On Construction of Multi-class Binary Neural Network Using Fuzzy Inter-cluster Overlap for Face Recognition

Neha Bharill, Om Prakash Patel, Aruna Tiwari and Megha Mantri

Abstract In this paper, we propose a Novel Fuzzy-based Constructive Binary Neural Network (NF-CBNN) learning algorithm for multi-class classification. Our method draws a basic idea from Expand and Truncate Learning (ETL), which is a neural network learning algorithm. The proposed method works on the basis of unique core selection, and it guarantees to improve the classification performance by handling overlapping issues among data of various classes by using inter-cluster overlap. To demonstrate the efficacy of NF-CBNN, we tested it on the ORL face data set. The experimental results show that generalization accuracy achieved by NF-CBNN is much higher as compared to the BLTA classifier.

1 Introduction

Neural networks have been successfully applied to problems in pattern classification, function approximation, fault tolerance, medical diagnosis. Usually, real-world problems involve data having multiple classes. Data belongs to multiple classes aims at generating a map from the input data to the corresponding desired output, for a given training set. This kind of mapping is called a classifier. Then, the designed classifier is used to predict class labels of new input instances. One of the most popular neural

N. Bharill (✉) · O. P. Patel · A. Tiwari
Department of Computer Science and Engineering,
Indian Institute of Technology Indore, Indore, India
e-mail: phd12120103@iiti.ac.in

O. P. Patel
e-mail: phd1301201003@iiti.ac.in

A. Tiwari
e-mail: artiwari@iiti.ac.in

M. Mantri
Department of Networth the Finance Club of Indian
Institute of Management Bangalore, Bengaluru, India
e-mail: meghamantri.iiti@gmail.com

M. Tanveer and R. B. Pachori (eds.), *Machine Intelligence and Signal Analysis*,
Advances in Intelligent Systems and Computing 748,
https://doi.org/10.1007/978-981-13-0923-6_56

network learning algorithms is Backpropagation Learning (BPL) algorithm, which requires an extremely high number of iterations to obtain even a simple binary-to-binary mapping. Also, in the BPL algorithm, finding the number of neurons required in the hidden layer to solve a given problem is a major issue. It has been seen that Stone–Weierstrass's theorem does not give the practical guideline in finding the required number of neurons [1]. Gray and Michel [2] reported Boolean-like training algorithm (BLTA). Its learning speed is vastly increased over the BPL. But the disadvantage is the formation of the four-layer network, and too many neurons must be used for the generation of binary-to-binary mapping.

In 1995 [4], ETL is proposed by Kim Park which initiates learning by selecting a core vertex from the available input data and then form a set of separating hyperplanes based on a geometrical analysis of the training inputs. But the selection of core vertex in ETL is not unique in the process of finding separating hyperplanes. Therefore, the number of separating hyperplanes for a given problem can vary depending upon the selection of the core vertex and the order of adding the training data during learning. Also, it works for two classes of problems. In 2003, Yi Xu and Chaudhari [3] proposed mETL. It uses geometric concepts of ETL [4] for classifying the given patterns into multiple categories. But they did not handle the overlapping issues of data belonging to multiple classes. If the data points belong to the multiple classes then overlapping between the data points occurs because the data does not belong to a particular cluster or neuron. It happens because the data sets may contain the noisy data or due to the effect of outliers. The overlap measure quantifies the degree of overlap between the neurons occurred due to the inclusion of data points within the neuron by computing an inter-cluster overlap [5]. Therefore, the inter-cluster overlap is an important issue that needs to be addressed in case of multi-class classification. In 2015, Wang et al. [6] proposed a feedforward kernel neural networks (FKNN) deep architecture for multi-class classification but the architecture is highly complex and computation intensive for multi-class classification and cannot handle overlapping issue.

In this paper, we proposed a Novel Fuzzy-based Constructive Binary Neural Network (NF-CBNN) learning algorithm for learning multi-class classification problems, which form a three-layer network structure and works on the unique core selection strategy. It also guarantees convergence by grouping all the samples for a given core selection. Further, it can handle the inter-cluster overlap between the neurons which occurs while grouping the data points belonging to multiple classes. Moreover, it eliminates the drawback of ETL [4] in which a number of trials with multiple core selections are required to get the convergence. In addition to it, the proposed learning algorithm NF-CBNN trains the network with single core selection which guaranteed convergence by handling learning of samples that are lying in the overlapped regions of multiple classes. Due to the single core selection, NF-CBNN eliminates processing overheads in hidden layer learning.

