Fifth IFSA World Congress (1993), 997 - 1000

Learning Algorithms of Fuzzy Counterpropagation Networks

Chi-Cheng Jou and Chi-Hsiao Yih

Department of Control Engineering
National Chiao Tung University
Hsinchu, Taiwan 30050, R.O.C.

Abstract

This paper presents a fuzzy neural network, called the fuzzy counterpropagation network, that structures its inputs and generates its outputs in a manner based on counterpropagation networks. The fuzzy counterpropagation network is developed by incorporating the concept of fuzzy clustering into the hidden layer responses. Three learning algorithms are introduced for use with the proposed network. Simulations demonstrate that fuzzy counterpropagation networks with the proposed learning algorithms work well on approximating bipolar and continuous functions.

I. Introduction

Counterpropagation networks proposed by Hecht-Nielsen (1987) provide an excellent example of hybrid learning networks. Counterpropagation networks involve a hidden layer between input and output layers. The weights associated with the connections between input and hidden layers are trained by unsupervised learning, and the weights associated with the connections between hidden and output layers are trained by supervised learning. The most appealing feature of counterpropagation networks is that they can usually be trained substantially faster than conventional back-propagation networks. However, there are a number of difficulties with the performance of counterpropagation networks. They often generalize less well on new patterns and form poorer approximations between input-output relationships than back-propagation networks. In many cases, the number of hidden units in a counterpropagation network must be large for the network to be practically useful. Using too few hidden units results in coarse boundaries between pattern clusters. This may lead to poor approximations, and occasionally the first layer becomes unstable during unsupervised training.

In this paper we present a fuzzy neural network, called the *fuzzy counterpropagation network*, that structures its inputs and generates its outputs in a manner based on counterpropagation networks. Since the fuzzy counterpropagation network is developed by incorporating the concept of fuzzy clustering into the hidden layer responses, it can be viewed as a result of applying fuzzy computing to neural network techniques. In this paper we show that this network can be trained to achieve a better level of accuracy than a counterpropagation network of comparable size. Three learning algorithms are introduced for use with the proposed network; all have the same unsupervised learning part but different supervised learning parts. Simulations demonstrate that fuzzy counterpropagation networks with the proposed learning algorithms work well on approximating bipolar and continuous functions.

II. The Model

Consider a counterpropagation network with n inputs, m hidden units, and p output units, in which input terminals are denoted by x_i, hidden units by h_j, and output units by y_k. There are connections w_{ij} from the inputs to the hidden units and v_{jk} from the hidden units to the output units (see Fig. 1). For a given input vector $\mathbf{x}^r$, hidden unit j generates h_j^r, specifying whether or not input vector $\mathbf{x}^r$ activates hidden unit j as winner:

$$h_j^r = \begin{cases} 1 & \text{if } \|\mathbf{x}^r - \mathbf{w}_j\| \le \|\mathbf{x}^r - \mathbf{w}_l\|, \text{ for all } l; \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Thus, a single hidden unit wins the competition and becomes activated, and the remaining hidden units become inactive.

In the fuzzy counterpropagation network proposed here, the hidden units in the network have normalized activation functions as follows (Jou, 1992):

$$h_j^r = \frac{(\|\mathbf{x}^r - \mathbf{w}_j\|^2)^{-\alpha}}{\sum_{l=1}^m (\|\mathbf{x}^r - \mathbf{w}_l\|^2)^{-\alpha}} \tag{2}$$

where $\alpha > 0$ is the weighting exponent. Note that h_j^r increases to 1 as input $\mathbf{x}^r$ approaches $\mathbf{w}_j$ and gradually decreases to 0 as $\mathbf{x}^r$ moves away from $\mathbf{w}_j$. Thus, hidden unit j gives a maximum response to the input vector nearest to $\mathbf{w}_j$.

Given pattern r, output unit k simply produces

$$y_k^r = \sum_{j=1}^m h_j^r v_{jk}. \tag{3}$$

The linear combination of the fuzzy activations of hidden units provides a kind of interpolation and extrapolation. This leads to significantly improved accuracy in mapping approximation.

III. Learning Algorithms

Below we present three learning algorithms, each combining unsupervised learning and supervised learning. The three learning algorithms have the same unsupervised learning part but different supervised learning parts. The supervised learning part always follows the unsupervised learning part of the algorithm.

