Extended Fuzzy Hyperline Segment Neural Network for Fingerprint Recognition

M. H. Kondekar *m_h_kondekar@yahoo.com*
College of Computer Science and Information Technology
Latur, Maharshtra, India.

U. V. Kulkarni *kulkarniuv@yahoo.com*
Professor, Dept. of C.S. & Engg.,
SGGS Institute of Engg. & Tech.
Nanded,Maharastra, India.

B. B. M. Krishna Kanth *bbkkanth@yahoo.com*
Research Scholar,
S.R.T.M. University,
Nanded, Maharashtra, India.

Abstract

In this paper we have proposed Extended Fuzzy Hyperline Segment Neural Network (EFHLSNN) and its learning algorithm which is an extension of Fuzzy Hyperline Segment Neural Network (FHLSNN). The fuzzy set hyperline segment is an n-dimensional hyperline segment defined by two end points with a corresponding extended membership function. The fingerprint feature extraction process is based on FingerCode feature extraction technique. The performance of EFHLSNN is verified using POLY U HRF fingerprint database. The EFHLSNN is found superior compared to FHLSNN in generalization, training and recall time.

Keywords: Biometrics, Fuzzy Neural Network, Hyperline Segment, Fingerprint Recognition, FingerCode

1. INTRODUCTION

The increased emphasis on secrecy and protection of information in databases, personal identification has become very important topic in today's network society. Biometric indicators have an advantage over traditional security identification methods, because these inherent attributes cannot be easily stolen. There are many biometric features that are used for people identification, like iris, face, retina, voice, gait, palm print and fingerprint. The convenience of current electronic applications has led to an explosive increase in their use. E-banking, electronic fund transfer, online shopping and virtual auctions are just some applications prevalently used by the public [1].

Fingerprints are widely used as personal identification technique around the world [2] due to its distinguished features from others. Fingerprints are fully formed at about seven months of fetus development and finger ridge configurations do not change throughout the life of an individual except due to accidents such as bruises and cuts on the fingertips [3]. This property makes fingerprints a very attractive biometric identifier. Fingerprint features are permanent and fingerprints of an individual are unique.

The fingerprint recognition algorithms can be broadly classified into minutiae-based and FilterBank-based algorithms. The minutiae-based matching algorithms first extract the local

minutiae such as ridge endings and ridge bifurcations from the thinning image [4] or the gray scale image, and then match their relative placement in a given fingerprint with the stored template. A number of matching techniques are available in the literature including point-based matching [4] and graph-based matching [5]. Although the minutiae-based matching is widely used in fingerprint verification, but it has problems in efficiently matching two fingerprint images containing different number of unregistered minutiae points. Further, it does not utilize a significant portion of the rich discriminatory information available in the fingerprints.

The FilterBank-based algorithm [6, 7, 8] uses a bank of Gabor filters to capture both local and global information in a fingerprint as a compact fixed-length FingerCode, which is suitable for matching and storage. Thus, it overcomes some of the problems with the minutiae-based matching algorithms. So, here we have used FilterBank-based algorithm Jain et al [8] for efficient and correct fingerprint feature extraction.

In this paper, we have applied EFHLSNN classifier which is an extension of Fuzzy Hyperline Segment Neural Network (FHLSNN) proposed by Kulkarni et al [9] to the problem of fingerprint recognition based on FingerCode feature data. The FHLSNN utilizes fuzzy sets as pattern classes in which each fuzzy set is an union of fuzzy set hyperline segments. The fuzzy set hyperline segment is an n-dimensional hyperline segment defined by two endpoints with a corresponding extended membership function.

The rest of the paper is structured as follows. The feature extraction method in our work is introduced in Section 2. Sections 3 give a brief introduction for the architecture of the EFHLSNN, followed by its learning algorithm in section 4. Section 5 demonstrates the testing results and performance comparison of the classifiers on fingerprint and Iris Fisher data set. Conclusions are made in Section 6.

2. FINGERPRINT FEATURE EXTRACTION

In this paper fingerprint feature extraction is done by using Poly U HRF Fingerprint database images of 320*240 sizes at 1200 dpi resolution. The feature extraction process is based on FilterBank-based FingerCode feature extraction algorithm which consists of following stages.

2.1 Reference Point Location

Fingerprints have many visible landmark structures and a combination of them could be used for establishing a reference point. Jain, Prabhakar, Hong, and Pankanti [8] had defined the reference point of a fingerprint as the point of maximum curvature of the concave ridges in the fingerprint image. The location of reference point is mainly dependent on good quality of image, for graceful handling of local noise in a poor quality fingerprint image; the detection should necessarily consider a large neighborhood in the fingerprint image. For locating a reference point of a fingerprint local ridge orientation is usually specified for a block rather than at every pixel; an image is divided into a set of non overlapping blocks and a single orientation is defined for each block.

