Fuzzy Parameter Adaptation in Neural Systems

Jai J. Choi* Payman Arabshahi, Robert J. Marks II† Thomas P. Caudell‡

Abstract

In common neural network practice, algorithms require heuristic rules. We introduce a better way of using heuristics by means of a fuzzy logic controller to adjust neural network parameters in conventional neural network training. Specific attention is given to control of learning parameters in ART 1 and back-propagation.

1 Introduction

Many nonlinear optimization algorithms, or classification procedures, utilize heuristics that are in general not well specified and are thus to some degree fuzzy. It is therefore natural to attempt to apply fuzzy set theory to such algorithms and techniques with the hope of improving their performance.

A general architecture of a neuro-fuzzy system is shown in Fig. 1. There is an attribute of the neural network that we wish to control. In the ART 1 example to follow, it is the number of classes. In the layered perceptron example, it is low training error. The difference between the target and actual attribute is fed into a fuzzy parameter controller. In ART 1 this parameter is the vigilance. In back-propagation the attributes will be the step size and momentum parameters. The output of the neural network and the actual performance attribute are the same for back-propagation learning. They differ for ART 1. The model in Fig. 1 can be applied to other models, including Kohonen's self organizing maps and the layered perceptrons trained by other parameterized methods (*e.g.* random search [1]).

*The author is with Boeing Computer Services, P.O.Box 24346, 6C-04, Seattle, WA 98124. E-mail:jai@sol.boeing.com

†The authors are with the Dept. of Electrical Engineering, University of Washington, FT-10, Seattle, WA 98195

‡The author is with Boeing Computer Services, P.O. Box 24346, 7L-22, Seattle, WA 98124

The on-line fuzzy controller is used to adapt the value of a parameter according to certain heuristics. This in turn results in a change in a network performance attribute. By adaptively updating the value of the parameter, we arrive at the desired value of the performance attribute.

The fuzzy controller consists of a set of fuzzy implications of the type "IF A THEN B", where A and B are fuzzy subsets on the universe of discourse of input and output, respectively. Consider for example the case of a single input, single output system and suppose there are N such implications. Each of these rules associates a fuzzy input subset to a fuzzy output subset, represented by their membership functions. Fuzzy set theory can be used to "quantify" such rule based descriptions, and as such, can serve as an interface between the imprecise descriptive nature of control and the control actions that need to be taken. Details of operation of a simple fuzzy controller can be found in [2].

In this paper, we illustrate parameter adaptation in neural systems by application to the layered perceptron and ART 1.

2 Fuzzy back-propagation

In this section we introduce a fuzzy logic controlled implementation of the back-propagation (BP) algorithm for layered feed forward networks. This is a more sophisticated version of the fuzzy BP algorithm considered in [3].

Our approach is to adjust the learning parameter based on Jacobs' [4] heuristics by use of a fuzzy controller for automatic tuning of the learning parameter depending on the shape of the error surface. Details of the heuristics and Jacobs' use of them in learning parameter adjustment are in our previous paper [3]. Fuzzy control of BP can greatly improve the speed of the training of a layered perceptron.

0-7803-0559-0/92 $3.00

2.1 Problem setting

The back-propagation algorithm [5] is a gradient descent search algorithm in the space of network weights, and aims to minimize an energy function E, normally defined as the sum of squared output errors,

$$E(\vec{w}) = \sum_{k=1}^{L}(t_k - y_k)^2, \quad (1)$$

where t_k and y_k are the target and neural network output of the k^{th} output unit, respectively. L represents the total number of output units. Weight vectors $\vec{w}$ will be updated according to:

$$\vec{w} \Longleftarrow \vec{w} - \alpha\frac{\partial E}{\partial \vec{w}}, \quad (2)$$

or

$$\Delta\vec{w} = -\alpha\nabla E(\vec{w}) + \eta\Delta\vec{w},$$

where $\Delta\vec{w}$ is the change in weight vector; α is the learning parameter, and η is the momentum gain.

