Proceedings of the World Congress on Engineering 2014 Vol I,
WCE 2014, July 2 - 4, 2014, London, U.K.

Modified Fuzzy Hyperline Segment Neural Network for Pattern Classification and Recognition

S. B. Bagal and U. V. Kulkarni

Abstract—**The Fuzzy Hyperline Segment Neural Network (FHLSNN) utilizes fuzzy set as pattern classes in which each fuzzy set is a union of fuzzy set hyperline segments. This is supervised classifier that forms n-dimensional hyperline segments defined by two end points with a corresponding membership function for learning and prediction. In this paper, we propose the modifications in the prediction phase of FHLSNN to improve its classification/recognition rate. In the first modification (MFHLSNN1), we propose the Euclidean distance computation between the input pattern and the centroid of the patterns falling on the hyperline segments, to decide the class of pattern. In the second modification (MFHLSNN2), we propose to use both, the membership value of hyperline segments for the input pattern and Euclidean distance to decide the class of pattern. The performance of both, MFHLSNN1 and MFHLSNN2 is evaluated using benchmark problems and real world handwritten character data set. The results are analyzed, discussed and compared with the FHLSNN. Both the proposed modifications improved the prediction accuracy of the FHLSNN without affecting its incremental learning.**

Index Terms—**Fuzzy hyperline segment neural network, Centroid, Euclidean distance computation, Test accuracy**

I. Introduction

THE fuzzy neural networks (FNN) combine the strength of the artificial neural networks, such as learning, adaption, fault tolerance, parallelism and generalization with human like thinking and reasoning using fuzzy logic. Because of all these advantages, the fuzzy neural networks are widely used for pattern classification and recognition. A very vast literature is available on the fuzzy neural networks which suggest various architectures and algorithms for the different applications [1].

Patrick K. Simpson proposed supervised learning neural network classifier known as fuzzy min-max neural network (FMN) that utilizes fuzzy sets as pattern classes where each fuzzy set is an aggregate (union) of fuzzy set hyperboxes. This learning algorithm has the ability to learn on-line and in a single pass through the data. Its performance is evaluated for commonly used and well-known fisher iris data set [2]. He has also proposed unsupervised fuzzy min-max clustering neural network (FMCN) in which clusters are implemented as fuzzy sets using membership function with a hyperbox core that is constructed from a min point and a max point [3]. Gabrys and Bargiela have proposed general fuzzy min-max neural network (GFMM) for classification and clustering, which is a fusion of supervised and unsupervised learning [4]. In the sequel of fuzzy min-max neural network classifier, Kulkarni U. V. *et al.* proposed fuzzy hyperline segment neural network classifier (FHLSNN), which utilizes fuzzy sets as pattern classes in which each fuzzy set is a union of fuzzy set hyperline segments [5]. This classifier is applied for rotation invariant handwritten character recognition and found superior than unsupervised four layer feedforward fuzzy neural network (FNN) of Kwan and Cai, [6] and FMN in terms of recognition rate, training time and recall time per pattern. U.V. Kulkarni *et al.* have also proposed unsupervised fuzzy hyperline segment clustering neural network and its performance is found superior as compare to FMCN when applied for clustering of Fisher Iris data [7]. P. M. Patil, U. V. Kulkarni and T. R. Sontakke have proposed general fuzzy hyperline segment neural network (GFHLSNN), which uses supervised and unsupervised learning and can be used for pure classification, pure clustering and hybrid classification/ clustering [8].

Many researchers have suggested the modification in the architecture and learning algorithm of FMN to improve its performance. Kim and Yang proposed a weighted fuzzy min-max neural network whose membership function considers the occurrences of input pattern along with frequency of occurrences [9]. In order to overcome low automation degree and to achieve the remarkable generalization capability Antonello Rizzi *et al.* proposed the two new learning algorithms for FMN as the adaptive resolution classifier (ARC) and its pruning version (PARC) [10]. Nandedkar and Biswas suggested the use of overlapped compensatory neuron and the containment compensatory neuron to resolve membership confusion in the overlapped area [11]. Reza Davtalab *et al.* proposed new fuzzy Min-Max classifier that uses modified compensatory neurons and it is online, single-pass and supervised method. In this method for handling overlapping regions that are mainly created in borders, a modified compensatory nod with a radios-based transition function is used which increases the classification accuracy in discriminating cases [12]. H. Zhang *et al.* proposed data-core based fuzzy min-max neural network (DCFMN) in which a new membership

Manuscript received March 10, 2014; revised April 1, 2014.