2.2 Filtering and FingerCode Feature Extraction

Fingerprints have local parallel ridges and valleys, and well defined local frequency and orientation. Properly tuned Gabor filters [10, 11] can remove noise, preserve the true ridge and valley structures, and provide information contained in a particular orientation in the image. Before filtering the fingerprint image, it is normalized to the region of interest in each sector separately to a constant mean and variance. Normalization is performed to remove the effects of sensor noise and gray level distortion due to finger pressure differences.

An even symmetric Gabor filter has the following general form in the spatial domain:

$$G(x, y; f, \theta) = \exp\left\{\frac{-1}{2}\left[\frac{x'^2}{\delta_{x'}^2} + \frac{y'^2}{\delta_{y'}^2}\right]\right\} \cos(2\pi f x') \quad (1)$$

$$x' = x\sin\theta + y\cos\theta \quad (2)$$

$$y' = x\cos\theta - y\sin\theta \quad (3)$$

Where f is the frequency of the sinusoidal plane wave along the direction θ from the x-axis, and $\delta_{x'}$ and $\delta_{y'}$ are the space constants of the Gaussian envelope along x' and y' axes, respectively. Jain et al [9] had performed the filtering in the spatial domain with a mask size of 33 X 33. In this algorithm they have used eight different values for θ $(0^0,\ 22.5^0,\ 45^0,\ 67.5^0, 90^0,\ 122.5^0,\ 135^0, \text{and } 157.5^0)$ with respect to the x-axis. The normalized region of interest in a fingerprint image is convolved with each of these eight filters to produce a set of eight filtered features. These eight directional-sensitive filters capture most of the global ridge directionality information as well as the local ridge characteristics present in a fingerprint. The mean of each sector in each of the eight filtered features defines the components of FingerCode feature vector. The gray level in a sector in a disk represents the feature value for that sector in the corresponding filtered image.

3. TOPOLOGY OF PROPESED EFHLSNN

The architecture of the EFHLSNN consists of four layers as shown in Figure 1. In this architecture first, second, third and fourth layer is denoted as F_A, F_B, F_D and F_C respectively. The F_A layer accepts an input pattern and consists of n processing elements, one for each dimension of the pattern. The F_B layer consists of m processing nodes that are constructed during training. There are two connections from each F_A to F_B node; one connection represents one end point for that dimension and the other connection represents another end point of that dimension, for a particular hyperline segment as shown in Figure 2.

Each F_B node represents hyperline segment fuzzy set and is characterized by the transfer function. In Let $R_h = (R_{h1}, R_{h2}, R_{h3}, \ldots\ldots, R_{hn})$ represents the h_{th} input pattern, $V_j = (v_{j1}, v_{j2}, \ldots\ldots, v_{jn})$ is one end of the hyperline segment e_j and $W_j = (w_{j1}, w_{j2}, \ldots, w_{jn})$ is the other end point of e_j. Then the membership function of j_{th} F_B node is defined as

