Turkish Journal of Computer and Mathematics Education Vol.12 No. 7 (2021), 2425-2436

Research Article

Nature Inspired Classifier Based on Binary Neural Network and Fuzzy Ant Colony Optimization Algorithm

[1]Sachin Bhandari, [2]Dharmendra Dangi, [3]Dr. Amit Bhagat, [4]Dheeraj Kumar Dixit

[1]Computer Engineering
Mpstme, NMIMS
Shirpur, India
Sachin.bhandari@nmims.eu
[2]Mathematics and Computer Application
MANIT Bhopal, India
dangi28dharmendra06@gmail.com
[3]Mathematics and Computer Application
MANIT Bhopal, India
am.bhagat@gmail.com
[4]Mathematics and Computer Application
MANIT Bhopal India dheeraj.dixit26@gmail.com

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 16 April 2021

***Abstract*—** Since last decade, classification methods are useful in a wide range of applications. Classification is a task to group the sample having similar properties. This capability can be introduced in computer system by designing various types of classifiers. In artificial intelligent there is a technique called neural network which has so many approaches to applying for this problem. Here, Nature Inspired Classifier Based on Binary Neural Network and fuzzy ant colony optimization (NBNNFA) is design and implemented for the application of multi class. Generally data mining model is used to classify unseen data. That's why so many efforts have taken to improve the performance. The multiclass classification performance improvement will be the main goal of this research, which is achieved by using fuzzy ant colony optimization (FACO). FACO able to control the continuous data type, and can remove information uncertainty to perform better than the traditional algorithm. FACO uses a rule-based feature selection concept to minimize the least influent attributes due to which computational time increases. **So the preprocessed input binary data first given to FACO than** it forms three layered network architecture i.e. Input, Hidden and Output layer. Firstly, it preprocesses data set for the sake of generating binary values. Then, preprocessed data is used for hidden layer as an input, to minimize the training time for all multiple classes the hidden layer training is done in parallel and using the concept of geometrical learning. This approach is tested with various benchmark datasets. The results corresponding to these datasets are generated by varying the learning parameters.

***Keywords*—**Supervised Learning; classification; binary neural network; geometrical expansion; fuzzy; ant colny optimization.

I. INTRODUCTION

Machine learning is a technique to modify the system models that are based on artificial intelligence. Like Robot control, prediction, planning, diagnosis, pattern recognition etc. Machine learning integrates knowledge to make a system capable for autonomous acquisition[1]. It means system capable to learn through observation and analysis by which system can improves its own efficiency and performance. Some important applications of machine learning are data mining, supervised learning etc. Data mining techniques are the emerging idea which can be easily apply to existing hardware and software platforms to integrate with new systems and products to work online. This is the need of today's world as the data grows rapidly, so every system needs a computational engine which is capable enough to work in a multiprocessor environment and cost effective also. Data mining applications are based on effective learning of artificial intelligence and neural network [2].

There will be a fundamental issue against neural network i.e. how many hidden layers should exist in a neural network? So whenever a comparison find between one and two hidden layer networks which contain linear threshold units can identified a class of function having two hidden layer network function by using two hidden layer network [13]. Binary to Binary mapping problem can be resolved by using Back propagation algorithm (BPL). In BPL algorithm hidden layer neurons are mandatory to solve the problem, so that input and output vector dimensions are used to measure the number of neurons which is important for the input and output layer of the system [11]. Since last two decades, single hidden layer feed forward neural network (SLFNs) is an advance research work, in which there are few new findings in SLFNs i.e. Additive hidden nodes(AHN) and Radial basis function(RBF) are comes in to picture for pattern recognition [16].