2.2 Fuzzy controller for fast BP convergence

The central idea behind fuzzy control of the BP algorithm is the implementation of heuristics used for faster convergence in terms of fuzzy IF ..... THEN rules. The heuristics we are using are mostly those of Jacobs (except for the momentum gain adjustement for which Jacobs does not provide a heuristic). They are driven by the behavior of the error E (Eq. 1). In the following heuristics the change of error CE is an approximation of the gradient and the change of CE (CCE) is the 2nd order gradient information which is related to the acceleration of convergence:

- IF CE is small with no sign changes in several consecutive time steps, THEN the value of the learning parameter should be increased.
- IF sign changes occur in CE for several consecutive time steps, THEN the value of the learning parameter should be reduced with no regard to the value of CCE.
- IF CE is very small AND CCE is very small, with no sign change for several consecutive time steps, THEN the value of the learning parameter as well as the momentum gain should be increased.

Notice that the sign change of the gradient is identical to the sign change of the CE. For example, if $E_{t-1} \leq E_t$ and $E_t > E_{t+1}$ then $CE_t = E_t - E_{t-1} \geq 0$ and $CE_{t+1} = E_{t+1} - E_t < 0$. This means there has been a sign change in route from the $(t-1)^{th}$ iteration to the $(t+1)^{th}$ iteration. Let us therefore introduce the sign change parameter,

$$SC_t = 1 - \left|\frac{1}{2}\{sgn(CE_{t-1}) + sgn(CE_t)\}\right|, \quad (3)$$

where the hard limiter $sgn(x) = 1$ if $x \geq 0$ and 0 otherwise. The factor $\frac{1}{2}$ is to ensure SC is either 0 (no sign change) or 1 (one sign change). The cumulative sum of SC (or CSC) thus can reflect the history of the sign changes, i.e.,

$$CSC_t = SC_t + SC_{t-1} + SC_{t-2} + \cdots.$$

The bigger the CSC, the more frequent the sign changes have occured. We use a five step tracking of the sign changes, and thus define

$$CSC_t = \sum_{m=t-4}^{t} SC_m.$$

The heuristic rules for this application are shown in the form of a table in Table 1. The fuzzy values that CE and CCE can take on (*NB, NS, ZE, PS, PB*) are defined in terms of their membership functions in Fig. 2. From this table for instance, one can read the following rule:

- IF CE is negative small, THEN IF CCE is zero, THEN $\Delta\alpha$ is positive small.

CE / CCE	NB	NS	ZE	PS	PB
NB	NS	NS	NS	NS	NS
NS	NS	ZE	PS	ZE	NS
ZE	ZE	PS	PS	PS	ZE
PS	NS	ZE	PS	ZE	NS
PB	NS	NS	NS	NS	NS

Table 1: Decision Table for the fuzzy controller. Table contents represent the value of the fuzzy variable $\Delta\alpha$ for a given choice of values for CE & CCE, for $CSC_t \leq 2$.

The universe of discourse for both CE and CCE is [-0.3, 0.3]. Values outside of this limit are clamped to -0.3 and 0.3 respectively.

CE / CCE	NL	NS	ZE	PS	PL
NL	-.01	-.01	--	--	--
NS	-.01	--	--	--	--
ZE	--	.01	.01	.01	--
PS	--	--	--	--	-.01
PL	--	--	--	-.01	-.01

Table 2: Decision Table for the fuzzy controller. Table contents represent the value of the fuzzy variable $\Delta\eta$ for a given choice of values for CE & CCE. -- denotes no adaptation. The maximum value that η can take on is set to 1.

2.3 Results

We present here results of a comparison between regular BP, Jacobs' Delta-bar-delta rule (DBD), and fuzzy BP as applied to a detection problem. The neural network is trained to distinguish between a constant signal corrupted with Laplace noise, and pure Laplace noise [6]. Typical training curves are shown in figures 3-5. Fuzzy BP results in dramatically faster convergence, and has a significantly smaller tail than regular BP. Note that the scales of the plots differ.

