Proceedings of the 4th WSEAS/IASME Int. Conf. on System Science and Simulation in Engineering, Tenerife, Spain, December 16-18, 2005 (pp211-216)

Modular General Fuzzy Hypersphere Neural Network

Pradeep M. Patil, S N Kulkarni, D D Doye and U V Kulkarni
Electronics and Computer science and engineering.
SGGS College of Engineering and Technology, Vishnupuri, Nanded.
INDIA.

Abstract: This paper describes Modular General Fuzzy Hypersphere Neural Network (MGFHSNN) with its learning algorithm, which is an extension of General Fuzzy Hypersphere Neural Network (GFHSNN) proposed by Kulkarni, Doye and Sontakke [1] that combines supervised and unsupervised learning in a single algorithm so that it can be used for pure classification, pure clustering and hybrid classification/clustering. MGFHSNN offers higher degree of parallelism since each module is exposed to the patterns of only one class and trained without overlap test and removal, unlike in Fuzzy Hypersphere Neural Network (FHSNN) [2], leading to reduction in training time. In proposed algorithm each module captures peculiarity of only one particular class and found superior in terms of generalization and training time with equivalent testing time. Thus, it can be used for voluminous realistic database, where new patterns can be added on fly.

Keyword: MGFHSNN, GFHSNN, FHSNN, MGFHLSNN, GFHLSNN, FHLSNN

1 Introduction

Fuzzy neural networks have become very popular and widely being used in the pattern recognition applications. Basically, there are two main training strategies employed by fuzzy neural networks; supervised and unsupervised learning. In supervised learning, class labels are provided with input patterns and the decision boundary between classes that minimizes misclassification is achieved. It is often referred as pattern classification problem. In unsupervised learning, training patterns are unlabeled and clusters of the patterns are formed with a suitable similarity measure, which is referred as clustering problem. In real high dimensional problems, the data is often a mixture of labeled and unlabeled instances. To make full use of all the available information carried by both labeled and unlabeled patterns, some attempts have been made by combining supervised and unsupervised learning in a single training algorithm.

Many papers using fuzzy neural networks are reported on studies of pattern classification and clustering. Pedrycz and Waletzky have shown that even a small percent of the labeled patterns substantially improve the result of clustering [3]. Gabrys and Bargiela have proposed general fuzzy min-max neural network (GFMM) for clustering and classification [4], which is an extension of
FMN, with a fusion of supervised and unsupervised learning [5,6]. Patil, Kulkarni and Sontakke have proposed general fuzzy hyperline segment neural network (GFHLSNN) [7] and found to be the best among all for recognition. Doye and Sontakke [8] have proposed modular general fuzzy min-max neural network, which is used for speech recognition of Marathi language (Language spoken in the state of Maharashtra, INDIA) and shown that the modular network has better average recognition accuracy. Patil, Kulkarni and Sontakke [9,10] have proposed modular fuzzy hypersphere neural network (MFHSNN), and modular fuzzy hyperline segment neural network (MFHLSNN) and found superior than FHSNN and FHLSNN in terms of generalization and training time with equivalent testing time.

In this paper we present the MGFHSNN algorithm, which is an extension of GFHSNN and it is applied for rotation invariant handwritten character recognition. Ring and Zernike features are used as feature extraction methods. Its performance is also tested using Fisher Iris database.

This paper is organized as follows. The topology and its learning algorithm are described in section II and III, respectively. The performance comparison of MGFHSNN with FHSNN, FMN and FNN algorithms using standard databases is presented in section IV. Finally the conclusions are stated in section V. The notations used in this paper are kept consistent with the original paper introducing GFHSNN, as far as possible for reference and comparison purposes.