S. B. Bagal is Associate Professor and Head, Electronics and Telecomm. Engineering, Kalyani Charitable Trust's Late G. N. Sapkal College of Engineering, Nasik-422212, (M. S.), India (phone: +91-25-942-20174; fax: +91-25-942-20174; e-mail: bagalsaheb@ rediffmail.com).

U. V. Kulkarni is Professor and Head, Computer & Engineering Science Department, S. G. G.S. College of Engineering & Technology, Vishnupuri, Nanded-431602, (M. S.), India. (e-mail: kulkarniuv@yahoo.com).

function for classifying neuron is defined on the basis of noise, the geometric center of the hyperbox and the data core. The performance of DCFMN is evaluated for the benchmark problems and pattern classification of oil pipeline [13].

As the recognition rate i.e. test accuracy is one of the important parameter to decide the performance of pattern classifier, Anas M. Quteishat and Chee Peng Lim [14] proposed two types of modifications in the prediction phase of FMN, to improve the test accuracy. In the first approach, the Euclidean distance is computed in the prediction phase to decide the class of pattern. In the second modification, they propose to employ both the membership value of the hyperbox fuzzy sets and the Euclidean distance for classification. These modifications improve the performance of classifier, in situations when the large hyperboxes are formed by the network.

In this paper, we propose the modifications in the prediction phase of FHLSNN to improve its classification/recognition accuracy. In MFHLSNN1, we propose the Euclidean distance computation with respect to centroid to decide the class of patterns. The MFHLSNN2 utilizes both the membership value of fuzzy hyperline segments and Euclidean distance to decide the class of the patterns. The performance of MFHLSNN1 and MFHLSNN2 is evaluated using benchmark problems and real world handwritten character recognition. Also to indicate the remarkable recognition ability of MFHLSNN1 and MFHLSNN2 over FHLSNN an artificial 2-D dataset is applied. The proposed modifications show significant improvement in the classification accuracy of the FHLSNN. The results are analyzed, discussed and compared with the FHLSNN.

This paper is organized as follows. In Section II, the architecture and learning algorithm of the FHLSNN is explained. The proposed modifications are explained in Section III. The experimental procedure, simulation results, description of data sets and discussions on the results are presented in the Section IV. Finally, we conclude the paper with the Section V.

II. THE FUZZY HYPERLINE SEGMENT NEURAL NETWORK (FHLSNN)

A. Topology of FHLSNN

The readers are advised to refer the [5] for the detail description of architecture and algorithm of FHLSNN. To make this article self contained, this section explain the architecture and algorithm of FHLSNN in short. The architecture of FHLSNN consists of four layers as shown in Fig. 1. In this architecture first, second, third and fourth layer are denoted as F_R, F_E, F_D and F_C respectively. The F_R layer accepts an input pattern and consists of n processing elements, one for each dimension of the pattern. The F_E layer consists of m processing nodes that are constructed during training. There are two connections from each F_R to each F_E node. Each connection represents an end point for that particular hyperline segment. These end points are stored in the two matrices V and W. Each F_E node represents hyperline segment fuzzy set and is characterized by the membership function.