A. Unsupervised Learning

When an input vector $\mathbf{x}^r$ is entered into the network, all the hidden units compete on the basis of the distances between their weight vectors $\mathbf{w}_j$ and the input vector. The on-line fuzzy c-means clustering rule (Bezdek, 1981) prescribes for the change of the weights w_{ij} the expression

$$\Delta w_{ij} = \eta_w (h_j^r)^{1+1/\alpha}[x_i^r - w_{ij}]. \tag{4}$$

The goal of the unsupervised learning scheme is to draw the weight vectors $\mathbf{w}_j$ so that relevant features of input stimuli are encoded. Since $h_j^r > 0$ for every j, r, this learning rule prevents stagnant weight vectors (or *dead units*), which may occur in standard competitive learning (Rumelhart and Zipser, 1985). The usual final result of the unsupervised learning scheme is that the weight vectors $\mathbf{w}_j$ distribute themselves in an almost equiprobable configuration in the input space.

B. Supervised Learning

For the networks we are considering, the error measure or cost function for learning a training set of input-output pairs $\{(\mathbf{x}^r, \mathbf{t}^r)\}$ is given by

$$E = \frac{1}{2}\sum_r \sum_{k=1}^p [t_k^r - y_k^r]^2. \tag{5}$$

This is a continuous differentiable function of every weight, so we can use gradient descent to train appropriate parameters.

Algorithm 1

After the unsupervised learning process has ended the weight vectors $\mathbf{w}_j$ are frozen. The weights v_{jk} of the network are trained by the gradient descent rule as

$$\Delta v_{jk} = \eta_v h_j^r [t_k^r - y_k^r] \tag{6}$$

where $0 < \eta_v < 1$. The hybrid learning scheme presented here is not optimal, since the hidden layer responses are based on input stimuli only and are not optimized with respect to output performance.

Algorithm 2

In addition to adjusting the weights v_{jk} in the second layer, it is desirable to modify the weights w_{ij} in the first layer during supervised learning. Since each hidden unit output is a continuous differentiable function of the input, the adaptation rule for w_{ij} can be derived by gradient descent as

$$\Delta w_{ij} = \frac{2\eta_w' h_j^r [x_i^r - w_{ij}]}{\|\mathbf{x}^r - \mathbf{w}_j\|^2} \sum_{k=1}^p [t_k^r - y_k^r][v_{jk} - y_k^r] \tag{7}$$

where $0 < \eta_w' < 1$.

Algorithm 3

In the above discussion the weighting exponent α is assumed to be constant. We may generalize the activation of each hidden unit in such a way that the hidden unit has the following normalized activation:

$$h_j^r = \frac{(\|\mathbf{x}^r - \mathbf{w}_j\|^2)^{-\alpha_j}}{\sum_{l=1}^m (\|\mathbf{x}^r - \mathbf{w}_l\|^2)^{-\alpha_l}} \tag{8}$$

where hidden unit j has its own weighting exponent α_j. In addition to modifying the weights w_{ij} and v_{jk} in the network, we may vary the weighting exponents α_j during supervised learning. The updating rule can also be computed using gradient descent

$$\Delta\alpha_j = \eta_\alpha h_j^r \ln(\|\mathbf{x}^r - \mathbf{w}_j\|^2) \sum_{k=1}^p [t_k^r - y_k^r][y_k^r - v_{jk}]. \tag{9}$$

where $0 < \eta_\alpha < 1$.

VI. Simulation Examples

To illustrate the computational capabilities of fuzzy counterpropagation networks and the effectiveness of the learning algorithms described above, we now present the results of simulations in which these networks and algorithms are applied to problems of function approximation. For each problem, four learning schemes are used, namely, one counterpropagation network using the hybrid learning algorithm (CPN) and three fuzzy counterpropagation networks using Algorithms 1, 2, and 3 proposed in this paper (FCPN1, FCPN2, and FCPN3, respectively).

In the first example, the four learning schemes were to learn a continuous function with two inputs and one output given by $f(x_1, x_2) = \cos(4\pi x_1)\cos(4\pi x_2)\exp[-10(x_1^2 + x_2^2)]$, where $-1 \leq x_1, x_2 \leq 1$ (see Fig. 2). The training patterns were generated from the given function so that at each training step, a training pattern was randomly selected and presented to the networks. The parameters involved in learning are listed in Table 1(a). To examine the effect of network size on learning performance, three different network sizes were used: 10, 100, and 1000 hidden units. All the networks were trained for 10,000 steps during unsupervised learning and 300,000 steps during supervised learning. In general, a fuzzy counterpropagation network should form a better approximation than a standard counterpropagation network of comparable size. Examination of the rms errors shows that the fuzzy counterpropagation networks using Algorithm 1 did not work much better than the standard counterpropagation networks. This is because in Algorithm 1 the hidden layer responses are not optimized with respect to output performance. Learning performance improved when the weights in both layers were adjusted during supervised learning (Algorithm 2), and improved further when Algorithm 3 was used. Fig. 3 illustrates the three-dimensional representations synthesized by CPN, FCPN1, and FCPN2 with 1000 hidden units and FCPN3 with 100 hidden units after learning the given continuous function.