The remainder of the paper is organized as follows: In Sect. 2, we present the proposed methodology along with the mathematical formulations. In Sect. 3, we apply the proposed technique on face recognition data set. Experimental results demonstrate that the proposed method performs more accurate face recognition by

drastically improving the generalization accuracy and also identifying the variety of faces in ORL face data sets. Finally, concluding remarks are presented in Sect. 4.

2 Proposed Model Description

In this section, the overall concept of Novel Fuzzy-based Constructive Binary Neural Network (NF-CBNN) method is presented. Section A is presented with the general idea of the model. Section B is presented with the preprocessing of images corresponding to faces so that they can be applied as the input to the neural network. In Sect. 1, we present the working of the hidden layer. Finally, section D is presented with the formation of the output layer.

2.1 Overview of the Proposed Model

The Novel Fuzzy-based Constructive Binary Neural Network is proposed which is based on the concept of Expand and Truncate Learning (ETL) [4]. It forms three-layer network structure consisting of the Input layer, Hidden layer, and the Output layer. The proposed approach works for multi-class classification problem, therefore, we need to classify the given set of variables to the multiple classes. Therefore, the input variables are partitioned into multiple groups such as $\{G_1, G_2, \ldots, G_m\}$ based on their outputs. For the multi-class classification problem, all the input variables belong to their respective groups and based on their output will be trained similarly as in case of two-class classification problem. Therefore, at a time the variables present in one group, i.e., G_1 are considered as a true variable concerning the other variables present in the remaining groups, i.e., $\{G_2, \ldots G_m\}$ are considered as false variables.

The proposed approach works by selecting a single core variable for finding all the neurons in the hidden layer because the selection of core affects the number of neurons in the hidden layer. The core variable is selected using the Euclidean distance measure because it works faster than other means of determining correlation. Once, the core is selected then to learn all the variables present in the group G_1, we find the separating hyperplanes, i.e., the hidden layer neurons based on the concept of geometric learning [4]. But, if some of the variables present in group G_1 are still left unlearned within the presently hidden layer neurons, then we compute the fuzzy inter-cluster overlap [5] measure of such variables belonging to group G_1. In this way, we can learn all the remaining variables. If Inter-cluster overlap measure is less than the desired threshold then, that variable is learned within the existing neurons otherwise, the new hyperplane, i.e., new neuron at the hidden layer is required which is generated using single core corresponds to the unlearned variable present in group G_1. Therefore, in this way, all the variables present in group G_1 are learned. Thus, it guarantees convergence by finding the hidden layer neuron corresponding to all the variables present in group G_1 with single core selection.

After training all the variables present in group G_1, these variables are regarded as "don't care" variables concerning the variables present in rest of the groups. All the variables present in group G_2 will be taken as the true variables and rest of the variables present in other groups such as $\{G_3, \ldots, G_m\}$ will be treated as the false variables. Similarly, all the input variables present in group G_2 are trained similarly as of group G_1. During training the input variables of group G_2, it may require more separating hyperplanes, i.e., hidden layer neurons in addition to the number of neurons found corresponding to group G_1 which will always separate the input variables present in group G_2. The same procedure is repeated for training all the groups till last group G_m until all the variables present in their respective groups get trained.

2.2 Preliminaries

Suppose, the set S contains the M input images denoted by $S = \{M_1, M_2, \ldots, M_n\}$ belongs to the multiple classes. Each input image is represented by a $m \times n$ matrix. First, the features are extracted corresponding to each input images. Therefore, the PCA algorithm is used for extracting the features [7]. The feature vectors are generated using the PCA algorithm varies in the range of [0–255]. Once, the feature vectors are extracted using the PCA algorithm, then these feature vectors are quantized by dividing the complete range of available values in three bins which varies in the range of [0–2]. Therefore, the first, second and third bin are given values 0, 1 and 2. The quantization is done to reduce the size of total number of bits of the feature vector. Once, the quantized feature vector is obtained for each image, it is directly converted into the binary form using normal decimal to binary conversion. Once, the quantized feature vector is converted into the binary form, it is fed as the input to the input layer of proposed NF-CBNN approach. Figure 1 describe the flowchart including all the steps involved in the preprocessing phase.

The learning of the network starts with the selection of the core variable. The process of selecting the core variable is discussed as follows: Suppose, the set $S = \{M_1, M_2, \ldots, M_n\}$ contain of all the input variables which belong to the multiple class where each input variable consist of $l\text{-}bits$. Then, input variables which belong to the same class are considered as true variables while other input variables are considered as false variables. Let set $S_1 = \{M_1, \ldots, M_c, \ldots, M_p\}$ represent the set of true input variables taken from set S and one variable M_c is selected from set S_1 as a core variable. The formulations involve in the selection of core variable from set S_1 is discussed next.