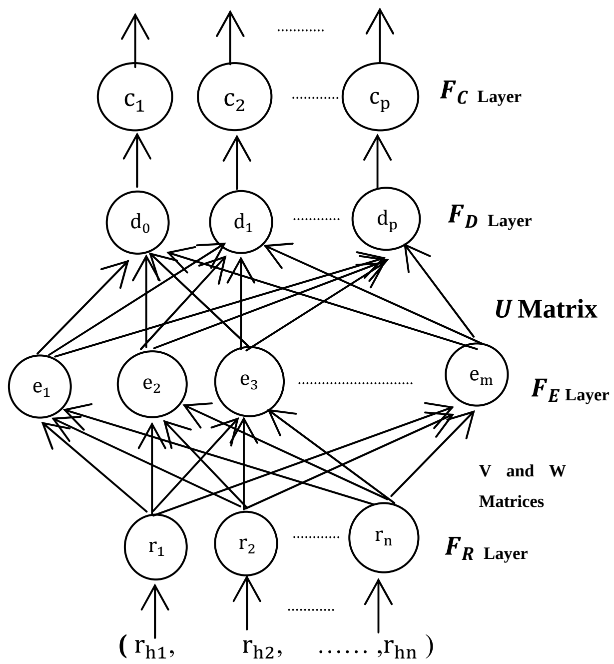

Fig. 1. Fuzzy Hyperline Segment Neural Network

Let $R_h = (r_{h1}, r_{h2}, \ldots\ldots\ldots, r_{hn})$ represents the hth input pattern, $V_j = (v_{j1}, v_{j2}, \ldots\ldots\ldots v_{jn})$ is the one end point of hyperline segment e_j and $W_j = (w_{j1}, w_{j2}, \ldots\ldots\ldots, w_{jn})$ is the other end point of e_j. Then the membership function of the jth F_E node is defined as

$$e_j(R_h, V_j, W_j) = 1 - f(x, \gamma, l) \tag{1}$$

in which $x = l_1 + l_2$, and the distances l_1, l_2 and l are defined as

$$l_1 = \left(\sum_{i=1}^{n}(w_{ji} - r_{hi})^2\right)^{1/2}, \tag{2}$$

$$l_2 = \left(\sum_{i=1}^{n}(v_{ji} - r_{hi})^2\right)^{1/2}, \tag{3}$$

$$l = \left(\sum_{i=1}^{n}(w_{ji} - v_{ji})^2\right)^{1/2}, \tag{4}$$

and f(.) is the three parameter ramp threshold function defined as

$$f(x, \gamma, l) = \begin{cases} 0 & \text{if } x = l, \\ x\gamma & \text{if } 0 \le x\gamma \le 1, \\ 1 & \text{if } x\gamma > 1 \end{cases} \tag{5}$$

The F_D layer gives soft decision and output of k^{th} F_D node represents the degree to which the input pattern belongs to the class d_k. The binary weights assigned to the connections between F_C and F_D layers are stored in the matrix U. The values assigned to these connections are defined as

$$u_{jk} = \begin{cases} 1 & \text{if } e_j \text{ is a hyperline segment of the class } d_k \\ 0 & \text{otherwise} \end{cases}$$

$$\text{for } k = 1, 2, \ldots\ldots p \text{ and } j = 1, 2, \ldots\ldots m. \tag{6}$$

where e_j is the j^{th} F_E node and d_k is the k^{th} F_D node.

The transfer function of each F_D node performs the union of the appropriate (of same class) hyperline segment fuzzy values, which is described as

$$d_k = \max(e_j u_{jk}) \quad \text{for } j = 1 \text{ to } m \quad \text{and} \quad k = 1 \text{ to } p \tag{7}$$

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Each F_C node delivers nonfuzzy output descried as,

$$c_k = \begin{cases} 0 & \text{if } d_k < T \\ 1 & \text{if } d_k = T \end{cases} \text{ where } T = \max(d_k), \text{ for k=1 to p} \quad (8)$$

B. *Learning Algorithm of FHLSNN*

The supervised FHLSNN learning algorithm for creating HLSs in the hyperspace consists of following steps.

Step 1: Initialization. To initialize HLS start with first pattern in the database, as

$$W_j = V_j = R_h. \quad (9)$$

Step2: Creation of hyperline segments. The maximum length of HLS is bounded by the parameter θ, where $0 \leq \theta \leq 1$, which is a user defined value and depends on the dimension of feature vector. The extension criterion that has to be met before HLS can extend to include R_h is

$$e_j\left(R_h, V_j, W_j\right) \geq 0. \quad (10)$$

Let the set of pattern is R, where $R \in \left\{R_h \middle| h = 1, 2, \dots \dots P\right\}$. Given the h^{th} training pair (R_h, d_h), find all the HLSs belonging to the class d_h. After this following cases are carried out for possible inclusion of the input pattern R_h.

Case 1: By using membership function, find out whether the pattern R_h falls on any one of the exiting HLSs. If R_h falls on any of the HLS then it is included. Therefore, in the training process all the remaining steps are skipped and training is continued with the next training pair.