2 Topology

During training phase, K modules of GFHSNN are used, if database consists of patterns of K number of classes as shown in Fig. 1, in which each module is a simple two layer feed forward neural network that grows adaptively to meet the demands of the problem. The first layer accepts the n-dimensional input pattern and second layer consists of hyperspheres

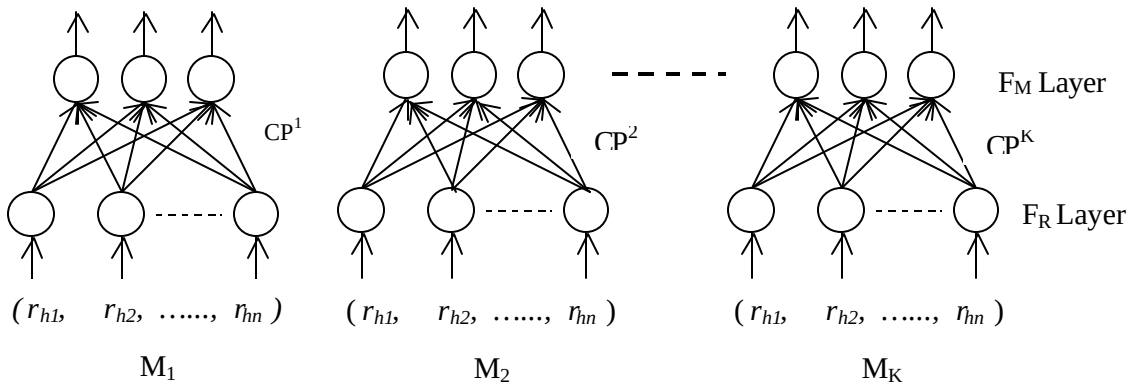

Fig. 1: Modules of MGFHSNN during training phase.

(HSs) that are created during training. Each module is trained with patterns of that class to which it represents. Hence, each module learns peculiar ities of a single class.

For any k^{th} module, the weights between first and second layer represents center points and radii of the HSs created during learning. These weights are stored in a matrix CP^k. Each row in CP^k is $(n+1)$ dimensional vector in which first n components represents center point and $(n+1)^{th}$ component contains radius of the HS. A center point, radius and a fuzzy membership function, characterizes each HS in a module. The fuzzy membership function returns values between 0 and 1. The processing performed by j^{th} fuzzy HS node in k^{th} module, i.e. m_j^k, is shown in the Figure 2.

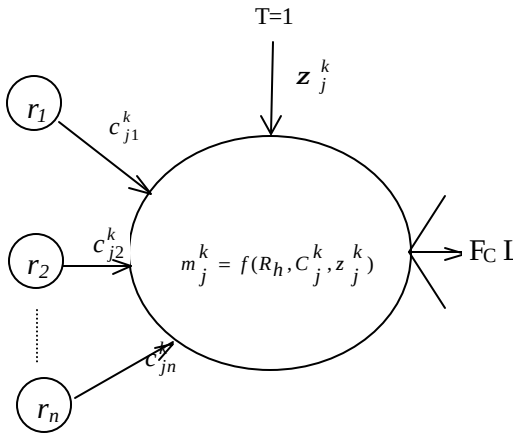

Figure 2: Implementation of Fuzzy HS

The threshold input of HS denoted as T is set to one and it is weighted by z_j^k, where z_j^k represents radius of HS m_j^k, which is updated during training. The maximum size of HS is bounded by a user defined value λ, called as growth parameter where $0 \le \lambda \le 1$. Hence, λ puts maximum limit on the radius of the HS. Assuming the training set defined as $R = \{R_h \mid h = 1,2,.., P\}$, where $R_h = (r_{h1}, r_{h2}, ..., r_{hn})$ is the h^{th} pattern, and representing center point of HS m_j^k, as $C_j^k = (c_{j1}^k, c_{j2}^k, ..., c_{jn}^k)$, the membership function of the HS node m_j^k is defined as,

$$m_j^k(R_h, C_j^k, z_j^k) = 1 - f(l, z_j^k, g), \quad (1)$$

where $f(\cdot)$ is three-parameter ramp threshold function defined as,

$$f(l, z_j^k, g) = \begin{cases} 0 & \text{if } 0 \le l \le z_j^k \\ lg & \text{if } z_j^k < l \le 1 \\ 1 & \text{if } l > 1 \end{cases} \quad (2)$$

and the argument l is defined as,

$$l = \left(\sum_{i=1}^{n} \left(c_{ji}^k - r_{hi} \right)^2 \right)^{1/2}. \quad (3)$$

