A Fast BP Learning Algorithm via A Hybrid Approach①

Liu Guangyuan Yu Juebang
(Dept. of Optoelectronic Technology, UEST of China Chengdu 610054)
Qiu Yuhui
(Southwest Normal University Chongqing 400715)

Abstract A hybrid approach based on changing the activation function and using a robust error estimator for improving the learning speed of BP network is suggested in this paper. The activation function used in this paper is a non-differentiable piecewise linear function and its derivative function is re-defined as a new continuous one. The network learning is proceeding under new error function. Significant improvement is observed in the simulations of the XOR and encoder/decode examples.

Key words neural network; BP algorithm; robust error estimator; XOR; encoder/decoder

Based on a continuous and differentiabe activation function and an LMS error estimator, a standard BP algorithm is used to reduce the learning speed. To enhance its learning speed, a series of modifications have been made: in Ref.[1], an adaptive learning rate is used; in Ref.[2], a nondifferentiable activation function is suggested; in Ref.[3], a new error estimator is proposed; in Ref.[4] and Ref.[5], robust error estimators are used. To improve the progress made in these literatures furtherly, we suggest a hybrid approach in this paper.

1 Cauchy Estimator

It is well-known that the main performance of a robust error estimator is the ability of suppressing the gross errors such as outliers in the learning process. The Cauchy estimator, one of the robust error estimators, is described by an error function[5]

$$\rho(r_i)=\frac{1}{2}\log(1+r_i^2) \tag{1}$$

where $r_i=t_i-y_i$ is the residual of pattern i with target t_i. Hence the influence function of the estimator is given by [4]

$$\psi(r_i)=\frac{\partial\rho(r_i)}{\partial r_i}=\frac{r_i}{1+r_i^2} \tag{2}$$

which is plotted as curve ① in Fig.1, in which the influence function of an LMS estimator is also plotted as curve ②.

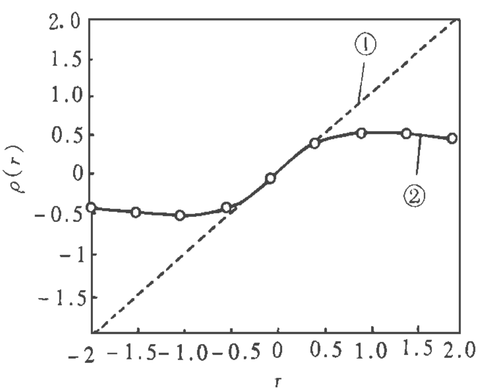

Fig.1 The influence function

It is obvious from Fig.1 that the LMS estimator is extremely sensitive to gross errors since its influence function increases proportionally with the residual, and therefore even a single large residual from an outlier would outweight the remaining small residuals thus prevent the algorithm from converging to its target function. On the other hand, the impact of gross errors including outliers on the Cauchy estimator will be limited since even for very large residuals, the outliers have no effects at all due to

① Received June 11, 1997 and revised December 14, 1997

$$\lim_{r_i \to \infty} \psi(r_i) = \lim_{r_i \to \infty} \frac{r_i}{1 + r_i^2} = 0 \tag{3}$$

The introduce of Cauchy error estimator is proved to be helpful for improving the convergence performance of our modified BP algorithm.

2 Alternatively Employing Different Activation Functions

The introducing of non-differentiable activation functions into the BP algorithm has been proved to be very efficient in enhancing its learning speed. However, in serveral benchmark-liked problems, such as XOR, encoder/decoder problem, oscilliations are occurred in most of the iteration process[2]. In our algorithm design, continuous differentiable activation function and non-differentiable activation funcation are alternatively employed for hidden layer and output layer to overcome the oscillation pheomena.

3 The Modified BP Algorithm Design

A p-q-n neuron three layered (L_A, L_B, L_C) MLP is used to implement our algorithm. The vector pattern for the three layers is denoted as A, B, C respectively, where $A=\{a_1, a_2, \cdots, a_p\}$, $B=\{b_1, b_2, \cdots, b_q\}$, $C=\{c_1, c_2, \cdots, c_n\}$. The connection weights between $L_A - L_B$, $L_B - L_C$ are denoted as W_{ih}, W_{hj} respectively, where $i=1, 2, \cdots, p$; $h=1, 2, \cdots, q$; $j=1, 2, \cdots, n$.