Case 2: If the input pattern R_h falls on any one of the hyperline passing through the two end points of HLS, then extend the HLS to include the pattern. Suppose e_j is that hyperline segment with end points V_j and W_j then l_1, l_2 and l are calculated using equation (2), (3), and (4). Subsequently algorithm executes sub-step (i) if $l_1 > l_2$, else the sub-step (ii). Otherwise the Case 3 is considered.

(i) Test whether the point V_j falls on the HLS formed by the points W_j and R_h using equation (1) and if verified then include the pattern by extending e_j as

$$V_j^{new} = R_h \text{ and } W_j^{new} = W_j \quad (11)$$

(ii) Test whether the point W_j falls on the hyperline segment formed by the points V_j and R_h and if verified, then include the pattern by extending e_j as

$$W_j^{new} = R_h \text{ and } V_j^{new} = V_j. \quad (12)$$

Case 3: If HLS is a point i.e. $W_j = V_j$, then extend it to include the pattern R_h, if extension criteria is satisfied as descried by equation (11).

Case 4: If the pattern R_h is not included by any of the HLSs then create a new HLS as

$$V_j^{new} = W_j^{new} = R_h. \quad (13)$$

Step 3: Intersection test. The learning algorithm allows intersection of HLSs from the same class and eliminates the intersection between HLSs from separate classes. Intersection test is carried out as soon as the HLS is either extended by *Case 2*, *Case 3* or created in *Case 4*.

Let $W_{lst} = [x_1, x_2, \dots \dots . x_n]$, and $V_{lst} = [y_1, y_2, \dots \dots . y_n]$ represent two end points of the extended or created HLS and $W_n = [x'_1, x'_2, \dots \dots . . x'_n], V_n = [y'_1, y'_2, \dots \dots y'_n]$ are the end points of the HLS of other class. The equation of hyperline passing through W_{lst} and V_{lst} is

$$\left[\frac{a_i - x_i}{y_i - x_i}\right] = r_1 \;\; for\; i = 1, 2, \dots \dots \dots, n. \quad (14)$$

and the equation of the hyperline passing through W_n and V_n is

$$\left[\frac{b_i - x'_i}{y'_i - x'_i}\right] = r_2 \;\; for\; i = 1, 2, \dots \dots \dots n. \quad (15)$$

where r_1, r_2 are the constants and a_i, b_i are the variables. The equations (14) and (15) leads to set of *n* simultaneous equations which are described as

$$r_1(y_i - x_i) + x_i = r_2\left(y'_i - x'_i\right) + x'_i \quad (16)$$

for $i = 1, 2, \dots \dots \dots \dots \dots \dots n$.

The values of r_1 and r_2 can be calculated by solving any two simultaneous equations. If remaining *n*-2 equations are satisfied with the calculated values of r_1 and r_2 then two hyperlines are intersecting and the points of intersection P_t is

$$P_t = (r_1(y_1 - x_1) + x_1, \dots \dots, \; r_1(y_n - x_n) + x_n) \quad (17)$$

The point of intersection P_t, if falls on both hyperlines segments then these HLSs are also intersect. This can be verified by the equation (1) and eliminated by contraction of appropriate HLS.

Step 4: Removing intersection. Depending on the cases, if extension of HLS produces an intersection then it is removed by restoring the end point V_j as $V_j^{new} = V_j^{old}$, and point W_j is restored as, $W_j^{new} = W_j^{old}$. Create a new HLS to include R_h as in equation (13).

If *Case 4* creates intersection then it is removed by restoring the end points of previous HLS of other class as

$$W_{new+1} = V_{new+1} = V_n \text{ and } V_n = W_n. \quad (18)$$

III. Modifications to FHLSNN

A. *Prediction based on the Centroid and Euclidean Distance*

After the learning, in the prediction phase, the FHLSNN classifies the applied pattern based on the membership function value calculated using equation (1). When we apply the pattern to the classifier for testing, it calculates the membership value for all the HLSs created during the learning phase. The applied pattern is classified to the class associated with the HLS that gives the highest membership value for this pattern.

We propose the new method to decide the class of applied pattern in the prediction phase of FHLSNN. This method uses centroid of the pattern falling on the hyperline segment and Euclidean distance for the classification of pattern in the prediction phase. In this method, instead of calculating the membership value, the centroid of patterns falling on the each HLS is computed using equation (19), as below.

$$C_{ji} = C'_{ji} + \frac{|R_{hi} - C_{ji}|}{N_j} \quad (19)$$

where, C'_{ji} is the centroid of the j^{th} HLS in the i^{th} dimension, C_{ji} is the centroid of the patterns falling on the j^{th} HLS in the i^{th} dimension, and N_j is the number of patterns falling on the j^{th} HLS.