The neural network stability will be estimated by error, i.e. minimum the error better will be the stability and higher the error the stability will be measured as worst. The excessive hidden neuron will causes over fitting. This problem of over fitting is critical in prediction [17]. Hidden layer is the intermediate unit between input and output data. It is used to measure the weighted sum of input. The result of output layer will be dependent on hidden layer [15]. There are various types of Applications of neural network like Datamining, Visualization and email spam filtering, pattern recognition etc, which can be prepared by learning method of ANN called supervised learning, unsupervised learning and reinforcement learning [14]. It is fact that most of the real world applications having element of uncertainty which effects the overall performance of the system. Fuzzy is the technique which can resolve the problem of information uncertainty by handling the data imprecision. Ant colony optimization algorithm can make fuzzy more optimized and powerful technique to remove uncertainty and increase the performance of the classifier [12].

Here a well-known optimization technique named Ant colony optimization (ACO) which is define for NP hard combinatorial problems of optimization, firstly it is tested for travelling salesman problem[12]. In this proposed work the ACO is merging with Fuzzy and named as Fuzzy ant colony optimization algorithm (FACO) which is used to generate the binary input for the system which is the clusters train as neurons in the systems, which can be generated through three processes first probabilistic neuron selection, second pheromone updating and third Termination conditions. After that it forms three layered network architecture [1], input layer, hidden layer and output layer. For hidden layer formation, the NBNNFA gives highest degree of parallelism. Hidden layer modules of NBNNFA are the patterns of only one class. In this proposed work for classifier parameter optimization an iterative learning method process is performed. In this classification system the problem of overlapping is handled to increase the performance of the classifier. This is done by variations in hyper sphere radius. The output of hidden layer is combined at the output layer.

II. RELATED WORK

Hasija try to proposed a recommender system which is used to provide recommendations on the basis of ratings given by user to a movie or TV serial. These ratings are based on some information like mood, time, day type and other factors. Author have focused on the scalability problem of recommender system which can be resolved by using ANN, but because of high dimensionality feature the ANN takes more time for training, so selection of feature subset is used to solve this problem. This combinatorial optimization problem will be solved by using ACO with fuzzy heuristic problem. Here author have used back propagation algorithm for training neural network [3].

Sonule at al. proposed a rule-extractor by using a combination of three powerful techniques of soft computing i.e. fuzzy, NN and ACO for decision making. They have imposing a light on the drawback of traditional system. Author says that, in account of consistency rule accuracy was not maintained by the earlier method of traditional systems. In proposed system they are making a rule book which is useful for decision making. The numbers of rules in proposed work are decrease in count and increase in rank[8].

Valdez et al. design a modular neural network for pattern recognition using ACO. By using the combination of ACO and modular neural network, they prepare a model to achieve image identification in less time[9].

Raghtate et al. proposed A hybrid intelligent system is proposed by author using the soft computing approaches i.e. Fuzzy c-means, ant colony optimization and neural network to achieve high classification rate from a brain MRI dataset. Here they have used fuzzy inference system for image enhancement combination of FCM with ACO for segmentation of images and to classify images in to two classes i.e. normal and abnormal. Further they have used Feed Forward NN for classification [7].

Lin tries To reduce the non linear uncertainties in a six phase copper rotor induction motor drive system, author proposed a back stepping control system which is based on popular techniques called fuzzy, NN and ACO. Author have used ACO along with recurrent neural network to reduce chattering in control effort [5].

Li et al. are focused on the problem of multiple-target prediction. The sytem is divided in to three important phases, in first phase they have used complex fuzzy sets to achieve multiple target, than they make a hybrid machine learning method by using multiswarm continuous ant colony optimization (MCACO) for updating the parameters in second phase and in third phase recursive least square estimation is used to make a machine learning model[4].

Zhang et al. proposed an improved ant colony optimization (IACO) with ant colony optimization (ACO) to design Type-2 TSK fuzzy logic system. For feasibility verification of the proposed approach they have predict the price of international petroleum and the zhongyan environmental protection share prices[10].

Mavrovouniotis idea behind the proposed approach is to achieve global optimization using hybridization of Back propagation and ant colony optimization. The back propagation get trapped in to local optimum and will not able to achieve global optimum, this drawback of BP will overcome by using Ant olony optimization. Author have proposed an hybrid framework in which ACO training will obtained some good initial weights values, and then apply it to BP training for improvement[6].

