Reduction of Neural Network Training Time Using an Adaptive Fuzzy Approach in Real Time Applications

Hamidreza Rashidy Kanan and Mahdi Yousefi Azar Khanian

***Abstract*—A major problem of neural network in real-time applications is their long training time. We present a modification of the neural network (NN) for reduction of training time. Unlike traditional training time reduction algorithms, we propose a new Adaptive Fuzzy technique to create ensembles of neural network using multiple projections of the same data obtained from different NNs. The purpose of this paper is to demonstrate the optimization of training that occurs with the application of fuzzy controller theory to neural network. A fuzzy system is employed to control the learning parameters of a neural network to reduce the possibility of overshooting during the learning process. Hence, the learning time can be shortened. This paper compares the training efficiency and accuracy between a NN and a fuzzy controlled neural network, when they are required to carry out the same assignment. We justify the suitability of the proposed method by some experiments in soccer robot trajectory generation tasks; the resulting fuzzy controlled neural network indicates a significant reduction in the training time by 30%.**

***Index Terms*—Neural Network, Adaptive Fuzzy Logic Controller, Backpropagation Learning Algorithm, Mobile Robot, Trajectory Generation.**

I. Introduction

Highlight Neural network performance is directly related to the size and quality of training samples. The ability of a mobile robot to generate real-time trajectory by neural network in unknown and unstructured environments by relying only on its sensory system is regarded as the key issue in an enormous number of research publications. In order to achieve its goal, the robot is usually required to determine in real-time a safe and smooth path from a starting location to an end location (target). Among all soft-computing methods, fuzzy logic based inference and neural network have been found to be the most attractive techniques that can be utilized for this purpose. Fuzzy system is tolerant to noise and error in the information coming from the sensory system, and most importantly it is a factual reflection of the behavior of human expertise. Neural network have been proven to be very efficient at handling a wide range of engineering applications [1]. The evolvement of neural network paradigms have provided a powerful tool to deal with mobile robot trajectory generation process, which exhibits incomplete and uncertain knowledge due to the inaccuracy and imprecision inherent from the sensory system. A formal training procedure is required by the neural network in order to learn. In general, it is useful only when the network architecture (i.e., model) is chosen correctly. Too small network cannot learn the problem well, but too large network will lead to over-generalization and thus poor performance [2]. Depending on the nature of data, hidden layers with many neurons may add a large amount of time to training phase and not produce any observable reduction in squared error. However, training a multilayer network with more than one hidden layer is slow and further it will not also give good results [3]. This is due to this fact that the weights at deep hidden layers are hardly optimized. The time required for training such a network is also extremely high. Training a system to make decision in the presence of uncertainties is a difficult problem, especially when computational resources are limited. However, the desirability of past decisions can usually be assessed after that, outcomes of their implementations are observed. Therefore, unsupervised training methods that do not utilize those assessments cannot take full advantage of the available knowledge. Adaptive Fuzzy controller systems have structured numerical knowledge that can exploit noisy or inexact situations [2, 3]. By employing a fuzzy control system to adaptively vary the learning parameters of the network, the training time can be reduced significantly. The fuzzy control system uses "expert knowledge" to determine the specific change in learning parameters.

The back-propagation algorithm has a major drawback: a long period of training time. That is, for any given problem, the network must have many solved examples presented repeatedly during the learning procedure before the network learns the data with an acceptable degree of accuracy. Therefore, the learning stage is a lengthy process. As far as we concerned, limited study directly has ever tried to combine the fuzzy set theory with neural network learning algorithms in order to control the learning parameters and reduce the training time in real-time applications. This process can be short time by using fuzzy controller systems. Incorporation of fuzzy techniques can improve performance in these cases. The method proposed in this paper takes advantages of the concept of the adaptive fuzzy controller systems to tune the learning parameters and reduce the training time by 30% in trajectory generation tasks for soccer robots.

The rest of the paper is organized as follows. Section II introduces some preliminaries about neural network basic training models, back-propagation learning algorithm, and standard training convergence time. The novel piecewise adaptive fuzzy logic control system is proposed in Section III. In Section IV, the case study on determination of learning parameters with adaptive fuzzy controller is present to demonstrate the effectiveness of the proposed approach. This

Manuscript received March 11, 2012; revised April 27, 2012.