Randomly, choose one true input variable M_c from set S_1 and assign $v = M_c$ and then, compute the Euclidean Distance(ED) between the variable v from the remaining M_{p-1} variables present in set S_1, is given as

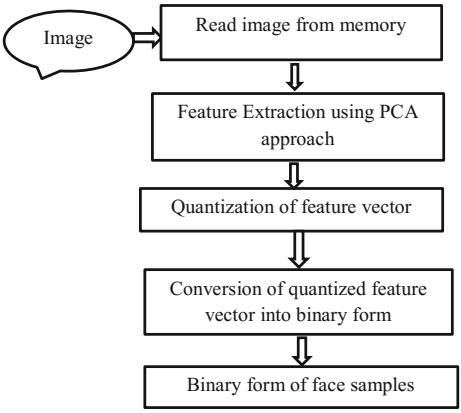

Fig. 1 Flowchart for preprocessing of Image into Binary format as input to Binary Neural Network

$$(ED)_j = \sqrt{\sum_{j=1}^{p-1}(M_j - v)^2} \tag{1}$$

Compute the Membership Degree (MD) corresponding to all the M_{p-1} true variables by taking the fractional distance of each true variable from variable v.

$$(MD)_j = \frac{(\frac{1}{ED_j})^{\frac{1}{m-1}}}{\sum_{j=1}^{p-1}(\frac{1}{ED_j})^{\frac{1}{m-1}}} \tag{2}$$

Check that Membership Degree (MD) obtained in Eq. 2 corresponding each true variable present in set S_1 should be equal to 1 where $j = \{1, 2, \ldots, p-1\}$.

$$MD_j = 1; \forall j \tag{3}$$

Update the value of the core variable using the Membership Degree obtained in Eq. 2.

$$C_v = \frac{\sum_{j=1}^{p-1}[(MD)_j]^2 M_j}{\sum_{j=1}^{p-1}[(MD)_j]^2} \tag{4}$$

Assign the value of the updated core variable to the initial core variable v.

$$v = C_v \tag{5}$$

In each iteration, the value of the core variable is updated using Eq. 4. Thus, the algorithm terminates when change between the two successive iterations in the value of core variable is less than the predefined threshold $\varepsilon = 0.001$ [8]. After finalizing

C_v in this way, the value of core variable is subsequently used in finding the neurons in the hidden layer.

2.3 Hidden Layer Learning

Once, the core variable is found using the above-discussed formulations then we find the separating hyperplanes, i.e., the hidden layer neurons corresponding to each group. Suppose, set S_1 contains all the true variables present in set S, a core variable C_v is found from set S_1 using the above-stated formulations. Then Hamming distance of all the true variables present in set S_1 from a core variable C_v is computed. The number of true variables whose Hamming distance from the core variable is less than d then the generated hyperplane in Eq. 6 will always separate the true variables from the rest of the variables.

$$w_1x_1 + w_2x_2 + \cdots + w_nx_n - T = 0, \tag{6}$$

$$\begin{aligned} w_i &= 1 \qquad if \quad C_v^i = 1 \\ w_i &= -1 \qquad if \quad C_v^i = 0 \end{aligned}$$

$$T = \sum_{p=1}^{n} w_p C_v^k - (d-1)$$

where C_v^i indicates the ith bit of the core variable C_v, T indicates the threshold.

All these true variables which are linearly separable from the rest of the variables are included in the TRUE list. If the true variables are remaining which are not included in the TRUE list, then the hyperplane is expanded so that the remaining true variables can be included in the TRUE list. Therefore, the Hamming distance of all the remaining true variables with the variables present in the TRUE list is computed. The first nearest true variable whose hamming distance from the variables present in the TRUE list is minimum is chosen as a trial variable and included in the TRUE list. After including the trial variable, the hyperplane is expanded using Eq. 7.

$$\begin{aligned} &(2HC_1 - HC_0)x_1 \cdots + (2HC_i - HC_0)x_2 \ldots \\ &\quad + (2HC_n - HC_0)x_n - T = 0 \end{aligned} \tag{7}$$

where, HC_0 is the total number of elements present in the TRUE list including the trial variable, HC_i is calculated as follows:

$$HC_i = \sum_{p=1}^{HC_0} M_p^i \tag{8}$$

If $f_{\min} > f_{\max}$ then there exists a separating hyperplane after including the trial variable in the TRUE list as defined in Eq. 7 where,

$f_{\min} = \min(\sum_{i=1}^{n}(2HC_i - HC_0)M_t)$ among all the variables present in the TRUE list

$f_{\max} = \max(\sum_{i=1}^{n}(2HC_i - HC_0)M_r)$ among all the rest of the variables which are not present in the TRUE list.