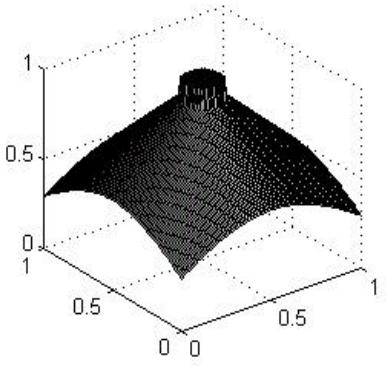

Fig. 3: Membership function plot

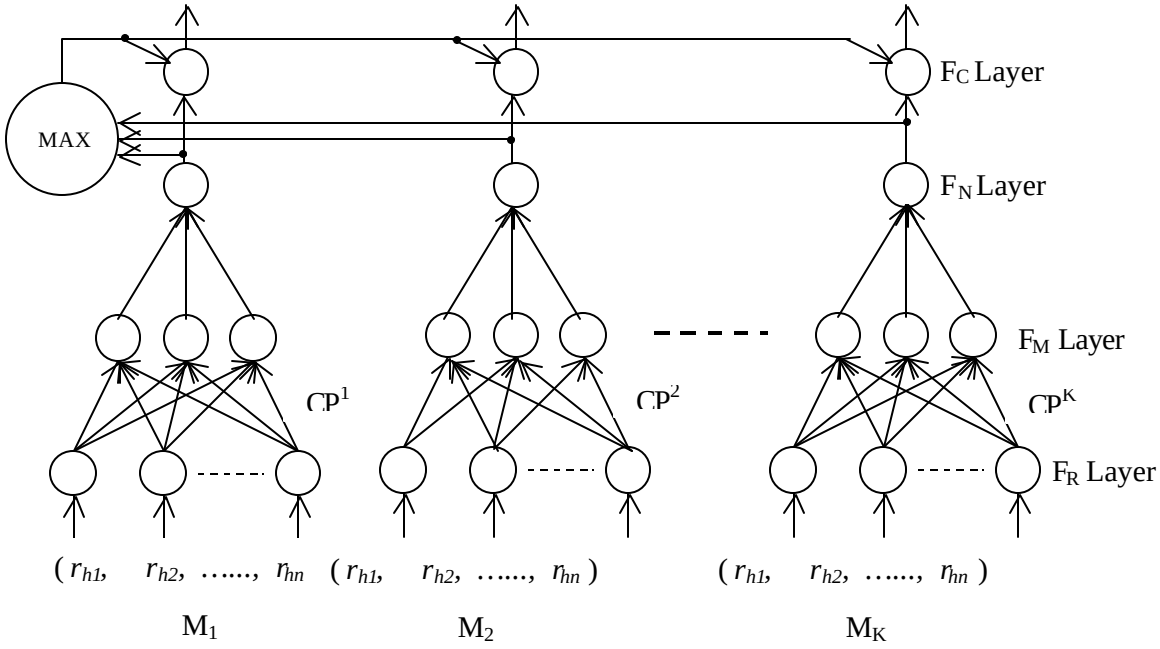

Figure 4: Architecture of MFHSNN in testing phase.

The membership function returns $m_j^k = 1$, if HS includes the pattern R_h. The sensitivity parameter g, $0 < g \le 1$, governs how fast the membership value decreases outside the HS, as the distance between R_h and C_j^k increases. The plot of membership function with center point = [0.5 0.5] and radius equal to 0.3 is shown in Fig. 3. After training the performance of MFHSNN is tested using the four-layer feedforward neural network architecture as shown in Fig. 4. The first two layers are constructed during training. The third layer uses K MAX Fuzzy Neurons (FNs), one for each module. The output of k^{th} module, n^k, is calculated as,

$$n^k = \max_{j=1}^{q^k} m_j^k \qquad \text{for } k = 1, 2, \ldots\ldots\ldots, \text{K} \tag{4}$$

where q^k represents number of HSs in k^{th} module created in training phase. Hence the output of third layer gives fuzzy decision and the output n^k indicates the degree of membership of the input pattern to the class k.

