lower level. For every sample, the average standard calibration error is about 0.04. The network fits for the general tendency of data set, not for every sample exactly. In this case the nonlinear characteristic of network just matches that of the data set, so it is quite reasonable to say that three hidden nodes make the optimal fitting models.

The noise level may have some relation with the number of the hidden nodes. Too few hidden nodes will cause the fitting precision to be lower than the noise level; too many will cause the fitting precision to be higher than the noise level, so it is quite reasonable to conclude that the overfitting may be controlled by adjusting the number of hidden nodes. In addition, fewer weights and offsets will make the network more reliable. So the optimal fitting model can be obtained by reducing the hidden nodes to the least number.

It should be pointed out that the CG algorithm is very useful in looking for the least number of hidden nodes. The convergence result could not be obtained with three hidden nodes for the above data set when using the BP algorithm. Meanwhile, it must be noticed that the local minimum should be avoided by initializing weights and offsets several times to get the "real" least number of hidden nodes.

Table IV. Calibration Results

no.	y	y_c	no.	y	y_c
2	1006	969.8	19	861	819.4
3	1000	1015.4	20	1098	967.6
4	702	680.9	21	580	978.4
5	1087	1070.7	22	573	576.2
6	900	903.1	23	832	961.5
8	1223	1196.6	24	1076	1109.6
9	803	803.3	25	1376	1333.7
10	715	716.6	26	914	921.2
11	784	928	27	861	866.4
12	535	530	28	1105	1225.2
13	949	957.2	29	1013	989.1
14	1012	1015.1	30	1249	1250.7
15	716	717.2	31	1039	1008
16	858	858.5	32	1502	1465
18	1015	946.6			

Table V. Prediction Results

no.	y	y_{ANN}	y_{PCR}	y_{PLS}
1	1030	969.3	1118.4	1114.6
7	708	602.7	795.8	919.2
17	826	945.7	901.4	875.6
33	1128	1244.1	828.9	964.9
S		103.16	137.15	142.24

The Optimal Model of the Furnace Lining Durability. As mentioned above, the network with three hidden nodes is the best fitting model for this data set (Figure 3). The calibration results are listed in Table IV where y is the experiment data and y_c the calibration results.

The prediction results are listed in Table V, where y is the experiment data, and y_{ANN}, y_{PCR}, Y_{PLS} are the prediction results of the ANN method, PCR regression, and PLS regression, respectively. S is the sum of squared error in the prediction set to describe the prediction ability of the model.

$$S = (\sum (y_{exp} - y_{cal})^2)^{1/2}$$

It is obvious that the ANN fitting model is somewhat superior to that of PCR and PLS for nonlinear fitting. Although such a method causes some unfavorable problems, ANN will be a promising way for nonlinear fitting.

CONCLUSIONS

It may be useful to note that an artificial neural network provides a unified form for nonlinear fitting. Such a method can be used as a "soft" model for different kinds of nonlinear problems. "Overfitting" can be detected by decomposing data set to calibration set and prediction set. Optimal fitting models can be obtained by reducing hidden nodes to the least number. The CG algorithm can be used to improve the performance of the network. The ANN method developed in this paper has some potential benefits in structure–activity study, process prediction, optimization, and so on.

ACKNOWLEDGMENT

The authors are indebted to Professor Siming Zhu at East China University of Chemical Technology for helpful discussions and suggestions concerning this paper.

RECEIVED December 2, 1991. Revised manuscript received October 15, 1992. Accepted October 30, 1992.