Then the Euclidean distance [15], between the centroid of patterns falling on the j^{th} HLS in the i^{th} dimension and the applied input pattern is calculated using equation (20),

$$E_{jh} = \sqrt{\sum_{i=1}^{n}\left(C_{ji} - R_{hi}\right)^2} \quad (20)$$

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

where, E_{jh} is the Euclidean distance between the centroid of patterns falling on the j^{th} HLS in the i^{th} dimension and the applied h^{th} input pattern.

Finally, the HLS with the smallest Euclidean distance is selected as winner and the pattern is so classified that it belongs to the class associated with that HLS.

The classification process for a two dimensional input pattern using this first modification is demonstrated in the Fig. 2. This figure shows the two hyperline segments of class 1 and class 2 with centroids of hyperline segment as C'_{12} , C'_{22} and the centroids of the patterns falling on the HLS as C_{12} , C_{22} respectively. The E_1 is the Euclidean distance between the input pattern and centroids C_{12} of first HLS. Similarly, E_2 is the Euclidean distance between the input pattern and centroid C_{22} of second HLS. As the distance E_1 is the smaller than distance E_2, the applied input pattern is classified as class 1. Thus the hyperline segment with smallest distance is selected as winner and the class of that hyperline segment is assigned to the input pattern.

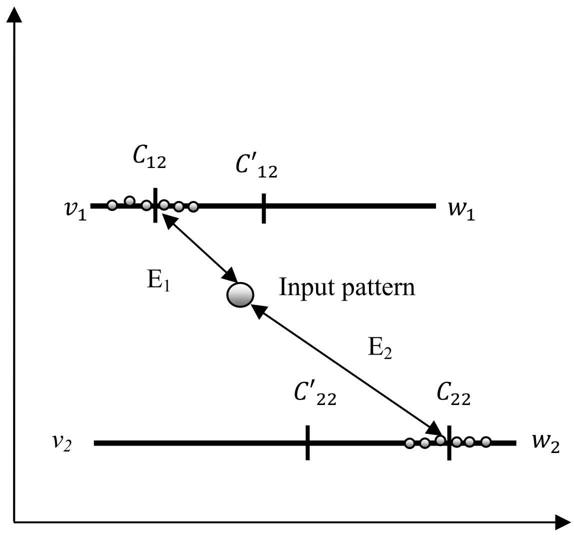

Fig. 2. The classification process of modified FHLSNN

B. Prediction based on Membership Function and Euclidean Distance

In this method, to predict the class of applied input pattern, we propose to use both, the membership value of HLSs for applied input pattern and Euclidean distance between the centroid of patterns falling on the HLS and the applied input pattern. After learning, first the membership values of all the HLSs created during learning are calculated for the applied input pattern. In FHLSNN, the applied pattern is classified to the class associated with the HLS that gives the highest membership value for this pattern. But, in the MFHLSNN2, instead of calculating the Euclidean distance for all the HLSs as like MFHLSNN1, a pool of HLSs that have high membership function is selected. The number of HLSs selected can be based on a user defined threshold. For example, the 30% HLSs having high membership values are selected. After that, the Euclidean distance between the centroid of patterns falling on the HLS and the applied input pattern is calculated using equation (20). Finally, the HLS with the smallest Euclidean distance is selected as winner and the pattern is so classified that it belongs to the class associated with that HLS.

Thus the both proposed modification do not affect the learning process of FHLSNN. Therefore the hyperline segments created during learning are same for both original and modified FHLSNN.

IV. Experiments and Results

A. Benchmark Problem

This proposed modifications are implemented using MATLAB R2013a and ran on Intel core i3 2328M, 2.2GHz PC. To explore the different capabilities of a pattern classifier, we choose three benchmark data sets from the UCI machine learning repository [16] and the real handwritten character database. The three Benchmark data sets are Wine data set, Iris data set and Sonar data set. From the total available patterns, training data set consists of approximately 75% patterns with equal proportion of all class and remaining 25% patterns with equal proportion of all class are used as testing data set. A description of each data set is as follows.