The fourth layer contains COMP-FNs defined as in [11]. Finally each F_C node delivers non-fuzzy output which is described as,

$$c^k = \begin{cases} 0 & \text{if } n^k < T \\ 1 & \text{if } n^k = T \end{cases} \qquad \text{for } k = 1 \text{ to K} \tag{5}$$

where, $T = \max\ (n^k)$, for $k = 1$ to K.

3 Learning Algorithm

The training set R consists of a set of P ordered pairs $\{R_h, d_h\}$, where $R_h = (r_{h1}, r_{h2}, \ldots . r_{hn}) \in I^n$ is the h^{th} input pattern and $d_h \in \{1,2,\ldots..,\text{K}\}$ is the index of one of the K classes. The learning algorithm of MFHSNN composed of two steps.

1. **Initialization:** All K modules are initialized by creating a HS in each with a first pattern belonging to the class of the module. This is a state when the network has K modules, each containing one HS having radius equal to zero and a center point initialized with the pattern of corresponding class.
2. **Training:** Actual training begins after the initialization. An input pattern of class k is applied to k^{th} module only and fuzzy membership of the input pattern with all the HSs within that module is calculated. After this any one of the two cases that are described below can happen.

Case 1: *Accommodation by expansion of HS:*
Each HS has maximum limit on its radius denoted by the parameter l. The pattern is included in the existing HS if radius of that HS after expansion is less than or equal to l. This constraint is stated in (6). The HS is expanded to include the input pattern by modifying its radius if the criterion stated in (6) is satisfied. This is described by following two steps.

Step 1: Determine using (1), whether the pattern R_h is contained by any one of the existing HSs. If

R_h is included then the remaining steps in the training process are skipped and the training continues with the next training pair.

Step 2: If the pattern R_h falls outside the HS, then the HS is expanded to include pattern if the expansion criterion is met.

For the HS m_j^k to include R_h,

$$\left(\sum_{i=1}^{n}\left(c_{ji}^k - r_{hi}\right)^2\right)^{1/2} \le l \ . \tag{6}$$

If the expansion criterion is satisfied then the pattern R_h is included as,

$$z_j^{k^{new}} = \left(\sum_{i=1}^{n}\left(c_{ji}^k - r_{hi}\right)^2\right)^{1/2} \tag{7}$$

Case II: *Accommodation by creation of new HS:*

If *case I* fails then to include the input pattern, a new HS is created as,

$$C_{new}^k = R_h \text{ and } z_{new}^k = 0 \ . \tag{8}$$

4 Simulation Results

MGFHSNN is implemented using MATLAB 5.1 on P-III, 733MHz PC. The results obtained are compared with FNN, FMN, and FHSNN. For comparison, same databases are used in the same sequence of data presentation. Fisher Iris data, which is well known to the pattern recognition community, is used in the experimentation, and collected from the machine-learning databases [12]. This database contains 150 patterns of 3 classes. Each class has 50 instances. Each class refers to a type of Iris plant. One class is linearly separable from other two but the latter are not linearly separable from each other. Pattern classes are Iris-Setosa, Iris-Versicolor and Iris-Virginica. Feature vector is 4-dimensional excluding class label. For recognition purpose set I, and II are prepared from Fisher Iris database by randomly selecting 75 patterns in each set. When MGFHSNN is trained with l = 0.095 with set II it has created 35 HSs and recognized 72 patterns in set I. When FHSNN is trained with l = 0.0587 for set II it has created 37 HSs and recognized 71 patterns from set I. From Table 1 it is clear that the MGFHSNN gives better recognition rate.