H. R. Kanan is with the Department of Electrical Engineering, Bu-Ali Sina University, Hamedan, Iran (Tel.+98-811 8257409; fax: +98-811 8257400; 65175-4161; email: h.rashidykanan@basu.ac.ir).

M. Y. A. Khanian is with the Electrical, Computer and IT Engineering Department, Qazvin Branch, Islamic Azad University, Qazvin, Iran.

section describes how the learning process is streamlined. In addition to the training efficiency, the neural network still obtains an impressive performance, which is evidenced by the experimental results, discussed in Section IV. F. Finally, some conclusions are drawn in Section V.

II. Basic Training Algorithms

A neural network (NN), an information-processing centre, mimics the human brain with respect to operation and processing ability. Neural network can be successfully used as a forecasting tool due to capability in identifying non-linear relations. In order to "learn" a solution, a training strategy must be used. One of the most popular training methods is backpropagation [4].

A. Backpropagation Learning Algorithm

The Backpropagation algorithm is one of the most popular and versatile forms of neural network classifiers used for the recognition of complex patterns. In a typical backpropagation training phase for a multilayer network, the supervisor presents input data to the network and compares the network's actual output, O_p, with the target (or desired output), T_p:

$$Er = \frac{(T_p - O_p)^2}{2} \tag{1}$$

This difference, or error, is used to change the connection weights between neurons in the network:

$$\Delta w_j(t+1) = \eta\left(\frac{-\partial(Er)}{\partial w_j}\right) + \alpha \Delta w_j(t) \tag{2}$$

With the back-propagation algorithm, neural network weights are adjusted in a gradient descent manner, which means the minimization of the error between the expected output and the actual output for a particular input. However, two major problems exist with this kind of the networks. First, slow convergence rate and second, the possibility of settling into a local minimum [5]. A potentially large number of iterations are required to train the network until it learns the data with an acceptable degree of accuracy.

B. Final Standard Training Convergence Time

A major problem of a back-propagation learning algorithm is its slow convergence time [6]. During training, a NN is trying to converge to the global minimum with the shortest path. Global minimum is the maximum performance that the neural network can yield given the size of the network. At the beginning of a learning process, the network learns rapidly, and its value of RMS error decreases fast. In general, the value of RMS reflects how a neural network is performing, and computed by equation (3).

$$Error_{RMS} = \sqrt{\frac{\sum_p \sum_k (t_{kp} - x_{kp})^2}{n_p n_o}} \tag{3}$$

Finally, the NN may complete its learning, and possibly takes thousands iterations of data presentations.

III. Adaptive Fuzzy Logic Controller

An Adaptive fuzzy logic control system consists of four components as shown in Figure 1. [7,8]. A fuzzy logic controller (FLC) is an intelligent control system that smoothly interpolates between rules, i.e. rules fire to continuous degrees and the multiple resultant actions are combined into an interpolated result.

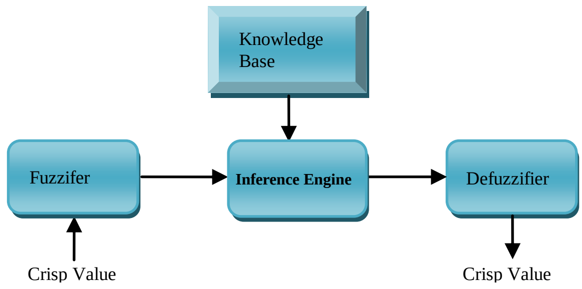

Fig. 1. A general scheme of a Fuzzy Logic Controller.

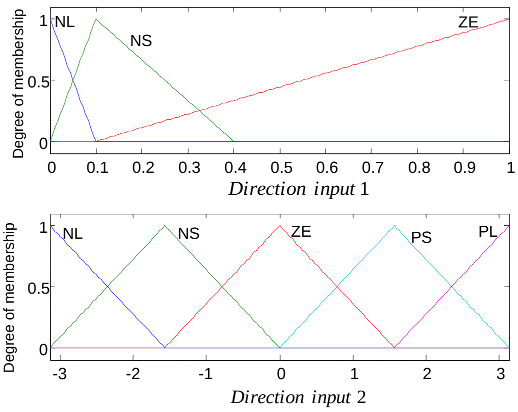

Fig. 2. An example of linguistic variables.