III. MODULAR INFORMATION OF THE SYSTEM

This block diagram provides information about an organization of the system. This model tells the working of the whole system, in which input data may be labeled and unlabelled or both. This will be evaluated in different domains of iris, glass, Riply and wine from the field like marketing, etc. As shown in Fig1.

The data which is used for input may be labelled and unlabelled both for training purpose. Pre-processing of data is done to remove inconsistency and to give more accurate result and efficient pre-processing. The created neurons in hidden layer are mapped to output layer for the classes.

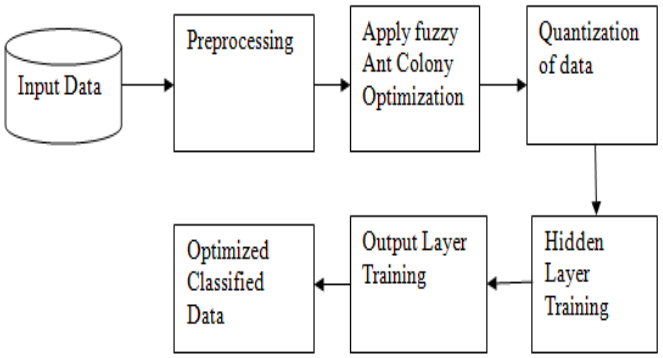

Fig 1. Overview of NBNNFA

IV. BASIC CONCEPTS

4. Initialization and execution of system

In the proposed system firstly the preprocessing will be done by min_max algorithm, after that the data apply on fuzzy ant colony optimization for make the data optimized. Once the data get optimized it will provide for quantization to convert it in to binary for the proposed system. By creating a hyper sphere all hidden layer N modules will initialize and pattern related to the class of the module'. 'In this stage the network having K modules, each module stores one hyper sphere whose radius is zero and a center initialized with the arrangement of the classes'. The expansion of hyper sphere possible when required and the purpose of clustering is to find the hyper spheres.

4.1 Min-Max Normalization Algorithm: Preprocessing

Preprocessing is done by min_max algorithm. It normalize the actual data by performing linear transformation.

Here min_a is the minimum valueand max_a is the maximum value for attribute A.

This min_max algorithm maps a value r of A to r' with range [new_min_a , new_max_a] by using following steps:

Step 1: Read the data file in a comma separated format and save it in a matrix X.

Step 2: Then, each sample, $x^k = (x_1^k, x_2^k, \ldots\ldots x_n^k)$ of matrix $X(i, j)$.

$$new\_x_i^k = \frac{x_i^k - \min(x_i)}{\max(x_i) - \min(x_i)}$$

Step 3: Save the transformed values in a new matrix and write it in a file. Then read the file for quantization.

4.2 Algorithm for performance enhancement

Fuzzy calculate distance between all the samples in which concept of centroid is used.

Step 1: The distance between two data samples Hi, Hj which is represented by three variables qij,rij, sij.

hereqij= rij(1- δ_1) lies in $0<=\delta_1<=1$,
$s_{ij} = r_{ij}(1+ \delta_2)$ lies in $\delta_2>=0$.

i.e. q_{ij} is lower value
rij is average value
sij is higher value
$\delta_1$and δ_2 are fuzziness factors.

Step 2: This is based on three conditions

a) $\delta_1 = 1$ then qij= 0 which means distance Hi and Hj might be zero and congruent.
b) $\delta_1 = 0$ then qij= rijstates that fuzzificationnot present in lower direction and distance between Hi and Hj might stable in lower values and not changed.
c) $\delta_2 = 0$ then sij =rij which means distance between Hi and Hj not effects on upper side so the distance is same on highest values.