The threshold T in Eq. 7 is computed using the equation defined as follows:

$$T = \left\lceil \frac{f_{min} + f_{max}}{2} \right\rceil \tag{9}$$

If $f_{min} \leq f_{max}$, then there does not exist a separating hyperplane. Then, the trial variable is removed from the TRUE list. Similarly, for the other true variables, the procedure is repeated similarly, i.e., another true variable is selected using the same criteria and test is performed to check whether the new trail variable can be included in the TRUE list or not. This procedure is repeated until no more true variables can be included in the TRUE list. However, if all the true variables are not included in the TRUE list, then more than one neuron is required at the hidden layer. Therefore, the other neurons are found by converting the true variables which are not present in the TRUE list into the false variables. Similarly, the false variable which is present in set S but not in set S_1 is converted into true variable. This is the temporary conversion of the desired output of each variable. The conversion is performed to find the separating hyperplanes. After conversion, as soon as all the variables are learned, then the proposed method converges by finding the separating hyperplanes. On the contrary, if some of the false variables are still left unlearned along with the true variables even after conversion. The reason why these variables are still left unlearned because adding them to any particular neuron might lead to the overlapping. Thus, we introduced the concept of inter-cluster overlap [5] based on fuzzy set theory [9], for which the new separating hyperplane corresponding to the unlearned samples can only be found out by membership degree. The discussion and formulation for the same are carried out in the subsequent section.

Further, the training of remaining instances is done on the following basis. Instead of choosing multiple core variables for training all variables as discussed in [4], we train the neural network architecture for the remaining variables (unlearned true and false variables) with the help of fuzzy inter-cluster overlap proposed in [5]. Suppose true variables are still left unlearned, we take those unlearned true variables as a trail variable and compute its membership degree (which determines the belongingness of true variable with the neurons) with an odd number of neurons which are generated earlier. The membership degree of the true variable with the odd number of neurons is computed only after updating the center of odd neurons using Eq. 8. Similarly, for the unlearned false variables, we compute the membership degree with an even

number of neurons only after updating the center of even neurons using Eq. 8. The formulation for computing the membership degree is defined as follows:

$$\mu_i(x_j) = \frac{\|x_j - HC_i\|^{\frac{2}{m-1}}}{\sum_{k=1}^{n} \|x_j - HC_k\|^{\frac{2}{m-1}}}; \forall i, j \tag{10}$$

where x_j represents the jth false or the true variables, HC_i represents the center of ith neurons, m is the weighting component.

Once, the cluster membership degree of all the unlearned variables with there respective neurons are computed then we compute the inter-cluster overlap measure. The formulations used for computing the inter-cluster overlap measure is defined as follows:

$$R(x_j, N_l, N_r) = \begin{cases} \delta(x_j), & if\Big(Dom_{\min}(x_j) > 0 \;\&\; \\ & \quad Dom_{\max}(x_j) < 1\Big) \\ 0.0, & Otherwise \end{cases} \tag{11}$$

where

$$Dom_{\min}(x_j) = \min(\mu_{F_l}(x_j), \mu_{F_r}(x_j)) \tag{12}$$

$$Dom_{\max}(x_j) = \max(\mu_{N_l}(x_j), \mu_{N_r}(x_j)) \tag{13}$$

Conditions:

1. If data point x_j is highly vague, i.e., if maximum degree of membership ($Dom_{\max}$) is less than or equal to 0.5 then, assign degree of overlap equal to 1.0.
   (a) $Dom_{\max}(x_j) \leq 0.5$ then, $\delta(x_j) = 1.0$.
2. Conversely, if the data point x_j is not vague, i.e., if maximum degree of membership ($Dom_{\max}$) is greater than 0.5 and less than 1.0 then, assign degree of overlap between 0.1 to 0.9
   (a) $0.5 < Dom_{\max}(x_j) \leq 1.0$ then, $\delta(x_j) \in [0.1, \ldots 0.9]$.
3. Otherwise, if the data point x_j is clearly classified to particular cluster then, assign degree of overlap equal to 0.
   (a) If $Dom_{\max}(x_j) = 1$ then, $\delta(x_j) = 0$.

where,

$Dom_{\max}(x_j)$ is the maximum degree of membership of trail variable x_j.