Implementation of a fuzzy controller requires assigning membership functions for inputs and outputs. Inputs to a fuzzy controller are usually measured variables, associated with the state of the controlled plant, that are fuzzified (assigned membership values) before being processed by an inference engine. The heart of the controller inference engine is a set of if-then rules whose antecedents and consequences are made up of linguistic variables and associated fuzzy membership functions. Consequences from fired rules are numerically aggregated by fuzzy set union and then collapsed (defuzzified) to yield a single crisp output as the control signal for the plant. For detailed introductions to fuzzy control, fuzzy set operations, and concepts of fuzzification, inference, aggregation, and defuzzification see one of [7, 9]. There are two reasons that fuzzy logic control systems are preferred: the infinite measurement resolution, and imprecise linguistic descriptions. Due to using fuzzy sets and linguistic variables, there is no limited resolution as it does in a conventional control system. "Forced efficiency" on the output can be the result caused by low resolution. An example of the use of linguistic variables is shown in Figure 2. In addition to use linguistic variables, fuzzy logic control systems can encapsulate complex input/output operations in the form of rules. Also, the operations of most control systems are based on “expert knowledge”.

IV. Determination of Learning Parameters

The implementation of fuzzy logic control system to adaptively determine learning parameters in neural network is discussed in this section. The experimental results are presented, too.

A. Tuning Membership Function

The procedure of the algorithm has several major steps and consists of clustering the data into classes. The membership functions are then generated from the classes obtained. The algorithm is described for one parameter, and as follows.

B. Find Difference Between Adjacent Values

Sort the values of each attribute of the training instances in a descending sequence and find the maximum attribute value and the minimum attribute value of each attribute. Randomly choose n instances as the training instances, and let the remaining n instances be the testing instances [9]. Then, we can find the maximum attribute value and the minimum attribute value of each species of the n training instances. Given a data set, there are *n* training samples. The values for the parameter in question, $\vec{X} = x_1, x_2, \ldots, x_n$ are sorted into ascending order, the values are sorted to find an association between adjacent values. Let the maximum attribute value and the minimum attribute value of any two attributes of each species of the training instances is the boundaries to form a plane. The difference between adjacent values in the sorted data is determined. The difference obtained will provide a way to calculate the similarity between adjacent values. The difference for a set of training set data is:

$$\Delta_i = X_{j+1} - X_j \quad , j = 1, 2, 3, \ldots, n-1 \tag{4}$$

where, X_j and X_{j+1} are adjacent values in the sorted data.

C. Find Similarities Between Adjacent Values

The following equation finds the similarities between adjacent values and maps them into real numbers between 0 and 1.

$$D_i = \begin{cases} 1 - \dfrac{X_i - \Delta_i}{C_p \times \sigma_s} & for \;\; \Delta_i \le C_p \times \sigma_s \\ 0 & otherwise \end{cases} \tag{5}$$

where, Δ_i is the difference between adjacent data, σ_s is the standard deviation of Δ_i, and C_p is the control parameter. The control parameter is used to determine the shape of the membership function.

D. Define the Membership Function for Each Class

One of the simplest membership functions is the triangular membership function, and will be used for the rest of the equations. The triangular membership function for class j consists of three parameters: the central vertex point, b_j, and the two endpoints, a_i and p_j. The central vertex point is determined for each class and is calculated by the following equation:

$$b_j = \frac{x_i \times p_i + x_{i+1} \times \dfrac{p_i + p_{i-1}}{2} + \cdots + x_{k-1} \times \dfrac{p_{k-2} + p_{k-1}}{2} + x_k \times p_{k-1}}{p_i + \dfrac{p_i + p_{i+1}}{2} + \cdots + \dfrac{p_{k-2} + p_{k-1}}{2} + p_{k-1}} \tag{6}$$

where *j* represents the j^{th} class, *k* represents the ending data index for this class, i.e., data x_i through x_k fall into class j, and p_i is the similarity between x_i and x_{i+1}. The endpoints of the membership function, a_j and c_j, are obtained using interpolation. The following equation determines the left and right endpoints.

$$\begin{cases} a_j = b_j - \dfrac{b_j - x_i}{1 - \mu_j(x_i)} \\ p_j = b_j + \dfrac{x_k - b_j}{1 - \mu_j(x_k)} \end{cases} \tag{7}$$

where, a_j is the left endpoint and c_j is the right endpoint. $\mu_j(x_i)$ and $\mu_j(x_k)$ is the membership determined by using equation (8) :

$$\mu_j(x_i) = \mu_j(x_k) = min(p_i, p_{i+1}, \cdots, p_{k-1}) \tag{8}$$

where, *k* represents the maximum data index value within the class. The membership functions for the classes determined by this method are shown in Figure 3.