Pheromone updating for fuzzy length of a visit of samples through ant k is given as follows

$$\Delta_{\tau_{ij}^k = \frac{Q}{(L^k, M^k, U^k)}} \quad (1)$$

$$\Delta_{\tau_{ij}^k = \frac{Q}{(L^k, M^k, U^k)}} \quad (2)$$

Pheromone intensity on the node *(i,j)* is given by

$$\tau_{ij} = \left(\tau_{ij}L, \tau_{ij}M, \tau_{ij}U\right) \quad (3)$$

Evaporation of pheromone is given by

$$\tau_{ij} = (1-\rho)\tau_{ij} + \Delta\tau_{ij} \quad (4)$$

Therefore

$$\boldsymbol{P_{ij}} = \frac{(\boldsymbol{\tau_{ij}L^{\alpha}, \tau_{ij}M^{\alpha}, \tau_{ij}U^{\alpha}}) * \left(\left(\frac{1}{u_{ij}}\right)^{\boldsymbol{\beta}}, \left(\frac{1}{d_{ij}}\right)^{\boldsymbol{\beta}}, \left(\frac{1}{l_{ij}}\right)^{\boldsymbol{\beta}}\right)}{\boldsymbol{\Sigma_k}(\boldsymbol{\tau_{ij}L^{\alpha}, \tau_{ij}M^{\alpha}, \tau_{ij}U^{\alpha}}) * \left(\left(\frac{1}{u_{ij}}\right)^{\boldsymbol{\beta}}, \left(\frac{1}{d_{ij}}\right)^{\boldsymbol{\beta}}, \left(\frac{1}{l_{ij}}\right)^{\boldsymbol{\beta}}\right)} \quad (5)$$

That's obtain,

$$P_{ij} = \frac{\left(\tau_{ij}L^{\alpha} * \left(\frac{1}{u_{ij}}\right)^{\beta}, \tau_{ij}M^{\alpha} * \left(\frac{1}{d_{ij}}\right)^{\beta}, \tau_{ij}U^{\alpha} * \left(\frac{1}{l_{ij}}\right)^{\beta}\right)}{\left(\Sigma_k\tau_{ij}L^{\alpha} * \left(\frac{1}{u_{ij}}\right)^{\beta}, \Sigma_k\tau_{ij}M^{\alpha} * \left(\frac{1}{d_{ij}}\right)^{\beta}, \Sigma_k\tau_{ij}U^{\alpha} * \left(\frac{1}{l_{ij}}\right)^{\beta}\right)} \quad (6)$$

Thus

$$P_{ij} = \left(\frac{\tau_{ij}L^{\alpha} * \left(\frac{1}{u_{ij}}\right)^{\beta}}{\Sigma_k\tau_{ij}U^{\alpha} * \left(\frac{1}{l_{ij}}\right)^{\beta}}, \frac{\tau_{ij}M^{\alpha} * \left(\frac{1}{d_{ij}}\right)^{\beta}}{\Sigma_k\tau_{ij}M^{\alpha} * \left(\frac{1}{d_{ij}}\right)^{\beta}}, \frac{\tau_{ij}U^{\alpha} * \left(\frac{1}{l_{ij}}\right)^{\beta}}{\Sigma_k\tau_{ij}L^{\alpha} * \left(\frac{1}{u_{ij}}\right)^{\beta}}\right) \quad (7)$$

Assume: $P_{ij} = (P_l, P_m, P_u)$

$$P_l = \frac{\tau_{ij}L^{\alpha} * \left(\frac{1}{u_{ij}}\right)^{\beta}}{\Sigma_k\tau_{ij}U^{\alpha} * \left(\frac{1}{l_{ij}}\right)^{\beta}} \quad (8)$$

$$P_m = \frac{\tau_{ij} M^{\alpha} * \left(\frac{1}{d_{ij}}\right)^{\beta}}{\Sigma_k \tau_{ij} M^{\alpha} * \left(\frac{1}{d_{ij}}\right)^{\beta}} \quad (9)$$

$$P_u = \frac{\tau_{ij} U^{\alpha} * \left(\frac{1}{l_{ij}}\right)^{\beta}}{\Sigma_k \tau_{ij} L^{\alpha} * \left(\frac{1}{u_{ij}}\right)^{\beta}} \quad (10)$$

Centroid method is useful for calculating the ranking probalities. Centroid is calculated as

$$P = \frac{P_l + P_m + P_u}{3}$$

4.3 Quantization Algorithm

After normalization, samples will be given as an input to quantization algorithm to get data in binary format. This is done by using steps below:

Step 1: Read the normalized file and save it in a matrix X.