$Dom_{\min}(x_j)$ is the minimum degree of membership of trail variable x_j.

The inter-cluster overlap measure computes the overlap due to the inclusion of true or false variable x_j as a trial variable within the respective neurons which is formally represented by $R(x_j, N_l, N_r)$ in Eq. 11. Each trail variable x_j is assigned a degree of overlap based on the belongingness of trail variable to the respective

neuron denoted by $\delta(x_j)$. If the degree of overlap $\delta(x_j)$ achieved by the trial variable is less than 1, then we can learn the trial variable within the existing neuron to which its $\mu_i(x_j)$, i.e., the membership degree of trail variable x_j concerning the neurons N_l and N_r are higher. However, if the degree of overlap $\delta(x_j)$ achieved by the trial variable is equal to the defined threshold 1.0, then the trial variable cannot be included within the existing neurons. This is because the inclusion of trial variable in the existing neuron will result in vagueness then, the new separating hyperplane is found out using Eq. 6. If the learned variable is a false variable then it is added to the FALSE list but, if the learned variable is true variable, then it is added to the TRUE list. Similarly, the process is repeated for training all the unlearned false and true variables. Thus, the algorithm converges by ensuring that all the true and false variables are learned in their respective neurons. The same process works for different groups $\{G_1, G_2, \ldots, G_m\}$ belongs to different classes.

2.4 Formation of Output Layer

Once all the neurons in the hidden layer are found using the single core variable, then another problem in case of multi-class classification problem is that we need to classify the given set of variables to the multiple classes. For this purpose, we required multiple output neuron in the output layer so that the input variables can be classified to the multiple classes. As discussed earlier that the input variables are partitioned into multiple groups such as $\{G_1, G_2, \ldots, G_m\}$ based on their outputs. Since only one output neuron will be fired corresponding to each group. As soon as, all the required neurons in hidden layer are found, the number of hidden layer neuron found corresponding to group G_1 will only be connected to the Ist output neuron. However, the number of hidden layer neuron found corresponding to group G_1 and G_2 will be connected to the IInd output neuron. Similarly, the mth output neuron is connected to the hidden layer neurons of only up to G_m. Therefore, it is inferred that proposed approach requires the less number of neurons in the hidden layer as compared to the multi-class architecture proposed in [3]. However, the weights and threshold of the output neurons are set as follows: The weight of a link from the odd-numbered hidden neuron to the output neuron is set to 1. Similarly, the weight of a link from the even-numbered hidden neuron to the output neuron is set to -1. The threshold of output neuron is set to 1 if the total number of hidden neuron layer generated by the particular group is odd. Otherwise, the threshold of output neuron is set to 0, if the total number of hidden neuron layer generated by the particular group is even. Thus, the proposed approach will always produce the correct output corresponding to the all the input variables. Figure 2 depicts the proposed three layers neural network architecture for the multi-class classification problem.

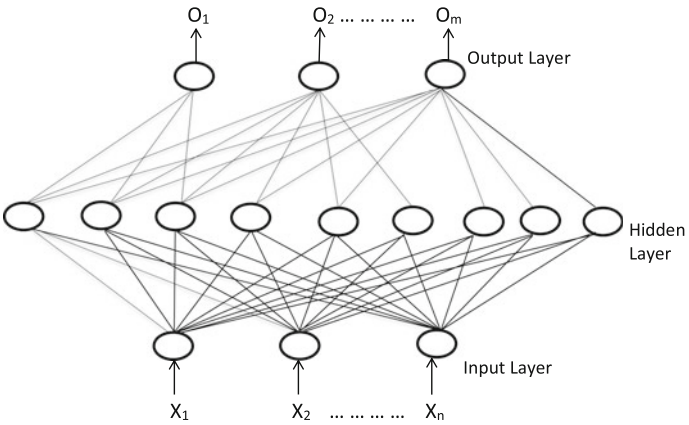

Fig. 2 Neural Network Architecture for the Multi-class Classification

3 Experiments and Analysis

In this section, the proposed NF-CBNN classifier for the multi-class problem is used for face recognition, and its performance is evaluated on the ORL face database. This section is presented with the description of face database, experimental setup, experimental results and discussion.