Table 1. Recognition with Fisher Iris data.

	Set-I	**Set-II**	l	**HS**
FHSNN	93.33	100	0.0587	37
MGFHSNN	96.00	100	0.095	35

Performance of MGFHSNN algorithm is also verified for rotation invariant handwritten character recognition and compared with FNN, FMN and FHSNN algorithms. The handwritten characters can be in arbitrary location, scale and orientation. Ten numerals from two hundred writers are scanned and stored in BMP format. The linear moment normalization discussed by Perantonis and Lisboa [13] is used to normalize characters to get translation and scale invariance. After normalization the rotation invariant ring features defined as in [14] are extracted from the normalized characters by setting ring width to two. These feature vectors are scaled within the range [0,1]. Therefore, the pattern space with 16 ring features is a 16-dimensional unit cube. The Zernike features up to order five defined in [15] are also extracted and then scaled within the range [0,1] along each dimension. Pattern space with Zernike features is 16-dimensional unit cube because; selected features start from second order moments.

The database of handwritten characters consists of two thousand characters. Four data sets are prepared from this database and used in the experiments. Set-1 is unrotated training set, i.e. original training set consisting of one thousand training patterns, which is reused to verify the recognition. Set-2 is rotated training set extracted from Set 1, i.e. each sample of Set-2 is a rotated version of sample in Set 1 with an angle of 15^0. Set 3 is unrotated testing set consisting of remaining one thousand patterns in the database that is used to evaluate generality. Set-4 is rotated testing set extracted from Set-3, i.e. each sample of Set-4 is a rotated version of sample in Set-3, with an angle of 15^0.

Table 2. Recognition with Zernike features

	Set-1	Set-2	Set-3	Set-4	Avg.
FNN	100	25.6	20.8	13.9	40.075
MFNN	100	27.9	20.1	15.4	40.85
FMN	100	51.9	28.7	21.5	50.525
FHSNN	100	53.5	29.3	24.2	51.75
MGFHSNN	100	56.05	29.5	24.5	52.51

The results obtained with Zernike features are tabulated in Table 2. The MGFHSNN algorithm created 997 HSs when trained with $l = 0.03$. Similarly, the FHSNN algorithm created 997 HSs. It is observed that the recognition rates of MGFHSNN are better for less number of HSs.

Table 3. Recognition with ring features

	Set-1	**Set-2**	**Set-3**	**Set-4**	**Avg.**
FNN	100	76.1	31.4	27.5	58.75
FMN	100	97.4	42.7	41.0	70.27
FHSNN	100	99.3	45.6	44.5	72.35
MGFHSNN	100	99.7	45.9	44.8	72.65

When the MGFHSNN algorithm is trained with ring features, its performance is found better than FNN and FMN and equiporable with the FHSNN. These results are listed in Table 3. The MGFHSNN algorithm gives better recognition rates for all the sets as compared to FNN and FMN algorithms. Tables 2 and 3 shows that recognition rates with ring features are superior than Zernike features.

The performance of these algorithms is also compared in unsupervised (pure clustering) mode using Fisher Iris data. Before presentation of data all the class labels are set to zero i.e. all patterns are made unlabeled. The performance is tested and tabulated in confusion matrices shown in Table 4. The order of data presentation is same for all the algorithms. It is observed that MGFHSNN yields less confusion than GFMM.

Table 4: Confusion matrices (a) MGFHSNN, (b) GFMM

λ=0.05 HSs=74

	1	2	3
1	100		
2	2	98	
3			100

(a)

HB size = 0.05, HBs=74

	1	2	3
1	100		
2		82	18
3			100

(b)

Timing analysis using ring features is listed in Table 5 which shows that the MGFHSNN algorithm is computationally efficient as compared to the FNN, FMN and FHSNN algorithms.