The activation function of neruon j in the output layer L_C is assumed to be a non-differentiable function

$$f_j(x) = \begin{cases} 1 & x > \frac{1}{2} \\ x + \frac{1}{2} & -\frac{1}{2} \leqslant x \leqslant \frac{1}{2} \\ 0 \quad 1 & x < -\frac{1}{2} \end{cases} \tag{4}$$

then the input fed into neuron j of hidden layer L_B will be

$$\text{net}c_j = \sum_{h=1}^{q} W_{hj} b_k \tag{5}$$

and its output is given as

$$c_j = f_j(\text{net}c_j) \tag{6}$$

For hidden layer L_B, the activation function is chosen as a continuous differentiable sigmoid function

$$f_h(x) = \frac{1}{1 + \exp(-x)} \tag{7}$$

The input fed into neuron h in layer L_B is given as

$$\text{net}b_h = \sum_{i=1}^{p} W_{ih} a_i \tag{8}$$

thus its output can be expressed as

$$b_h = f_h(\text{net}b_h) \tag{9}$$

Since the piecewise linear function given by Eq.(4) is not differentiable, its derivative can be defined as

$$f_j(x) = 1 - \lambda x^2 \qquad \lambda \in (0, 1) \tag{10}$$

Thus by the use of Eqs. (2)、(5) and (9), we obtain the general error of neuron i in the output layer

$$d_j = \frac{\partial E_j}{\partial(\mathrm{net}c_j)} = \frac{\partial \rho(r_i)}{\partial r_i}\frac{\partial r_i}{\partial(\mathrm{net}c_j)} = \frac{r_j}{1+r_j^2}(1-\lambda c_j^2) \tag{11}$$

where E_j is the cost function.

Analogously, the general error of neuron h in L_B is

$$e_h = \frac{\partial E_h}{\partial(\mathrm{net}b_h)} = b_h(1-b_h)\sum_{j=1}^{n} d_j W_{hj} \tag{12}$$

By applying delta rule to adjust the connection weights and threshold values, we have

$$\Delta W_{ih}(n) = \alpha a_i e_h(n) + \beta \Delta W_{ih}(n-1) \tag{13}$$

$$\Delta W_{hj}(n) = \eta b_j d_j(n) + \beta \Delta W_{hj}(n-1) \tag{14}$$

$$\Delta \gamma_j(n) = \eta d_j(n) + \beta \Delta \gamma_j(n-1) \tag{15}$$

$$\Delta \theta_h(n) = \alpha e_h(n) + \beta \Delta \theta_h(n-1) \tag{16}$$

respectively, where parameters $\alpha, \eta, \beta \in (0, 1)$, β is the momemtum.

Based upon the above equations, the modified BP algorithm can be described as below:

1) Initialize $\{W_{ih}\}$, $\{W_{hj}\}$, $\{\theta_h\}$, and $\{\gamma_j\}$ by radomly assigned values between ± 1.0.

2) Randomly select an I/O pattern (A^k, C^k) for the MLP network.

3) Compute the input and output of neuron h in the hidden layer by Eq. (8) and

$$b_h = f_h\left(\sum_{i=1}^{p} W_{ih} a_i + \theta_h\right) \tag{17}$$

where $f_h(\circ)$ is given by Eq. (7).

4) Compute the input and output of neuron j in the output layer by Eq. (5) and

$$c_j = f_j\left(\sum_{h=1}^{q} W_{hj} b_h + \gamma_j\right) \tag{18}$$

where $f_j(\circ)$ is given by Eq. (4).

5) Adjust $\{W_{ih}\}$, $\{W_{hj}\}$ $\{\theta_h\}$ and $\{\gamma_j\}$ by the use of Eq. (13) ~ Eq. (16).