Step 2: For each value of the matrix X repeat the step 3.

Step 3: Each value $X(i, j)$ quantization will be done by matlab function described below:

Y = uencode (u, n, v)

Step 4: Save the quantized value in a new matrix and write it in new file.

Once the data quantized, its mean data have been converted into binary format this pre-processed data will considered as input for training.

4.4 *Training Phase*

Input:

1) Training dataset X = binary dataset

Output:

1) Iterations
2) Training time
3) Accuracy

Step 1. Initially core vertex selected

Step 2. Measure the difference of bit between rest of the vertex and core vertex

Step 3. Find final vertex which is having shortest distance

Step 4. SITV contain Core vertex and rest of vertex are put in to rest

Step 5. Compute distance between rest vertex and SITV

Step 6. Select trial vertex whose output is 1 from rest whose distance from core vertex is less

then other vertex in rest

Step 7. Calculate hyperplanes based on the given equation.

Here

$(w_1x_1 + w_2x_2 + \ldots\ldots + w_nx_n - T) = 0$

If

$V_c^i = 1$ then $Wi > 0$ else $Wi < 0$

Where

$W_i = 1$, if $F(V_i) = 1$ and $V_c^i = 1$,

$W_i = -1$, if $F(V_i) = 1$ and $V_c^i = 0$,

$W_I = 1$, if $F(V_i) = 1$ and $V_c^i = 1$,

$W_i = -1$, if $F(V_i) = 1$ and $V_c^i = 0$,

Step 8. Find separating hyperplane which can be represented as

$(2c_1 - c_0)\, x_1 + (2c_2 - c_0)x_2 + \ldots\ldots\ldots + (2c_n - c_0)\, x_n - T = 0$

Where,

T = constant called threshold. That is, if there exists a separating hyperplane

$\sum_{i=1}^{n}(2C_i - C_o)\, v_t^i - T <= 0$ for each true vertex'V_t' in SITV,

$\sum_{i=1}^{n}(2C_i - C_o)\, v_r^i - T <= 0$ for each rest vertex'V_r' not included in SITV

Step 9. Calculate tmin and fmax

T = tmin + fmax / 2

Step 10. If tmin > fmax

Goto Step 6 until all true vertices are covered.

Step 11. If rest contain vertex whose output is 1 then convert true in to false & false into true

Goto step 6.

Step 12. To calculate the center of the class region initialize the matrix and find the center of the region by given formulation

$$C_j = \frac{\sum_{i=1}^{N}\left[u_j(x_i)\right]^m x_i}{\sum_{i=1}^{N}\left[u_j(x_i)\right]^m}$$

Step 13. Initialize U= [uij] matrix, U (0)

Step14. Updated cluster membership matrix.

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{||x_i - c_j||}{||x_i - c_k||}\right)^{\frac{2}{m-1}}}$$

Step 15. At kth-step: calculate the centers vectors C (k) = [cj] with U (k)

Step 16. Update U (k), U (k+1)

Step 17. find intra cluster compactness in the feature space is the clustering objective function

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{C}\left(u_j(x_i)\right)^{m}\left||x_i - c_j|\right|^{2}, 1 \le m < \infty$$

Step 18. If $U^{(k+1)} - U^{(k)} < Є$ then STOP; otherwise return to step 15.

4.5 Output Layer Training

Step 1: For every occurred hyper sphere.
Step 2: find the number of class and number of output.
No. of output neuron=round up (log_2 (k)) where k represent number of class
Step 3: map the class of output (o/p) with the output (o/p) neuron.