Our second example is to learn to classify the training patterns in a two-dimensional space into two classes, as shown in Fig. 4. The training pattern was associated with a class value 1 if it belonged to class 1 (the interior of the "cross") and a class value −1 if it belonged to class 2 (the exterior of the "cross"). Thus, the networks we used had two inputs and one output, and they learned the mapping from feature vector to class value. Two network sizes, 20 and 100 hidden units, were used; all the networks were run with identical learning rates, as shown in Table 1(b). During unsupervised learning, the networks were trained for 10,000 steps. During supervised learning, the training process was continued for 100,000 steps. Fig. 5 depicts the three-dimensional representations of the potential fields synthesized over the two-dimensional input space by the networks with 100 hidden units. The decision boundaries learned by the networks are shown in Fig. 6. The counterpropagation network had only one winning hidden unit, and thus a sharp three-dimensional representation was obtained. FCPN1 had a smoother three-dimensional representation than CPN did, but no significant improvement was observed. Accurate decision boundaries were obtained by both FCPN2 and FCPN3. Examining Fig. 5 and 6 reveals that FCPN3 provided a better three-dimensional representation than FCPN2 did, since the weighting exponents controlling the hidden layer responses were adjusted by FCPN3.

V. Conclusion

In this paper we have introduced the concept of a fuzzy counterpropagation network and presented a number of learning algorithms. The fuzzy counterpropagation network architecture was developed by incorporating the concept of fuzzy clustering into a standard counterpropagation network. Because they relax the winner-take-all constraint, fuzzy counterpropagation networks are able to exhibit interpolating behavior. Three learning algorithms with the same unsupervised part but different supervised parts were introduced. The network using the hybrid learning algorithm (Algorithm 1) did not really work much better than a standard counterpropagation network trained by a similar hybrid algorithm. A back-propagation algorithm (Algorithm 2) was then developed so that all the weights of the fuzzy counterpropagation network were optimized with respect to output performance. The great appeal of this scheme lies in its computational capability and the small number of hidden units it requires. In addition to adjusting the weights in the network, we may also modify the weighting exponents using gradient descent, which results in Algorithm 3. The last algorithm demonstrated the best learning performance, since all the parameters involved in the network were adjusted to minimize the output error. The present exercise provides a suitable justification for current efforts to characterize unknown systems using the fuzzy neural network approach.

References

Bezdek, J.C. (1981). *Pattern recognition with fuzzy objective function algorithm.* New York, NY: Plenum Press.

Hecht-Nielsen, R. (1987). Counterpropagation networks. *Applied Optics*, 26, 4979-4984.

Jou, C.C. (1992). A fuzzy competitive learning algorithm for clustering. *Proceedings of the International Joint Conference on Neural Networks*, III, 631-636.

Rumelhart, D.E. and D. Zipser (1985). Feature discovery by competitive learning. *Cognition Science*, 9, 75-112.

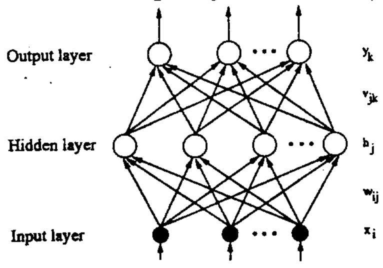

Figure 1: A counterpropagation network.

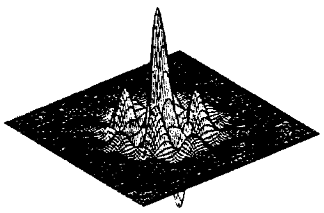

Figure 2: A continuous function to be learned by the neural networks.

(a)

Scheme	η_w	η_v	η'_w	η_α	α
CPN	0.3	0.2	-	-	-
FCPN1	0.3	0.2	-	-	2.0
FCPN2	0.3	0.2	0.02	-	2.0
FCPN3	0.3	0.2	0.02	0.8	-

(b)

Scheme	η_w	η_v	η'_w	η_α	α
CPN	0.3	0.2	-	-	-
FCPN1	0.3	0.2	-	-	2.0
FCPN2	0.3	0.2	0.01	-	2.0
FCPN3	0.3	0.2	0.01	0.1	-

Table 1: Learning parameters.