6) Repeat 1) ~5) until the error between $\{C_j\}$ and its target pattern being sufficiently small.

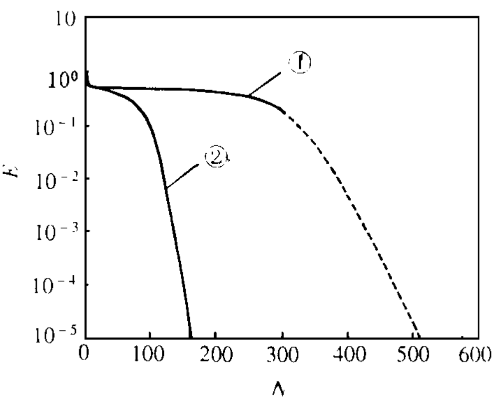

Fig. 2 The iteration process of solving the XOR Problem

4 Simulation

In this paper, we demonstrated the performance of the hybrid approach by applying the algorithm to both the XOR and the encoder/decoder problems with a three-layered MLP network. The first experiment is to solve the XOR problems. To do this, the algorithm dipicted above is used to train a 2-2-1 MLP network. The numerical simulation results are given in Fig. 2, in which curve ① corresponds to $\beta=0$, and curve ② corresponds to $\beta=0.8$. Both curves use the parameter values $\alpha=\eta=0.4$, $\lambda=0.01$. No oscillation phenomenon is observed in the iteration process. The I/O pattern after the training can be seen in Tab. 1.

The encoder/decoder problem is a more difficult problem than the XOR. The convergence results are given in Tab.2, in which the training parameter values $\alpha=\eta=0.4$, $\lambda=0.01$ and $\beta=0.8$. The convergence criterion $E=10^{-3}$. Obviously the results in Tab.1 and Tab.2 show that the hybrid approach is better than the conventional BP algorithm.

Tab. 1 I/O pattern of the trained MLP

Input	Output	
	$\beta=0.8$	$\beta=0$
0 0	0.002 741	0.002 999
1 1	0.001 391	0.001 299
0 1	0.997 794	0.007 484
1 0	0.997 920	0.008 385

Tab. 2 Simulations on encoder/decoder problems

Size	Trials	Average Iterations	
		LMS	hybrid
4-2-4	20	5 792	56
6-3-6	20	7 219	113

5 Conclusion

The algorithm based on a hybrid approach has been proved via two examples to be efficient in enhancing the learning speed of a BP network and other of its performances will be furtherly studied.

References

1 Weir M K. A method for self-determination of adaptive learning rates in back propagation. IEEE Transon NN, 1991, 11:371~379

2 Mu Wenquan, Liao Xiaofeng, Yu Juebang. Performance analysis of neural network with non-differentiable activation funcation. J of UESTC, 1996, 25:377~380

3 Jaroud B, Makhoul J. A new error criterion for poster probability estimation with neural nets. Proc of I-JC-NN′90, San Diego, 1990, 1:185~192

4 Kadir Liano. Robust error measure for superivsed neural network learning with outliers. IEEE Trans on NN, 1996, 7:246~250

5 Liao Xiaofeng, Mu Wenquan, Yu Juebang. Robust BP theory and algorithms based on several kinds of error estimators. Proc of ICSP′96, Beijing, 1996, 2:1 453~1 456

一种快速BP算法的研究①

刘光远* 虞厥邦

（电子科技大学光电子技术系 成都 610054）

邱玉辉

（西南师范大学 重庆 400715）

【摘要】 提出了一种新的BP算法，其实质是将传统的激活函数（Sigmoid函数）改为分段线性函数，将均方误差估计器改为具有稳健特性的柯西误差估计器，大大加快了收敛速度。以XOR问题和编/解码问题为例的计算机模拟实验证实了算法的有效性。

关 键 词 神经网络； BP算法； 稳健误差估计器； 异或问题； 编/解码问题

中图分类号 TP39； TP31

编辑 叶 红

① 1997年6月11日收稿，1997年12月4日修改定稿

* 男 36岁 博士生 副教授