4.6 *Testing Phase*
Input:

1) Weight matrix, Thresholds of NBNNFA (learned during training phase).
2) Semilabelled data

Output:

1) Clusters of Semilabelled data.

BEGIN

Step 1: for each testing samples i=1 to m1 and for j=1 to q equal to the number of attribute in the training sample, Feed all of the semilabelled data in the neural network trained by NBNNFA.

Step 2: Find the count and mapping of the labelled data in classes formed by step 1.

Step 3. Calculate the majority output of labelled data present in each class by applying probabilistic method.

$$p(\omega_l / x_i) = \sum_{k=1}^{k} p(c_k / x_i)\, p(\omega_l / c_k)$$

// *where $p(\omega_l / x_i)$ is posterior probability can be modeled with the cross product of $p(c_k/\mathrm{x}_i)$ represents the posterior probabilites of the presence of corresponding sample also known as cluster membership degree of the test sample and $p(\omega_l / c_k)$ denotes the cluster posterior probabilites of class membership which is obtained in the training phase of the classifier in terms of Hidden layer, output determines the degree at which the cluster K belongs to a class L.*//

Step 5. The output class labels $f(\mathrm{x}_i)$ coresponding to every test sample is detemined by defuzzifying the posterior probability using maximum operator.

Step 4. Assign the majority output calculated in step 3 to all the unlabelled data present in the class. And form a cluster based on the output of data.

V. EXPERIMENT AND RESULTS

Various benchmarks datasets are used for experimentation as discussed in 5.1, few datasets have been taken from UCI repository [18] with different domain like Iris, Spambase, Wine, and Balance scale etc.

These parameters used in the construction of NBNNFA system are C, cc and λ which play a major role to train a classifier. The core vertex cc and the value of cluster center C is decided at the time of training and the value of scale factor λ is decided at the time of testing.

The different values of these parameters were taken to carry out experiments, as given under:

1. Parameter C: The value of number of cluster center C is in the range from number of classes up to C_{max}. Here the parameter C_{max} is set to $\sqrt{N}$ where N is the number of the training samples.
2. Parameter cc: The value of core vertex is chosen at the time of training, it can be changed means we can take any sample as a core as per requirement.
3. Parameter λ: The value of scale factor λ is taken {0.01, 0.05, 0.1, 0.5, 1, 5, 10, and 15}. To achieve higher accuracy lowest value of scale factor λ is chosen. Here, the value of λ is 0.01 is taken for all the experimentation of datasets.

Here, Iris dataset is considered as a semi-supervised data, in which some samples are labelled and rest of the samples are unlabelled. Iris data having 4 feature dimensions, for quantization we use 2 levels, 4 levels, 8 levels, 16 level data for each feature. The required bit for quantization is as follows

2 levels	1 bit per feature
4 levels	2 bit per feature
8 levels	3 bit per feature
16 levels	4 bit per feature

The number of bits required in data for 4 features are
2 level → 1 * 4 = 4 bits
4 level → 2 * 4 = 8 bits
8 level → 3 * 4 = 12 bits
16 level → 4 * 4 = 16 bits

5.1 Experimentation Performed on IRIS dataset

Here Iris data set is considered as semi-labelled datasets. By using iris we make some datasets of 4 bit, 8 bit, 12 bit, and 16 bit and perform training and testing on these datasets.

TABLE 1. DETAILS OF DATASETS USED FOR EXPERIMENTATION

Dataset	Number of Instances	Number of Classes	Number of Features	Number of Instances in training set	Number of Instances in testing set
4 bit	150	3	4	75	75
8 bit	150	3	8	80	70
12 bit	150	3	12	80	70
16 bit	150	3	16	75	75

5.2 Experimentation Performed on 4 bit dataset

Classifier performance is evaluated by testing the classifier on the test samples. The result for this dataset is described in Table 2. Results have been calculated for various training parameters; according to values of parameter the performance of the NBNNFA system is changing.