3 Fuzzy control of ART 1

In the Adaptive Resonance Theory model [7] the number of clusters formed directly depends on the value of the vigilance parameter ρ. The higher the value of ρ, the greater the number of clusters required to represent the input data. In applications where the number of desired clusters is known before hand, the value of ρ can be incrementally changed so that finally, complete classification into the fixed number of clusters is achieved. We propose to do this incremental updating of the value of the vigilance parameter automatically by means of a simple external fuzzy controller.

Assume that the number of clusters we wish to obtain is known *a priori* and is equal to some desired number N_d. The set of all input vectors is presented to the stand-alone ART 1 network and classified accordingly into N_a (actual) classes. This preliminary classification is undesirable as the requirement that N_d clusters should be formed has not been satisfied. Some adjustment of the vigilance parameter followed by another presentation of the input data is thus necessary. For instance, if $N_d < N_a$, an increase in the value of ρ would be desired, and if $N_a > N_d$, then we would want to decrease ρ. The question naturally arises as to the magnitude of the change in ρ that would make $N_a = N_d$ (in a single sweep after the initial classification), or let N_a approach N_d gradually (after multiple sweeps).

If one chooses to change ρ in small increments of $\delta\rho$, then a simple fuzzy controller can be used to arrive at the optimum value for ρ. The controller will seek to regulate the value of the vigilance parameter based on how far we are from achieving N_d classes, having started out with N_a clusters. Its policy can be formulated as a set of rules of the form shown in Table 3.

Following the controller's updating of the value of ρ by $\delta\rho$, the ART 1 network performs a second classification with this new value for ρ. If the number of classes resulting from this classification is again unsatisfactory, we repeat the process described above. Eventually, the number of classes formed by the network N_a will approach and become equal to N_d, at which point the classification process is over.

The heuristic rules for this application are shown in the form of a Table in Table 3. The fuzzy values that E and CE can take on (*NB, NS, ZE, PS, PB*) are defined in terms of their membership functions in Fig. 6. Here $E = N_d - N_a$ and CE is the change in E.

E / CE	NB	NS	ZE	PS	PB
NB	--	NS	ZE	PS	--
NS	NB	NS	ZE	PS	PB
ZE	NB	NS	ZE	PS	PB
PS	NB	NS	ZE	PS	PB
PB	--	NS	ZE	PS	--

Table 3: Decision Table for the fuzzy controller of ART 1. Table contents represent the value of the fuzzy variable $\delta\rho$ for given values of $E = N_d - N_a$ and CE.

3.1 Results

Shown in Fig. 7 is the number of classes in an ART 1 classifier versus the vigilance parameter. We used the same Boeing parts data used by Caudell *et. al.* [8]. The fuzzy vigilance controller, for a fixed number of classes, always found the appropriate vigilance parameter. In this one dimensional proof of principle example, the advantage of a fuzzy vigilance control over a simpler search approach (*e.g.* interval halving) is not obvious. We anticipate that in a more complex ART architecture [8], fuzzy multidimensional vigilance control will be quite effective.

4 Conclusion

We have presented in this paper the general structure of a neuro-fuzzy controller, applicable to many diverse neural systems. As an example, we considered fuzzy control of the back-propagation training technique for multilayer perceptrons where significant speedup in training was observed. We also considered fuzzy control of the number of classes in an ART 1 classifier. This can be advantageous in situations where the number of classes that we wish to classify our input data into, is known *a priori*.

Acknowledgments

This work was supported in part by Boeing Computer Services and the Washington Technology Center.

References

[1] J.J. Choi, S. Oh, and R.J. Marks II, "Training layered perceptrons using low accuracy computations," *Proc. IJCNN,* Singapore, Vol. 1, pp. 554-559, 1991.

[2] W. Pedrycz, **Fuzzy Control and Fuzzy Systems.** John Wiley, 1989.

[3] P. Arabshahi, J.J. Choi, R.J. Marks II and T.P. Caudell, "Fuzzy Control of Backpropagation," *Proc. First IEEE International Conf. Fuzzy Systems,* San Diego, CA, 1992.

[4] R. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks,* Vol. 1, pp.295-307, 1988.

[5] D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, *Parallel Distributed Processing (Vol. 1),* MIT Press, Cambridge, MA, 1986.