$$e_j(R_h, V_j, W_j) = 1 - f(x, y, l) \quad (4)$$

Which $x = l_1 + l_2$ and the distance l_1, l_2 and l are defined as

$$l_1 = \left(\sum_{i=1}^{n} \left| w_{ji} - r_{hi} \right|\right), \quad (5)$$

$$l_2 = \left(\sum_{i=1}^{n} \left| v_{ji} - r_{hi} \right| \right), \tag{6}$$

$$l = \left(\sum_{i=1}^{n} \left| w_{ji} - v_{ji} \right| \right), \tag{7}$$

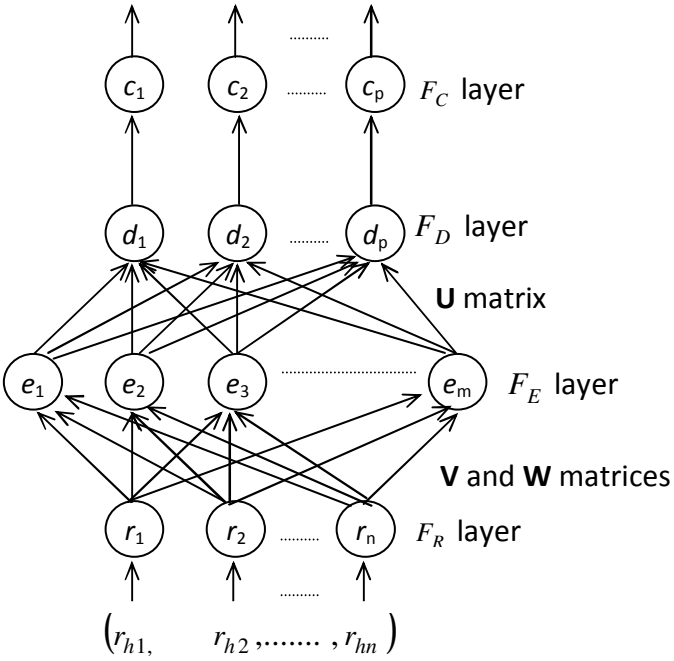

FIGURE 1: Extended Fuzzy hyperline segment neural network.

Here, in this paper we have used Manhattan distance for computing the values of l_1, l_2 and l as shown in equation (5), (6) and (*7).* The Manhattan distance has given best performance in terms of generalization, training and recall time in comparison with Euclidian distance [9] and distance between two position vectors as shown in equation (8), (9) and (10).

$$l_1 = \max \left(\sum_{i=1}^{n} \left| w_{ji} - r_{hi} \right| \right) \tag{8}$$

$$l_1 = \max \left(\sum_{i=1}^{n} \left| v_{ji} - r_{hi} \right| \right) \tag{9}$$

$$l_1 = \max \left(\sum_{i=1}^{n} \left| w_{ji} - v_{ji} \right| \right) \tag{10}$$

$f()$ is three parameter ramp threshold function defined as

$$f(x,\gamma,l)=0, \quad if \quad x = 1 \ otherwise$$

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

The fuzzy hyperline segment membership function for γ = 1, and with end points *w*=[0.5 0.3] and *v*=[0.5 0.7] is shown in Figure 3. This membership function returns highest membership value equal to one if the pattern $R_h$falls on the hyperline segment joined by two end points V_j and W_j. The membership value is governed by the sensitivity parameter γ, which regulates how fast the membership value decreases when the distance between R_h and e_j increases. For the given input pattern R_h, e_j 's output value is computed using equation (4).

Each node of F_C and F_D layer represents a class. 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 weights assigned to the connections between F_E and F_D layers are binary values and stored in matrix *U*, and these values assigned to these connections are defined as

$$u_{jk}=\begin{cases} 1\ if \quad e_j is\ a\ hyperline\ of\ class\ d_k \\ 0 \quad otherwise \end{cases} \qquad (11)$$

for = 1, 2, ..., and =1, 2, ...,m.

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 performs the union of appropriate (of same class) hyperline segment fuzzy values, which is described as

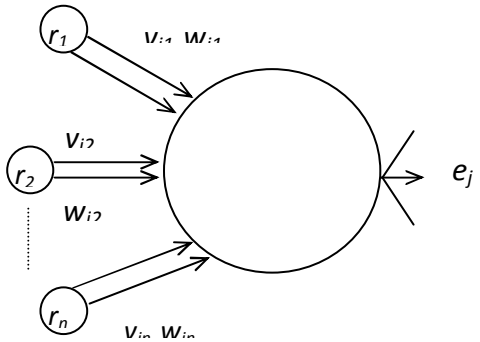

FIGURE 2: Implementation Extended Fuzzy Hyperline Segment

$$n_k = \max_{j=1}^{m} e_j u_{jk} \quad for\ k = 1,2,\ldots\ldots\ldots, p \qquad (12)$$

Each $F_C$node delivers non-fuzzy output, which is described as

$$C_k = \begin{cases} 0 \;\; if \;\; d_j > T \\ 1 \;\; if \;\; d_k == T \end{cases} \quad for\, T = \max(d_k) \quad for \;\; k = 1 \;\; to \;\; p. \tag{13}$$

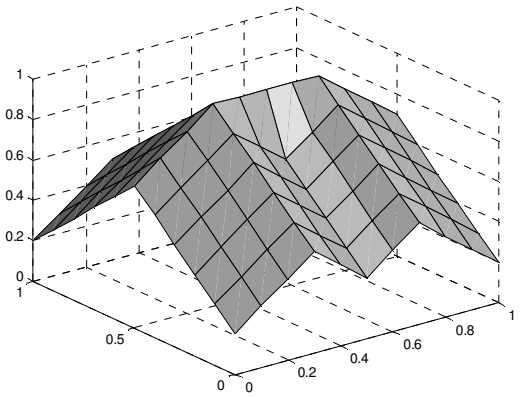

FIGURE 3: Extended Fuzzy Hyperline Segment membership function

4. EFHLSNN LEARNING ALGORITHM

The supervised FHLSNN learning algorithm for creating fuzzy hyperline segments in hyperspace consists of three steps, A: Creation of hyperline segments, B: Intersection test and C: Removing intersection. These steps are described below in detail.