Table 5. Timing analysis

	Training time (sec)	Recall time / pattern (sec)
FNN	1228.5	1.4252
FMN	435.45	1.4986
FHSNN	200.32	0.384
MGFHSNN	97.71	0.2647

5 Conclusions

Proposed algorithm can be used for pure classification, pure clustering and hybrid classifications and clustering. It has ability to learn the patterns faster because it creates/expands HSs without any overlap test and its removal, which is a substantial overhead in FMN and FHSNN. Thus it can be used in voluminous realistic database recognition purposes where less training time is the prime demand. Percentage recognition of MGFHSNN algorithm is found superior with less number of HSs when observed for handwritten character recognition. Clustering performance of MGFHSNN gives less confusion formance of MGFHSNN gives less confusion with that of GFMM. Recognition rates with ring features are superior than Zernike features.

References:

[1] U V Kulkarni, D D Doye and T R Sontakke, "General fuzzy hypersphere neural network," IEEE-IJCNN-2002, Honolulu USA, 2002.

[2] U V Kulkarni, D D Doye, and T R Sontakke, "Fuzzy hypersphere neural network for rotation invariant handwritten character recognition," *Proc.of 10th int. IEEE Conference on Fuzzy Systems to be held* at University of Melbourne, Australia, December 2001.

[3] W Pedrycz and J Waletzky, "Fuzzy clustering with partial supervision," *IEEE Trans. on Systems Man and Cybern.,* Vol. 27, No. 5, 1997, pp. 787-795.

[4] B Gabrys and A Bargiela, "General fuzzy min-max neural network for clustering and classification," *IEEE Trans. on neural networks,* Vol. 11, No. 3, May 2000, pp. 769-783.

[5] P K Simpson, "Fuzzy min-max neural networks - part1: classification," IEEE Trans. neural networks, Vol. 3, No. 5, 1992, pp. 776-786.

[6] P K Simpson, "Fuzzy min-max neural networks - Part2: clustering", IEEE Trans. fuzzy sys. Vol.1, No.1, 1993, pp 32-45.

[7] P M Patil, U V Kulkarni and T R Sontakke, "General Fuzzy Hyperline Segment Neural Network," In proceedings of IEEE International conference on systems, Man and Cybernetics, SMCO2, Hammamet, Tunisia. Vol 4 TA 2P4 Pages 06, 2002.

[8] D D Doye, T R Sontakke, "Speech Recognition using modular General Fuzzy Min-Max Neural Network," *Journal of IETE*,2003.

[9] P M Patil, U V Kulkarni and T R Sontakke, "Modular Fuzzy Hypersphere Neural Network", In proceedings of the IEEE 12th International Conference on Fuzzy Systems, FUZZ-IEEE 03, Marriot Pavilion Downtown Hotel, St. Louis, Missouri, USA. Vol 1, 2003, pp.232-236.

[10] P M Patil, U V Kulkarni and T R Sontakke, "Modular Fuzzy Hyperline Segment Neural Network", In proceedings of the IEEE 12th International Joint Conference on Neural Networks, IJCNN 03, at Port Land, Oregon, USA Vol 3, 2003, pp.1939-1944.

[11] H K Kwan and Y Cai, "A fuzzy neural network and its applications to pattern recognition," *IEEE Trans on fuzzy systems,* Vol 2, No 3, 1994, pp 185-192.

[12] C Blake, E Keogh, and C J Merz UCI repository of machine learning databases, Univ. California, Irvine, 1998.

[13] S J Perantonis and P J G Lisboa, "Translation, rotation and scale invariant pattern recognition by high-order neural networks and moment classifiers," *IEEE Trans. neural networks*, Vol 3, No 2, 1992, pp 241-251.

[14]H P Chiu and D C Tseng, "Invariant handwritten Chinese character recognition using fuzzy min-max neural networks," *Pattern recog. letters*, Vol. 18, 1997, pp. 481-491.

[15]Khotanzad and J H Lu, "Classification of invariant image representations using a neural network," *IEEE Trans. ASSP*, Vol 38, No 6, 1990, pp. 1028-1038.