TABLE 2.RESULTS OF NBNNFA CLASSIFIER FOR 4 BIT DATASET

C (No. of Clusters)	Iterations	Training time in seconds	Classification Accuracy
3	33	0.275633	92%
4	28	0.177357	66.66%
5	46	0.327219	73.58%
9	48	0.462996	66.66%

5.3 Experimentation Performed on 8 bit dataset

The result for 8 bit dataset dataset is described in Table 3 given below. Results have been calculated for various training parameters, according to values of parameter the performance of the NBNNFA system is changing.

TABLE 3.RESULTS OF NBNNFA CLASSIFIER FOR 8 BIT DATASET

C (No. of Clusters)	Iterations	Training time in seconds	Classification Accuracy
2	62	16.09078	64.74%
5	119	52.90411	68.04%
14	248	214.6095	69.48%
18	257	258.7856	72.22%

5.4 Experimentation Performed on 12 bit dataset

12 bit Dataset is a dataset with 50% of dataset used for training and rest 50% of the dataset is used for testing purpose. The result for 12 bit dataset is described in Table 4 given below. Results have calculated for various training parameters, according to values of parameter performance of the NBNNFA system is changing.

TABLE 4.RESULTS OF NBNNFA CLASSIFIER FOR 12 BIT DATASET

C (No. of Clusters)	Iterations	Training time in seconds	Classification Accuracy
2	54	1.317097	89.09%
4	76	3.49803	90.86%
6	160	9.548971	89.79%

5.5 Experimentation Performed on 16 bit dataset

The outcome is based on various training parameters - core vertex (cc), cluster center C and scale factor. Results have been calculated for various training parameters. It is observed that according to values of parameters, performance of the NBNNFA system is changing.

TABLE 5.RESULTS OF NBNNFA CLASSIFIER FOR 16 BIT DATASET

C (No. of Clusters)	Iterations	Training time in seconds	Classification Accuracy
3	56	0.69163	68.54%
4	293	2.528757	71.91%
5	84	0.88263	69.66%
6	137	1.594787	71.91%
8	139	1.748581	71.91%

It is observed that the number of cluster centre C will be decided based on the parameters as described earlier. The value of cluster center C is selected carefully because the appropriate value of C lies within that range, on which the classifier achieved the highest accuracy. The accuracy achieved with the proposed classification system is an acceptable accuracy, in the range of 60-90 % (depending on the number of labeled data in semilabelled dataset). Therefore the proposed algorithm can be used for clustering the semilabelled data set of real world and results can be used for classification.

VI. CONCLUSION

A binary neural network learning algorithm for multiclass classification is studied, analyzed and implemented. This is nature inspired classifier because firstly the data sample is optimized by fuzzy ant colony optimization, by which the problem of overlapping in hidden layer of match region and claim region is reduce to negligible. This Classification works on the principle of geometrical expansion & evaluates number of hyper spheres. A hyper sphere represents a neuron and the approach states the three layered 'binary neural network architecture' is trained. In NBNNFA the 'hidden layer' of each module will belongs to only one class pattern. To reduce the learning time, hidden layer parallel trained for all classes. It works for handling the overlapping problem for all the classes of samples. So that it will be increase the accuracy and classify both labeled and unlabelled data using this classifier. It is tested with various datasets. Accuracies of various datasets depends on the learning parameters$\sigma 1$, $\sigma 2$ and $\delta 1$, $\delta 1$. This classifier is used to classify any type of data whether it is labeled or unlabelled or both.

REFERENCES

[1] Narendra S. Chaudhari and Aruna Tiwari ,"Binary Neural Network Classifier and it's Bound for the Number of Hidden Layer Neurons", ICARCV2010 978-1-4244-7815-6/10/$26.002010 IEEE

[2] D.Wang and N.S.Choudhari, "A novel training algorithm for Boolean neural networks based on multi level geometrical expansion," Neurocomputing 57C (2004) 455-461

[3] Hasija, H. (2017). An effective approach of feature selection for recommender systems using fuzzy C means cluster-ing along with ant colony optimization and neural networks. *8th International Conference on Computing, Commu-nications and Networking Technologies, ICCCNT2017*.https://doi.org/10.1109/ICCCNT.2017.8203914