1) The Wine data Set: This data set is another example of multiple classes with higher number of continuous features. This data set contains 178 samples, each with 13 continuous features from three classes. 136 patterns are randomly selected with equal proportion of all classes and given for training. Remaining 42 patterns with equal proportion of all classes are given for testing.

2) The Iris data set: This data set contains 150 samples, each with four continuous features (*sepal length, sepal width, petal length, and petal width*), from three classes (*Iris setosa, Iris versicolor, and Iris virginica*). This data set is an example of a small data set with a small number of features. One class is linearly separable from the other two classes, but the other two classes are not linearly separable from each other. 120 patterns are randomly selected with equal proportion of all classes and given for training. Remaining 30 patterns with equal proportion of all classes are given for testing.

3) The Sonar data set: This is a high-dimensional data set and is useful for evaluating the scalability capability of pattern classifier. It contains 208 samples, each with 60 input features (s_1, s_2, . . .s_{60}). The data set contains 111 patterns of class 1 and 97 patterns of class 2, i.e., sonar signals from mine (metal cylinders) (class 1) and rocks (class 2), respectively. 156 patterns are randomly selected with equal proportion of two classes and given for training. Remaining 52 patterns with equal proportion of two classes are given for testing.

The Table I to Table III shows the percentage recognition rate of FHLSNN, MFHLSNN1 and MFHLSNN2 classifiers for Wine data, Iris data and Sonar data set respectively. The effect of parameter θ on the number of HLS creation in the learning phase and on the recognition rate in the testing is also observed.

The experiments are repeated by varying the parameter θ to determine the number HLSs created and thereafter the recognition rate. It is observed that as we increase the value of θ, the number of hyperline segments created decreases. This also decreases the recognition rate in the testing phase. All the summarized results show that both the proposed modifications in FHLSNN show significant improvement in recognition rate for all the selected data sets. We also observed that the second modification (MFHLSNN2) gives more recognition rate as compare to the FHLSNN and MFHLSNN1. Thus the second modification gives a significant improvement in the recognition rate with less number of HLSs.

TABLE I
RECOGNITION RATE FOR WINE DATA SET

Theta θ	HLS Created	FHLSNN (%)	MFHLSNN1 (%)	MFHLSNN2 (%)
0.045	66	66.66	80.95	80.95
0.05	66	66.66	88.09	90.47
0.1	58	61.90	64.28	66.66
0.15	53	61.90	92.85	92.85
0.2	48	59.52	66.66	66.66

TABLE II
RECOGNITION RATE FOR IRIS DATA SET

Theta θ	HLS Created	FHLSNN (%)	MFHLSNN1 (%)	MFHLSNN2 (%)
0.1	62	96.67	60	100
0.15	59	96.67	63.3	100
0.2	59	96.67	63.3	100
0.25	59	96.67	63.3	100
0.3	59	96.67	63.3	100

TABLE III
RECOGNITION RATE FOR SONAR DATA SET

Theta θ	HLS Created	FHLSNN (%)	MFHLSNN1 (%)	MFHLSNN2 (%)
0.6	133	30.76	34.61	50
0.65	131	30.76	34.61	50
0.70	128	30.76	34.61	50
0.75	124	30.76	36.53	50
0.80	119	30.76	36.53	50

B. Real handwritten character database:

This database consists of 1000 Devanagari numeral character. Ten numerals from one hundred writers are scanned and stored in BMP format. After moment normalization [17], the rotation invariant ring-data features defined by Ueda and Nakamura [18] and extended by Chiu and Tseng [19], are extracted from the character by setting ring width to two.

The extracted ring-data vector is a 16-dimensional feature vector. 800 patterns are randomly selected with equal proportion of ten classes and given for training. Remaining 200 patterns with equal proportion of ten classes are given for testing. The Table IV shows the percentage recognition rate of FHLSNN, MFHLSNN1 and MFHLSNN2 classifiers for handwritten character data set for various values of parameter θ.