[6] C.F. Bas and R.J. Marks II, "Layered Perceptron versus Neyman Pearson optimal detection," *Proc. IJCNN,* Singapore, Vol. 2, pp. 1486-1489, 1991.

[7] B. Moore, "ART1 and Pattern Clustering," *Proc. 1988 Connectionist Summer School,* Morgan Kaufmann Publishers, 1988.

[8] T. Caudell, S. Smith, G. Johnson and D. Wunsch, "An application of neural networks to group technology," *Proc. of SPIE,* Orlando, May 1991.

[9] Zadeh, L.A., "Fuzzy sets," *Inform. and Control,* Vol. 8, pp.338-353, 1965.

Fig. 2(a): Membership functions for CE. The same membership functions are used for CCE. $NB \equiv$ negative big, $NS \equiv$ negative small, $ZE \equiv$ zero, $PS \equiv$ positive small, $PB \equiv$ positive big.

Fig. 2(b): Membership functions for $\Delta\alpha$.

Fig. 3: Typical learning curve for the Laplace noise detection problem using regular BP (4 input neurons, 1 output, one 15 unit hidden layer, $\alpha = 0.9$, $\eta = 0.5$, 800 training data).

Fig. 4: Typical learning curve for the Laplace noise detection problem using Jacobs' DBD rule (4 input neurons, 1 output, one 15 unit hidden layer, initial $\alpha = 0.9$, $\eta = 0.5$, 800 training data).

Fig. 5: Typical learning curve for the Laplace noise detection problem using Fuzzy BP (4 input neurons, 1 output, one 15 unit hidden layer, initial $\alpha = 0.9$, initial $\eta = 0.5$, 800 training data).

Fig. 6(a): Membership functions for E.

Fig. 6(b): Membership functions for CE.

Fig. 6(c): Membership functions for $\delta\rho$.

Fig. 7: Number of classes versus vigilance parameter for fuzzy controlled ART 1 (total of 47 classes, data from [8]).

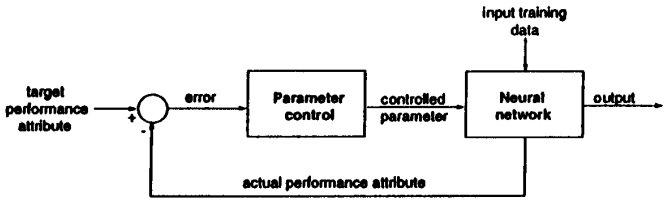

Fig. 1 Neuro-fuzzy system architecture

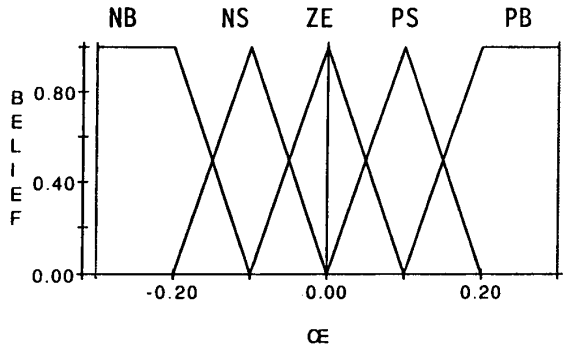

Fig. 2(a): Membership functions for CE. The same membership functions are used for CCE. $NB \equiv$ negative big, $NS \equiv$ negative small, $ZE \equiv$ zero, $PS \equiv$ positive small, $PB \equiv$ positive big.

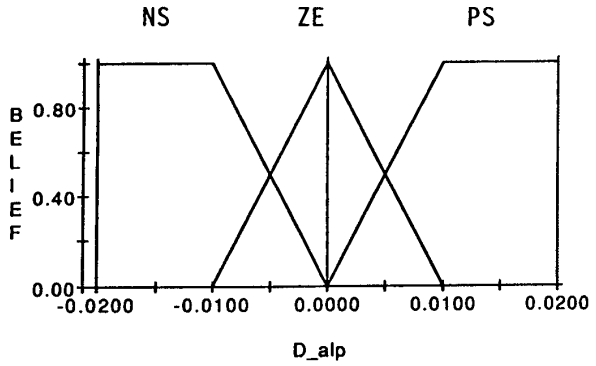

Fig. 2(b): Membership functions for $\Delta\alpha$.