[4] Li, C. (2019). Complex Neural Fuzzy Prediction Using Multi-Swarm Continuous Ant Colony Optimization. *Current Trends in Computer Sciences & Applications*, *1*(3),68–79.https://doi.org/10.32474/ctcsa.2019.01.000115

[5] Lin, C. H. (2019). Backstepping control and revamped recurrent fuzzy neural network with mended ant colony opti-mization applied in SCRIM drive system. *Journal of Intelligent and Fuzzy Systems*, *36*(4), 3447–3459. https://doi.or g/10.3233/JIFS-181201

[6] Mavrovouniotis, M. (2013). *Evolving Neural Networks using Ant Colony Optimization with Pheromone Trail Limits . Cci*, 16–23.

[7] Raghtate, G. S., & Salankar, S. S. (2016). Automatic Brain MRI Classification Using Modified Ant Colony System and Neural Network Classifier. *Proceedings - 2015 International Conference on Computational Intelligence and Communication Networks, CICN 2015*, 1241–1246. https://doi.org/10.1109/CICN.2015.239

[8] Sonule, P. M., & Shetty, B. S. (2017). An enhanced fuzzy min–max neural network with ant colony optimization based-rule-extractor for decision making. *Neurocomputing*, *239*, 204–213. https://doi.org/ 10.1016/j. neucom. 2017.02.017

[9] Valdez, F., Castillo, O., & Melin, P. (2016). Ant colony optimization for the design of Modular Neural Networks in pattern recognition. *Proceedings of the International Joint Conference on Neural Networks*, *2016-Octob*, 163–168. https://doi.org/10.1109/IJCNN.2016.7727194

[10] Zhang, Z., Wang, T., Chen, Y., & Lan, J. (2018). Design and application of Type-2 fuzzy logic system based on improved ant colony algorithm. *Transactions of the Institute of Measurement and Control*, *40*(16), 4444–4454. https://doi.org/10.1177/0142331217750221

[11] Sachin Bhandari And Dr. Aruna Tiwari, "Design and Implementation of Binary Neural Network Learning with Fuzzy Clustering," *Advance Computer Science and Information Technolog**y,*** CS & IT 06, pp. 349–356, 2012**.** DOI : 10.5121/csit.2012.2334

[12] Alsawy, A. A., Hefny, H. A., & Licy, F. El. (2014). *Fuzzy Ant Colony Optimization Algorithm*. *November 2010*. https://doi.org/10.1109/ICCTD.2010.5645952

[13] Nakama, T. (2012). Systematic comparisons of single- and multiple-hidden-layer neural networks. *Neurocomputing: Learning, Architectures and Modeling*, *1*, 147–162.

[14] Panchal, F. S., & Panchal, M. (2014). International Journal of Computer Science and Mobile Computing Review on Methods of Selecting Number of Hidden Nodes in Artificial Neural Network. *International Journal of Computer Science and Mobile Computing*, *3*(11), 455–464.

[15] Pitambare, D. P. (2016). *Survey on Optimization of Number of Hidden Layers in Neural Networks*. *5*(11), 537–540. https://doi.org/10.17148/IJARCCE.2016.511115

[16] Seifollahi, S., Yearwood, J., & Ofoghi, B. (2012). Novel weighting in single hidden layer feedforward neural networks for data classification. *Computers and Mathematics with Applications*,*64*(2),128–136. https://doi.org/10.1016/ j.camwa .2012.01.042

[17] Sheela, K. G., & Deepa, S. N. (2013). Review on methods to fix number of hidden neurons in neural networks. *Mathematical Problems in Engineering*, *2013*. https://doi.org/10.1155 /2013/425740

[18] C.Blake, E. Keogh, and C. J. Merz, UCI Repository of Machine learning Databases, Dept. Inf. Computer science, Univ. California Irvine, Irvine, CA, 1998