3.1 ORL Database Description

Experiments were performed on the ORL face database [10–12] as shown in Fig. 3, which consist a set of faces taken by the Olivetti Research Laboratory in Cambridge, U.K.[3] between April 1992 and 1994. The database contains 10 different images of each of the 40 distinct subjects numbered from 1 to 40. For some persons, the images were taken at different times by varying light with different facial expressions (open/closed eyes, smiling/not smiling) and facial details (glasses/no glasses).

3.2 Experimental Setup

The experimentation is carried out on Intel(R) Xeon(R) $E5-1607$ Workstation PC with 64 GB of memory and running with a processing speed of 3.0 GHz on Windows 7 Professional operating system. All codes are written in the MATLAB computing environment. To evaluate the effectiveness of proposed approach, we use stratified tenfold cross-validation method. The images of 40 distinct subjects are divided into ten subsets in which each subset is partitioned into two halves: one

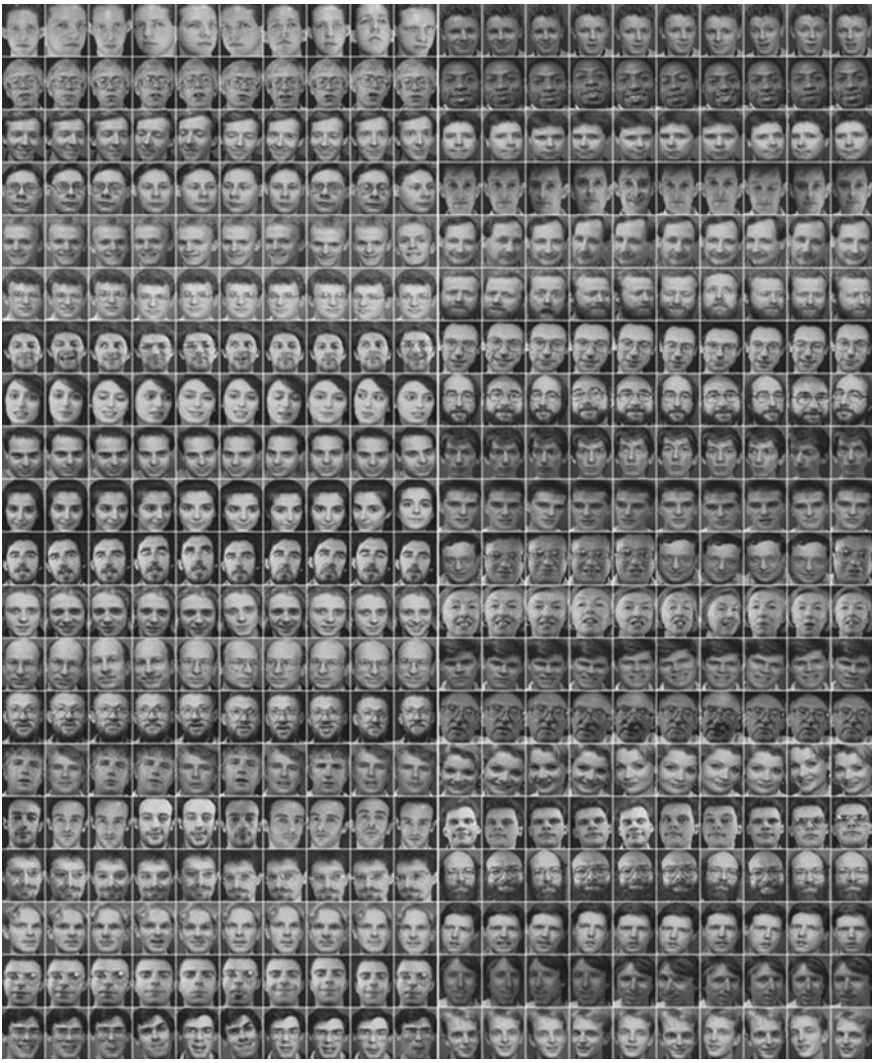

Fig. 3 The ORL face database. There are 10 sample images of each of 40 distinct persons

half is used for training contain 70% images of each subject and the other half is used for testing contain 30% images of each subject. Thus, in all 280 images are used for training and 120 images are used for testing. Then, ten rounds of training and testing runs are conducted repeatedly and independently. Results produced are reported subsequently.