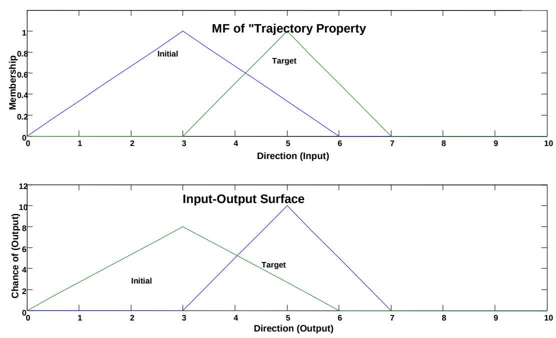

Fig. 3. Determining the membership functions.

E. Find The Proposed Approach

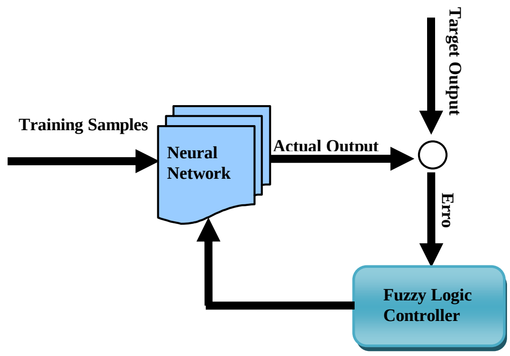

Fig. 4. A basic architecture of fuzzy logic controlled neural network.

A fuzzy control system to adaptively determine learning parameters (i.e. learning rate and momentum) in neural network is the focus of this study. As proposed in [10], this contrasts with regular neural network, in which the learning parameters are fixed. The on-line fuzzy logic controller is utilized to adapt the learning parameters based on the RMS error generated by the neural network. A basic architecture for such an integrated system is shown in Figure 4.

The purpose of the FLC is to automatically adjust the learning rate η and α momentum term according to the RMS error surface. To describe the error surface, there are four parameters used to create the rules for the FLC. They are relative error (RE), change in relative error (CRE), sign change in error (SC) and cumulative sum of sign change in error (CSC) which is described as follows:

$$\begin{cases} RE(t) = E(t) - E(t-1) \\ CRE = RE(t) - RE(t-1) \\ SC(t) = 1 - \left\| \frac{1}{2} \times \left[sgn(RE(t-1) + sgn(RE(t)) \right] \right\| \\ CSC(t) = \sum_{m=t-4}^{t} SC(m) \end{cases} \qquad (9)$$

The collection of rules created (called the rule base) reflects the use of these parameters to adaptively change the neural network's learning rate and momentum. The rule base can be summarized concisely in the form of a decision table. Two of the fuzzy rule bases (decision tables) for the applications discussed in section 4.2 are shown. To demonstrate the idea, two of the fuzzy IF-THEN based rules that are employed in the FLC are listed [9], [10]:

- Rule 1:

If RE is small and CSC is less or equal to two, then the value of learning rate η and α momentum term should be increased.

- Rule 2:

If CSC is larger than three, then the value of learning rate η and α momentum term should be decreased regardless the value of RE and CRE.

Table I describes the rule base for the change in the learning parameter, $\Delta\eta$, and Table II describes the rule base for the change in momentum, $\Delta\alpha$. In these decision tables, the utilization of five linguistic variables has been used. Specifically, NL represents a "Negative Large" value, NS is "Negative Small," ZE is zero, PS is "Positive Small," and PL is "Positive Large".

TABLE I: DECISION TABLE FOR $\Delta\eta$, WHEN CSC $\le$ 2.