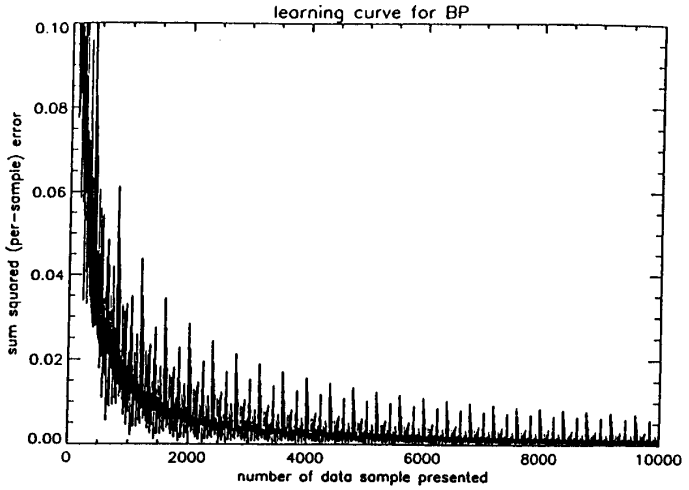

Fig. 3: Typical learning curve for the Laplace noise detection problem using regular BP (4 input neurons, 1 output, one 15 unit hidden layer, $\alpha = 0.9$, $\eta = 0.5$, 800 training data).

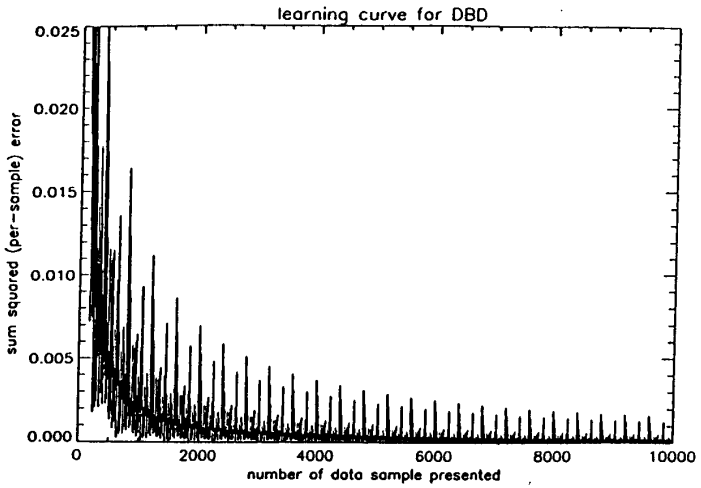

Fig. 4: Typical learning curve for the Laplace noise detection problem using Jacobs' DBD rule (4 input neurons, 1 output, one 15 unit hidden layer, initial $\alpha = 0.9$, $\eta = 0.5$, 800 training data).

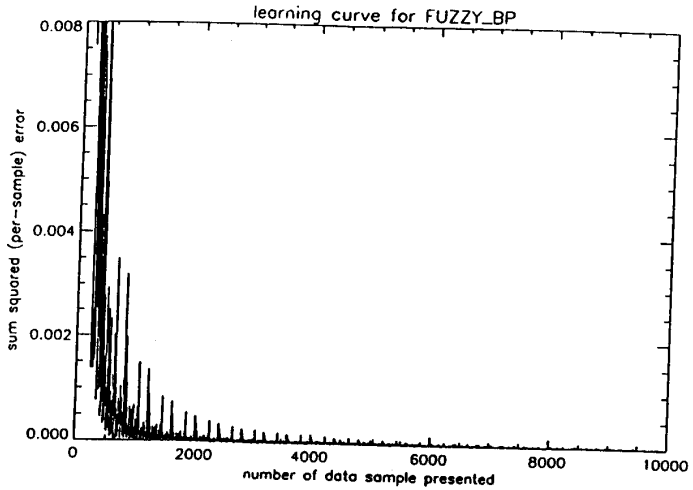

Fig. 5: Typical learning curve for the Laplace noise detection problem using Fuzzy BP (4 input neurons, 1 output, one 15 unit hidden layer, initial $\alpha = 0.9$, initial $\eta = 0.5$, 800 training data).