TABLE IV
RECOGNITION RATE FOR HANDWRITTEN DATA SET

Theta θ	HLS Created	FHLSNN (%)	MFHLSNN1 (%)	MFHLSNN2 (%)
0.45	403	41	43	42
0.5	402	41	43.5	43.5
0.55	401	41	42	42.5
0.6	400	41	43	43
0.65	400	41	41.5	42

The Table IV shows recognition rate and number of hyperline segments created during training for FHLSNN, MFHLSNN1 and MFHLSNN2. As the modification is proposed in the prediction phase of FHLSNN, the number of hyperline segments created during training remains same for all the three networks. Thus the proposed modification does not affect the incremental learning of FHLSNN. As shown in the table IV, that both the proposed modifications in FHLSNN achieve significant improvement in recognition rate for handwritten character data set also. We also observed that the second modification (MFHLSNN2) gives more recognition rate as compare to the FHLSNN and MFHLSNN1.

C. Example in 2-D space

To explore the recognition ability of MFHLSNN1 and MFHLSNN2, an artificial 2-D dataset is created which is of 28 patterns of two classes. It is worth to mention here that this 2-D data is created for not to fever the proposed modification but to show that in a particular constrained situation, modified FHLSNN can perform well. Table V shows training data set which consist of 20 patterns of class 1 and 2 with its dimensions.

TABLE V
2-D TRAINING DATA SET

Pattern No.	Feature 1	Feature 2	Class Index
1	1	3.1	1
2	1.15	3.565	1
3	1.1	3.41	1
4	1.2	3.72	1
5	1.8	5.58	1
6	1	1.5	1
7	2.9	4.35	1
8	3.11	4.665	1
9	3.15	4.725	1
10	3.2	4.8	1
11	0.2	0.8	2
12	0.25	1	2
13	0.3	1.2	2
14	0.34	1.36	2
15	1.5	6	2
16	0.6	0.6	2
17	1.66	1.66	2
18	1.77	1.77	2
19	1.88	1.88	2
20	2	2	2

Table VI shows the 2-D testing data set of 8 patterns of class 1 and 2 with its dimensions. We intentionally select the dimensions of class 1 patterns so that they fall very closer to the hyperline of class 2 and vice-versa.

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE VI
2-D TESTING DATA SET

Pattern No.	Feature 1	Feature 2	Class Index
1	1.18	4.72	1
2	1.2	4.8	1
3	1.3	5.2	1
4	1.35	5.4	1
5	1	1.5	2
6	1.2	1.8	2
7	1.25	1.875	2
8	1.29	1.935	2

The Table VII shows the performance of FHLSNN, MFHLSNN1 and MFHLSNN2 classifiers in terms of percentage recognition rate for this 2-D data set. We observed that with higher value of theta and less number of hyperline segments, the FHLSNN fails to recognize the patterns, where as both the proposed modifications (MFHLSNN1 and MFHLSNN2) shows 100 % recognition rate. Thus the misclassification of FHLSNN is overcome by adding additional parameter that is the computation of Euclidean distance between the centroid of pattern falling on hyperline segment and applied pattern, in the prediction phase of classifier. Thus the MFHLSNN1 and MFHLSNN2 can perform very well with less number of hyperline segment, means with less complexity.

TABLE VII
RECOGNITION RATE FOR ARTIFICIAL 2-D DATA SET

Theta θ	HLS Created	FHLSNN (%)	MFHSNN1 (%)	MFHSNN2 (%)
0.5	5	25	50	50
0.55	5	25	50	50
0.60	4	0	100	100
0.65	4	0	100	100
0.70	4	0	100	100

V. CONCLUSIONS

The Fuzzy Hyperline Segment Neural Network (FHLSNN) utilizes fuzzy set as pattern classes in which each fuzzy set is a union of fuzzy set hyperline segments. This is supervised classifier that forms *n*-dimensional HLSs defined by two end points, for learning and prediction. In this paper, we have proposed the two modifications in the prediction phase of FHLSNN, which has improved its classification performance for selected benchmark problems and real world handwritten character data set. The proposed modifications are applied in the prediction phase. Thus the incremental learning and other properties of FHLSNN are not affected. All the summarized results show that both the proposed modifications in FHLSNN show significant improvement in recognition rate for all the selected data sets. The second modification gives a significant improvement in the recognition rate with less number of HLSs. The evaluation of proposed classifiers, with 2-D data set explores its capability to classify closely spaced patterns of different classes. Thus the proposed modification improves the prediction accuracy of FHLSNN without affecting its incremental learning. In future, the proposed method can be applied to other fuzzy neural network classifiers with the different data sets.