4.1 Creation of Hyperline Segments

The length of hyperline segment is bounded by the parameter ζ, $0 \le \zeta \le \zeta_m$ and ζ_m depends on the dimension of feature vector. In the learning process appropriate values of ζ is selected and hyperline segment is extended only when the length of hyperline segment after extension is less than or equal to ζ. Assuming that the training set defined as $R \in \{R_h \mid h = 1,2,..., P\}$, the learning starts by applying the patterns one by one from the pattern set R. Given the h_{th} training pairs(R_h, d_h), find all the hyperline segments belonging to the class d_h. After this following four sub steps are carried out sequentially for possible inclusion of input patterns R_h.

Step 1: Determine whether the pattern $R_h$falls on any one of the hyperline segments. This can be verified by using fuzzy hyperline segment membership function described in equation (4). If R_h falls on any one of the hyperline segment then it is included, therefore in the training process all the remaining steps are skipped and training is continued with the next training pair.

Step 2: If the pattern R_h falls on any one of the hyperline passing through two end points of the hyperline segment, then extend the hyperline segment to include the pattern. Suppose e_j is that hyperline segment with end points V_j and W_j then $l1$, $l2$ *and* l are calculated using equations (5), (6) and (7). Where $l1$ is the distance of R_h from end pointW_j, $l2$ is the distance of R_h from end point V_j *and* l is the length of the hyperline segment.

2 (a): If $l1 > l2$ then test whether the point V_j falls on the hyperline segkent formed by the points W_j *and* R_h. This condition can be verified using equation (1) i.e. if $e_j(V_j, R_h, W_j) = 1$, then the

hyperline segment is extended by replacing end point V_j by R_h to include R_h, if extension criteria is satisfied. Hence

$$V_j^{new} = R_h \text{ and } W_j^{new} = W_j \quad (14)$$

2 (b): If $l2 > l1$ then test whether the point W_j, falls on the hyperline segment formed by the points V_j and R_h.If $e_j(V_j, R_h, W_j) = 1$, hyperline segment is extended by replacing end point W_j with R_h to include R_h, if extension criteria is satisfied. Hence

$$W_j^{new} = R_h \text{ and } V_j^{new} = V_j \quad (15)$$

Step 3: If hyperline segment is a point then extend it to include the pattern R_h, if extension criteria is satisfied as described by equation (11).

Step 4: If the pattern R_h is not included by any of the above sub-steps then new hyperline segment is created for that class, which is described as

$$W_{new} = R_h \text{ and } V_{new} = R_h \quad (16)$$

4.2 Intersection Test

The learning algorithm allows intersection of hyperline segments from the same class and eliminates the intersection between hyperline segments from separate classes. Intersection test is carried out as soon as the hyperline segment is extended either by sub-step 2 or sub-step 3 or created in sub-step 4.

Let $W_{ist} = [x_1, x_2, \ldots, x_n], V_{ist} = [y_1, y_2, \ldots, y_n]$represent two end points of extended or created hyperline segment and $W_n = [x'_1, x'_2, \ldots, x'_n], V_n = [y'_1, y'_2, \ldots, y'_n]$ are end points of the hyperline segment bf other class. First of all test whether the hyperlines passing through end points of two hyperline segments intersect. This is described by the following equations. The equation of hyperline passing through W_{ist} and V_{ist} is

$$\left[\frac{a_i - x_i}{y_i - x_i}\right] = r_1 \text{ for } i = 1,2, \ldots, n \quad (17)$$

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

$$\left[\frac{b_i - x'_i}{y_i - x'_i}\right] = r_2 \text{ for } i = 1,2, \ldots, n \quad (18)$$

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

$$r_2(y_i - x_i) + x_i = r_2(y'_i - x'_i) + x'_i \text{ for } i = 1,2,\ldots,n \quad (19)$$

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 thentwo hyperlines are intersecting and the point of intersection p_t is

$$P_c = (n_1(y_1 - x_1) + x_1, \ldots\ldots, n_i(y_n - x_n) + x_n) \quad (20)$$

4.3 Removing Intersection

If step 2(a) and step 3 has created intersection of hyperline segments from separate classes then intersection is removed by restoring the end point V_j as $V_j^{new} = V_j$, if sub-step 2(b) has created intersection then intersection is removed by restoring the end point W_j as $W_j^{new} = W_j$, and new hyperline segment is created to include the input pattern A_h, which is described by equation (13).