		RE				
		NL	NS	ZE	PS	PL
CRE	NL	NS	NS	NS	NS	NS
	NS	NS	ZE	PS	ZE	NS
	ZE	ZE	PS	ZE	NS	ZE
	PS	NS	ZE	PS	ZE	NS
	PL	NS	NS	NS	NS	NS

TABLE II: DECISION TABLE FOR $\Delta\alpha$, WHEN CSC $\le$ 2.

		RE				
		NL	NS	ZE	PS	PL
CRE	NL	-0.01	-0.01	0	0	0
	NS	-0.01	0	0	-0.01	0
	ZE	0	0.01	0.01	0.01	0
	PS	0	0	0	0	0.01
	PL	0	0	0.01	0	0.01

F. Experimental Results

Here we describe a fuzzy MLP for rule generation [11] and present its effectiveness in soccer robot trajectory generation problems. At the end of the training phase the network is supposed to have encoded the input–output information distributed among its connection weights. This constitutes the knowledge base of the desired decision-making system. Handling of imprecise inputs is possible and natural decision is obtained associated with a certainty measure denoting the confidence in the decision. Moreover, we have included both rule extraction and rule refinement in the broader perspective of rule generation.

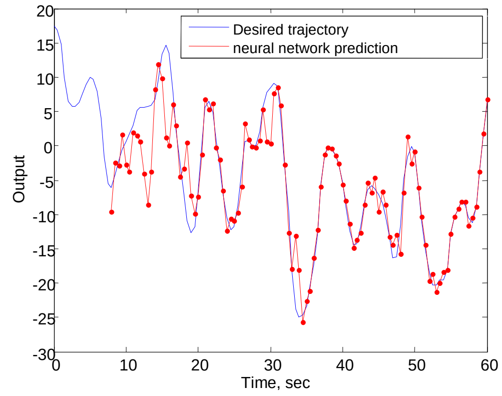

Fig. 5. Accuracy of neural network controller.

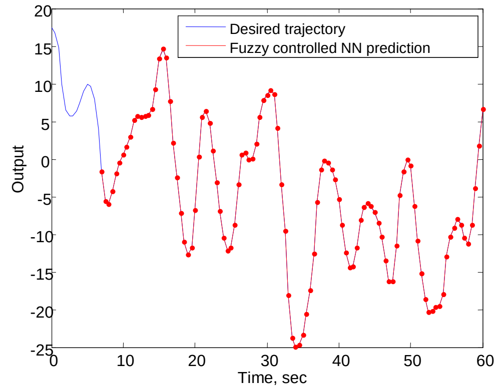

Fig. 6. Accuracy of fuzzy controlled neural network.

The system was tested on a classic application area, to soccer robot trajectory generation. The fuzzy logic controlled neural network showed 30% average improvement in training time. That is, in comparision with the performance of a regular neural network with different value learning rate, and momentum term trained in the normal fashion, the fuzzy logic controlled system requires about 2/3 of the iterations (measured to convergence by an error limit), which only takes less than 5 seconds in our PC with 3.0 GHz CPU and 1Gbyte RAM. Futhermore, the fuzzy logic controlled system assesses performance with each iteration, and the performance is significantly improved. In a specific case when both systems have a learning rate of 0.8 and momentum term of 0.8, the regular NN fails to learn in about 100 iterations. Figures 5 and 6 illustrate the efficiency and performance of the proposed algorithm. Here the NN output with the proposed new algorithm is much closer to the original index data while the fuzzy logic controlled neural network system is able to continue learning. It should be mentioned that the "best" predictions were obtained from our first control file in which we utilized two hidden layers of neurons. It also "learned" much faster than the first example.

From Table III the proposed approach has the best MSE and standard deviation. In addition, it is faster and it has best success 92% to reach the goal.

TABLE III: THE COMPARISON BETWEEN THE AVERAGE OF THE TRAINING PERFORMANCE BETWEEN NN, AND THE FUZZY CONTROLLED NEURAL NETWORK.