Table 1 Results obtained with NF-CBNN classifier and BLTA classifier on face database. Comparative results are reported across tenfold in terms of three parameters are mean, standard deviations (Std. Dev.), classification accuracy

Set of images	NF-CBNN classifier			BLTA classifier		
	Mean	Std. Dev.	Classification accuracy	Mean	Std. Dev.	Classification accuracy
Image1	0.1000	0.0000	100.0 ± 00.00	0.4666	0.1721	46.66 ± 17.21
Image2	0.1000	0.0000	100.0 ± 00.00	0.1333	0.1721	13.33 ± 17.21
Image3	0.1000	0.0000	100.0 ± 00.00	0.3333	0.2721	33.33 ± 27.21
Image4	0.1000	0.0000	100.0 ± 00.00	0.2333	0.2249	23.33 ± 22.49
Image5	0.1000	0.0000	100.0 ± 00.00	0.1999	0.2811	19.99 ± 28.10
Image6	0.1000	0.0000	100.0 ± 00.00	0.2333	0.2744	23.33 ± 27.44
Image7	0.5666	0.2250	56.66 ± 22.50	0.3333	0.2222	33.33 ± 22.22
Image8	0.1000	0.0000	100.0 ± 00.00	0.1666	0.2357	16.66 ± 23.57
Image9	0.1000	0.0000	100.0 ± 00.00	0.2333	0.2249	23.33 ± 22.49
Image10	0.1000	0.0000	100.0 ± 00.00	0.2999	0.2918	29.99 ± 29.18
Image11	0.1000	0.0000	100.0 ± 00.00	0.2666	0.2629	26.66 ± 26.29
Image12	0.3999	0.3063	39.99 ± 30.63	0.0666	0.1405	06.66 ± 14.05
Image13	0.1000	0.0000	100.0 ± 00.00	0.1999	0.2331	19.99 ± 23.30
Image14	0.5666	0.2250	56.66 ± 22.50	0.1666	0.2357	16.66 ± 23.57
Image15	0.4666	0.2811	46.66 ± 28.11	0.0666	0.1405	14.05 ± 6.660
Image16	0.1000	0.0000	100.0 ± 00.00	0.1666	0.2367	23.57 ± 16.66
Image17	0.1000	0.0000	100.0 ± 00.00	0.1333	0.2331	23.30 ± 13.33
Image18	0.1000	0.0000	100.0 ± 00.00	0.0999	0.1609	16.10 ± 9.990
Image19	0.7666	0.1611	76.66 ± 16.11	0.0333	0.1054	10.54 ± 3.330
Image20	0.1000	0.0000	100.0 ± 00.00	0.2666	0.2108	21.08 ± 26.66
Image21	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image22	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image23	0.1000	0.0000	100.0 ± 00.00	0.0333	0.1054	03.33 ± 10.54
Image24	0.6333	0.3668	63.33 ± 36.68	0.0666	0.2108	06.66 ± 21.08
Image25	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image26	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image27	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image28	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image29	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image30	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image31	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image32	0.1000	0.0000	100.0 ± 00.00	0.1666	0.2357	16.66 ± 23.57
Image33	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image34	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image35	0.6999	0.2918	69.99 ± 29.19	0.0666	0.1405	06.66 ± 14.05
Image36	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image37	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image38	0.0000	0.0000	00.00 ± 00.00	0.0000	0.0000	00.00 ± 00.00
Image39	0.0333	0.1054	03.33 ± 10.54	0.0000	0.0000	00.00 ± 00.00
Image40	0.1000	0.0000	100.0 ± 00.00	0.0999	0.1609	09.99 ± 16.10
Overall	0.1483	0.2148	65.00 ± 33.86	0.1133	0.1228	58.25 ± 23.31

3.3 Experimental Results and Discussion

In this section, experimental results are reported to investigate the effectiveness of proposed NF-CBNN classifier for multi-class face recognition problem in comparison with another classifier works on a multi-class problem known as BTLA [2]. In Table 1 the results of the NF-CBNN classifier in comparison with BLTA classifier across stratified tenfold cross-validation is reported in terms of three parameters; Mean, Standard Deviation, Classification Accuracy. It can be inferred from the Table 1, that the maximum classification accuracy achieved by the proposed NF-CBNN classifier 100 ± 00.00 on images sets are $Image1$-$Image6$, $Image8$-$Image11$, $Image13$, $Image16$-$Image18$, $Image20$, $Image23$, $Image32$ and $Image40$. However, for the other images sets the classification accuracy varies in the range from 00.00 ± 00.00 to 100 ± 00.00. On the contrary, the BLTA classifier achieves 46.66 ± 00.00 as the maximum classification accuracy only on the images set $Image1$. However, it achieves the minimum classification accuracy as 00.00 ± 00.00 on images sets are $Image21$-$Image22$, $Images25$-$Image31$, $Image33$-$Image34$ and $Image36$-$Image39$ whereas on the other sets of images the classification accuracy varies in the range from 00.00 ± 00.00 to 46.66 ± 00.00. Thus, we can observe that the classification accuracy achieved by the proposed NF-CBNN classifier is comparatively much higher than the classification accuracy achieved by the BLTA classifier on stratified tenfold cross-validation. Hence, the above observation leads to the realization that NF-CBNN classifier outperforms over the BLTA classifier.