If the sub-step 4 creates intersection then it is removed by restoring the end points of previous hyperline segment of other class.

$$W_{new+1} = V_{new+1} = V_n \text{ and } V_n = W_n \quad (21)$$

5. SIMULATION RESULTS AND PERFORMANCE COMPARISON

The EFHLSNN is implemented using MATLAB 7.0. The results are obtained and compared with FHLSNN using fingerprint feature data set. The timing analysis of training and recall are depicted in Table 1. Table 2 gives performance comparison using recognition rates along with number of hyperline segments created.

Classifier	Training in seconds	Testing
FHLSNN using Euclidian Distance	0.1648	0.811566
EFHLSNN using Manhattan distance	0.1631	0.743732
EFHLSNN using Distance between two position vectors	0.1806	0.966453

TABLE 1: Timing Analysis with fingerprint data features

Classifier	Recognition Rate	Theta	Hyperline Segments
FHLSNN	100 %	1.4	200
EFHLSNN using Manhattan distance	100 %	0.2	259
EFHLSNN using Distance between two position vectors	100 %	0.2	259

TABLE 2: Percentage Recognition Rate with FHLSNN and EFHLSNN

As shown in Table 1 the training and testing time using EFHLSNN classifier takes less time compared to FHLSNN classifier using Euclidian distance and Distance between two position vectors.

The EFHLSNN classifier is also applied on standard Fisher Iris Database which also takes less time compared to FHLSNN classifier as depicted in Table 3.

Classifier	Training in seconds	Testing
FHLSNN using Euclidian Distance	0.5178	0.811566
EFHLSNN using Manhattan distance	0.4390	0.738011

TABLE 3: Timing Analysis with Fisher Iris dataset

Hence, the EFHLSNN classifier gives better recognition rates in comparison with FHLSNN in terms of less training and recall time along with 100 % recognition rate.

5. CONCLUSION

The EFHLSNN classifier using Manhattan distance has ability to train and recall patterns faster than FHLSNN classifier using Euclidian distance and Distance between two position vectors. Thus it can be used in real time applications for recognition purpose where less training and recall time is the prime demand. Generalization, training and recall time is also verified using Fisher Iris dataset , where almost similar performance is observed.

REFERENCES

[1] P.Meenen, and R.Adhami, "Fingerprinting for security", Proceedings of the IEEE Potentials, pp. 33-38, 2001.

[2] T. Song, Liang Huang, Che-Wei Liu, Jui-Peng Lin, Chien-Ying Li, and Ting-Yi Kuo, "A Novel Scheme for Fingerprint Identification", Proceedings of the Second Canadian Conference on Computer and Robot Vision (CRV'05) 0-7695-2319-6/05.

[3] W. J. Babler, "Embryologic Development of Epidermal Ridges and Their Configuration", Birth Defects Original Article Series, vol. 27, no. 2,1991.

[4] A.K. Jain, L. Hong, S. Pankanti, and R. Bolle, "An identity authentication system using fingerprints", Proceedings of the IEEE, Vol. 85, No. 9, pp. 1365-1388, 1997.

[5] A. K. Hrechak and I. A. Mchugh, "Automated fingerprint recognition using structural matching", Pattern Recognition, vol. 23, no. 8, pp. 893-904, 1990.

[6] A. K. Jain, S. Prabhakar, and L. Hong, "A multichannel approach to Fingerprint Classification", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 4, pp. 348-359, 1999.

[7] K. Jain, S. Prabhakar, L. Hong, and S, Pankanti, "FingerCode: a FilterBank for fingerprint representation and matching", Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition(CVPR), vol. 2, pp. 187- 193, 1999.

[8] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti, "FilterBank-based fingerprint matching", IEEE Transactions on Image Processing, vol. 9, no. 5, pp. 846-859, 2000.

[9] U.V. Kulkarni, T.R. Sontakke and G. D. Randale, "Fuzzy Hyperline Segment Neural Network for Rotation Invariant Handwritten Character Recognition", Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on Neural Network, Washington, DC, USA, 2001, pp. 2918 - 2923 vol.4.

[10] J. G. Daugman, "High confidence recognition of persons by a test of statistical independence", IEEE Transactions on Pattern Analysis and Machine Intelligance, vol. 15, no. 11, pp. 1148-1161, 1993.

[11] J. G. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters", J.Opt. Soc. Amer. A. vol. 2, pp. 1160-1169, 1985.