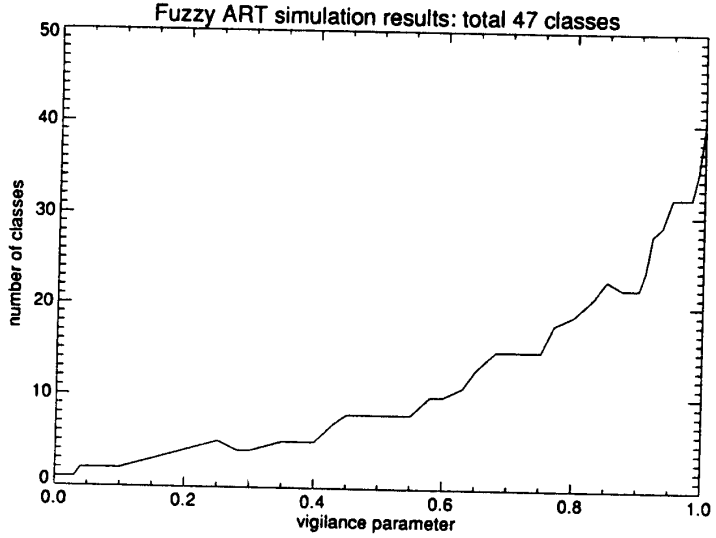

Fig. 7: Number of classes versus vigilance parameter for fuzzy controlled ART 1 (total of 47 classes, data from [8]).

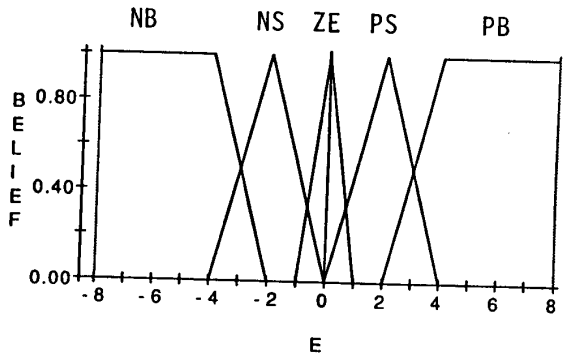

Fig. 6(a): Membership functions for E.

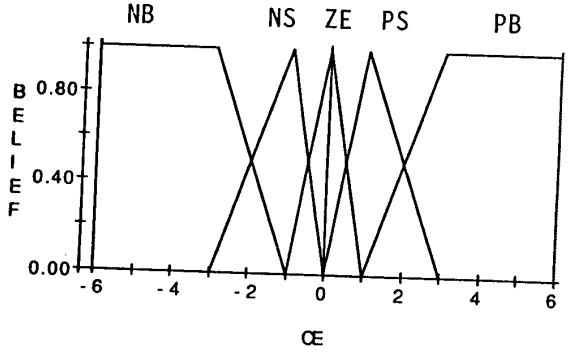

Fig. 6(b): Membership functions for CE.

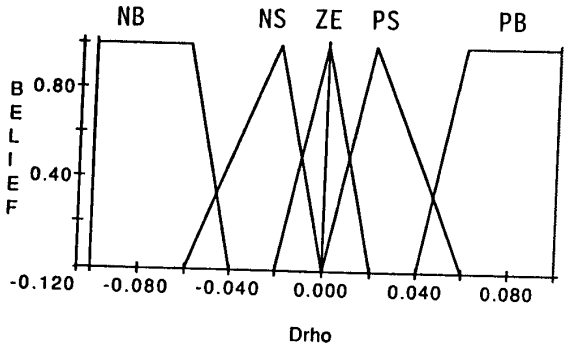

Fig. 6(c): Membership functions for $\delta\rho$.