Algorithm	MSE	Standard deviation	Success
NN	5.61013e-012	3.0766e-003	89%
The Proposed Approach	5.62941e-013	2.3754e-003	92%

V. CONCLUSION

This study indicated that the neural network's training time is reduced significantly by combining an adaptive fuzzy control system with a neural network in order to adaptively vary the learning parameters and reduces the training time in real-time applications. The method proposed in this paper takes advantages of the concept of the adaptive fuzzy controller systems to tune the learning parameters and reduce the training time by 30% in trajectory generation tasks for soccer robots and it yields 92% accuracy. In addition, this technique can reduce the possibility of overshooting and sometimes help the network get out of a local minimum .The network's ability to converge during training and the final performance are dependent on the learning parameters. Our study reinforces this fact, as our simulations have presented that a "wrong" value of learning rate can lead to poor accuracy. Furthermore, the methodology is flexible, and can be exercised on other neural network applications.

REFERENCES

[1] G. Leng, T. M. McGinnity, and G. Prasad, "Design for self-organizing fuzzy neural networks based on genetic algorithms," *IEEE Trans. on Fuzzy Systems*, vol. 14, issue. 6, pp. 755-766, Dec. 2006.

[2] G. E. Hinton, and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science, vol. 313, no. 5786, pp. 504–507, July 2006.

[3] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in Advances in Neural Information Processing Systems, MIT Press, 2007, ch. 19, pp. 153-160.

[4] L. Huijun, "A nonlinear predictive model based on BP neural network," in *Proc. Control and Decision Conference (CCDC)*, China, 2010, pp. 73-77.

[5] D. Srinivasan, X. Jin, and R. L. Cheu, "Adaptive neural network models for automatic incident detection on freeways," Neurocomputing, vol. 64, pp. 473-496, March 2005.

[6] S. Zhou, Z. Q. Liu, and J. Y. Zhang, "Fuzzy causal networks: general model, inference, and convergence," *IEEE Trans. on Fuzzy Systems*, vol. 14, issue. 3, June 2006.

[7] J. Zhang, "Modelling and optimal control of batch processes using recurrent neuro-fuzzy networks," *IEEE Trans. on Fuzzy Systems*, vol. 13, no. 4, pp. 417-427, August 2005.

[8] M. C. Choy, D. Srinivasan, and R. L. Cheu, "Neural networks for continuous online learning and control," *IEEE Trans. on Neural Networks*, vol. 17, issue. 6, pp. 1511-1531, Nov. 2006.

[9] D. A. Kaur, and K. Kaur, "Fuzzy expert systems based on membership functions and fuzzy rules," in *Proc. International Conf. on Artificial Intelligence and Computational Intelligence (AICI)*, 2009, pp.513-517.

[10] S. M. Bai, and S. M. Chen, "Automatically constructing grade membership functions of fuzzy rules for students' evaluation," *Expert Systems with Applications*, vol. 35, issue. 3, pp. 1408-1414, October 2008.

[11] T. H. Yu, K. H. Huarng, "A neural network-based fuzzy time series model to improve forecasting," *Expert Systems with Applications*, vol. 37, issu. 4, pp. 3366-3372, April 2010.

Hamidreza Rashidy Kanan was born in 1978 in Hamedan. He received B.Sc. degree in Electronic Engineering from Tabriz University in 2000, M.Sc. degree in Electronic Engineering from Tarbiat Modarres University with the first rank in 2002 and Ph.D. degree in Electronic Engineering from Amirkabir University of Technology (Tehran Polytechnic) in 2008. From 2008 to 2009, he was employed as a Research Scientist at Iran Telecommunication Research Center (ITRC), ON, where he was involved in research for multimodal biometrics. In 2009, he joined the Faculty of Engineering at Bu-Ali Sina University, Hamedan, Iran, as an Assistant Professor. His major research interests include digital image processing, pattern recognition, computer vision, face detection and recognition, facial expression recognition, and gender classification.

Mahdi Yousefi Azar Khanian was born in 1983 in Qazvin. He received B.Sc. degree in Electronic Engineering from Islamic Azad University (Qazvin Branch) in 2007, M.Sc. degree in Computer Engineering (Artificial Intelligence) from Islamic Azad University (Qazvin Branch) in 2011 and He is a Ph.D. student in Biomedical Engineering in Amirkabir University of Technology (Tehran Polytechnic) since 2011. From 2003 to 2009, he was employed as a Researcher at Mechatronic Research Laboratory (MRL). His major research interests include Artificial intelligence and robotics, Theory and application of biological signal processing, human motion analysis, Computational neuroscience and neural engineering.