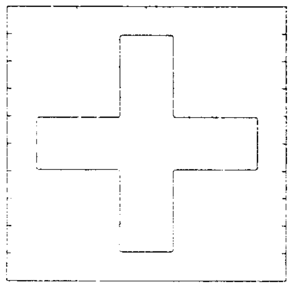

Figure 4: A set of bipolar training patterns to be learned by the neural networks.

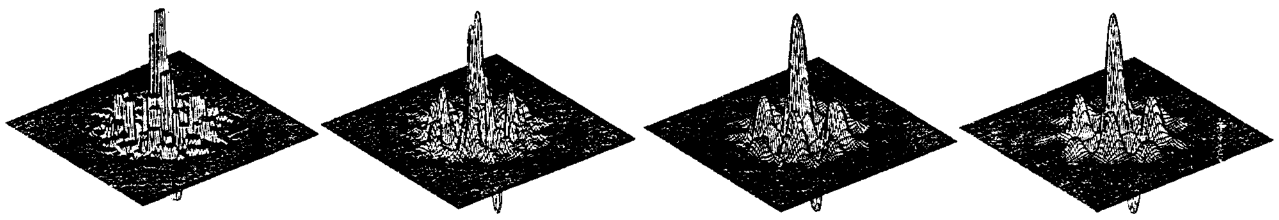

Figure 3: Three-dimensional representations synthesized by (a) CPN, (b) FCPN1, (c) FCPN2, and (d) FCNP3 after learning the continuous function.

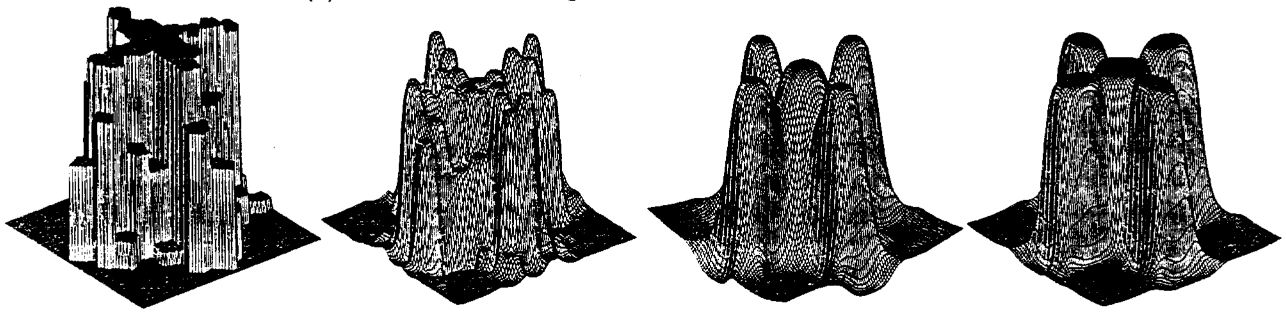

Figure 5: Three-dimensional representations synthesized by (a) CPN, (b) FCPN1, (c) FCPN2, and (d) FCNP3 after learning the set of bipolar training patterns.

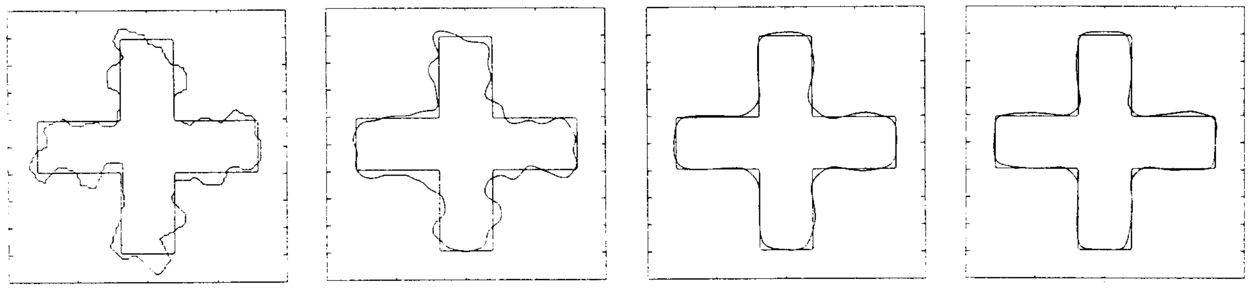

Figure 6: Decision boundaries learned by (a) CPN, (b) FCPN1, (c) FCPN2, and (d) FCNP3 on the set of bipolar training patterns.