4 Conclusion

In this paper, we proposed a Novel Fuzzy-based Constructive Binary Neural Network (NF-CBNN) learning algorithm for solving multi-class classification problem. The proposed approach works on the basic idea taken from the Expand and Truncate Learning (ETL) [4]. Thus, it guarantees convergence with unique core selection. The proposed approach converges by learning all the samples including the samples which are confusing in a sense belonging to more than one classes. This issue is handled by using the inter-cluster overlap measure [5]. Thus, it ensures improvement in the classification accuracy. To validate the efficacy of the proposed NF-CBNN classifier, we perform the experimentation on face data set [10] using 10 fold cross-validation approach. The performance of NF-CBNN classifier is judged in comparison with the BLTA classifier [2] using same data set. It is observed that classification accuracy achieved by NF-CBNN classifier is improved greatly as compared to BLTA classifier because the proposed approach handles the confusing samples (samples with belongs to various classes) using the fuzzy inter-cluster overlap measure [5]. Also, NF-CBNN classifier achieves higher accuracy with 3 layer networks structure in comparison with BLTA classifier which forms four-layer network structure. From the results reported

in Table 1, it is obvious that BLTA classifier is unable to classify many images sets properly and thus it achieves very poor classification accuracy. However, the proposed NF-CBNN classifier can classify more image sets. Reported results are verifying that the identification of images with our method is very high in comparison with BLTA classifier and achieves significant improvement in the classification accuracy. In addition to that, our method claims that if more variations of the same image samples have been taken, then there is an improvement of membership degree. Thus, it helps in improving the learning of unclassified samples more appropriately.

References

1. Cotter, N.E.: The stone-weierstrass theorem and its application to neural networks. IEEE Trans. Neural Netw./a Publ. IEEE Neural Netw. Counc. **1**(4), 290–295 (1989)
2. Gray, D.L., Michel, A.N.: A training algorithm for binary feedforward neural networks. IEEE Trans. Neural Netw. **3**(2), 176–194 (1992)
3. Xu, Y., Chaudhari, N.: Application of binary neural networks for classification. In: 2003 International Conference on Machine Learning and Cybernetics, pp. 1343–1348. IEEE (2003)
4. Kim, J.H., Park, S.K.: The geometrical learning of binary neural networks. IEEE Trans. Neural Netw. **6**(1), 237–247 (1995)
5. Bharill, N., Tiwari, A.: Enhanced cluster validity index for the evaluation of optimal number of clusters for fuzzy c-means algorithm. In: IEEE World Congress on Computational Intelligence. International Conference on Fuzzy Systems(FUZZ-IEEE), pp. 1526–1533. IEEE, Beijing, China (2014)
6. Wang, S., Jiang, Y., Chung, F.L., Qian, P.: Feedforward kernel neural networks, generalized least learning machine, and its deep learning with application to image classification. Appl. Soft Comput. **37**, 125–141 (2015)
7. Pentland, A.: Looking at people: sensing for ubiquitous and wearable computing. IEEE Trans. Pattern Anal. Mach. Intell. **22**(1), 107–119 (2000)
8. Setnes, M., Babuska, R.: Fuzzy relational classifier trained by fuzzy clustering. IEEE Trans. Syst. Man Cybern. Part B: Cybern. **29**(5), 619–625 (1999)
9. Kim, D.W., Lee, K.H., Lee, D.: On cluster validity index for estimation of the optimal number of fuzzy clusters. Pattern Recogn. **37**(10), 2009–2025 (2004)
10. Pigeon, S., Vandendorpe, L.: The m2vts multimodal face database (release 1.00). In: Audio-and Video-Based Biometric Person Authentication, pp. 403–409. springer (1997)
11. Angadi, S.A., Kagawade, V.C.: A robust face recognition approach through symbolic modeling of polar FFT features. Pattern Recogn. **71**, 235–248 (2017)
12. Li, H., Suen, C.Y.: Robust face recognition based on dynamic rank representation. Pattern Recogn. **60**, 13–24 (2016)