REFERENCES

[1] J. –S. R. Jang, C. –T. Sun and E. Mizutani, *Neuro-Fuzzy and Soft Computing, A Computational Approach to Learning and Machine Intelligence.* Pearson Prentice Hall: South Asia, 2008.

[2] P. K. Simpson, “Fuzzy min–max neural networks—Part 1: Classification,” *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 776–786, Sep. 1992.

[3] P. K. Simpson, “Fuzzy min–max neural networks—Part 2: Clustering,” *IEEE Trans. Fuzzy Systems,* vol. 1, no. 1, pp. 32–45, Feb. 1993.

[4] B. Gabrys and A. Bargiela, “General fuzzy min-max neural network for clustering and classification,” *IEEE Trans. Neural Networks*, vol.11, pp. 769-783, May 2000.

[5] U. V. Kulkarni, T. R. Sontakke, and G. D. Randale, “Fuzzy hyperline segment neural network for rotation invariant handwritten recognition,” published in *Int. Joint Conf. on Neural Networks: IJCCNN’01* held in Washington DC, USA, July 2001, pp. 2918–2923.

[6] Kwan H. K. and Yaling Cai, “A fuzzy neural network and its applications to pattern recognition,” *IEEE Trans. Fuzzy Systems,* vol. 2, no. 3, pp. 185-192, Aug. 1994.

[7] U. V. Kulkarni, T. R. Sontakke and A. B. Kulkarni, “Fuzzy hyperline segment clustering neural network,” *IEE Electronics Letters* vol. 37, no. 05, pp. 301-303, March 2001.

[8] P. M. Patil, U. V. Kulkarni, and T. R. Sontakke, “General fuzzy hyperline segment neural network,” in *Proc. IEEE Int. Conf. on Systems, Man Cybernetics*, Hammamet, Tunesia, Connection Sci, 2002, vol. 4, pp. 6–27.

[9] H. J. Kim and H. S. Yang, “A weighted fuzzy min-max neural network and its application to feature analysis,” *Advances in Natural Computation (Lecture Notes in Computer Science),* vol.3612. New York: Springer-Verlag, Aug.2005, pp.1178–1181.

[10] A. Rizzi, M. Panella, and F. M. F. Massciloi, “Adaptive resolution min-max classifiers,” *IEEE Trans. Neural Networks,* vol.13, no.2, pp.402–414, Mar.2002.

[11] A. V. Nandedkar and P. K. Biswas, “A fuzzy min-max neural network classifier with compensatory neuron architecture,” *IEEE Trans. Neural Networks,* vol.18, no.1, pp.42–54, Jan.2007.

[12] R. Davtalab, M. Parchami, M. H. Dezfoulian, M. Mansourizade, and B. Akhtar, “M-FMCN: modified fuzzy min-max classifier using compensatory neurons,” in *Proc. 11th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases,* Wisconsin, USA, Feb. 2012, pp. 77-82.

[13] Huaguang Zhang, Jinhai Liu, Dazhong Ma, and Zhanshan Wang, “Data-core-based fuzzy min–max neural network for pattern classification,” *IEEE Trans. Neural Networks*, vol.22, no.12, pp.2339-2352, Dec. 2011.

[14] Anas M. Quteishat and Chee Peng Lim, “A Modified fuzzy min-max neural network and its application to fault classification,” *Soft computing in industrial application, ASC*, vol. 39, pp.179-188, 2007

[15] Johnson, R.A., and D.W. Wichern, *Applied multivariate Statistical Analysis*. New Jersey: Prentice Hall. 1998, pp. 226-235.

[16] P. M. Murphy and D. W. Aha, UCI Repository of Machine Learning Databases, (Machine-Readable Data Repository). Irvine, CA: Dept. Inf. Comput. Sci., Univ. California, 1995.

[17] Perantonis S. J. 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, pp. 241-251, 1992.

[18] Udea K. and Y. Nakamura, “Automatic verification of seal impression pattern,” in *Proc. 9th Int. Conf. on Pattern Recognition.*1984, vo1. 2, pp. 1019-1021.

[19] Hung-Pin Chiu and Din-Chang Tseng, “Invariant handwritten Chinese character recognition using fuzzy min-max neural networks,” *Pattern Recog. Letters*, vol. 18, pp. 481-491, 1997.

ISBN: 978-988-19252-7-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)