The CSI Journal on
Computer Science and Engineering
Vol. 5, No. 2&4 (b), 2007
Pages 1-8
Regular Paper

Fuzzy Swarm Net (FSN) for Classification in Data Mining

Bijan Bihari Misra[1] Satchidananda Dehuri[2] Ganapati Panda[3] Pradipta Kishore Dash[4]

[1]Department of Computer Science, College of Engineering Bhubaneswar, Bhubaneswar-751024, India
[2]Department of Information and Communication Technology, Fakir Mohan University, Vyasa Vihar, Balasore-756019, India
[3]Department of Electronics and Telecommunications, National Institute of Technology, Rourkela, India
[4]Department of Electrical Engineering, College of Engineering Bhubaneswar, Bhubaneswar-751024, India

Abstract

In this paper, we have used a fuzzy net trained by the particle swarm optimization (PSO) technique for classification task of data mining. Fuzzy net uses a single layer neural network to reduce the complexities associated with multi-layer perceptron (MLP). In addition, we trained the network by using PSO to obtain optimal weight vectors, which reduces the problem of being trapped into a local minimum. From the simulation study, it is observed that the fuzzy swarm net (FSN) gives a promising result. Further, we compared the result obtained from FSN with multi-layer perceptron (MLP) radial basis function (RBF). The results obtained from FSN draws a clear edge between FSN and RBF and a competitive result with MLP.

Keywords: Fuzzy Net, Multi-layer Perceptron, Particle Swarm Optimization, Classification, Data Mining, Radial Basis Neural Network.

1. Introduction

The classification task of data mining and knowledge discovery in databases [1, 2] from a huge amount of data in real world databases has received much attention in recent years and is growing very fast. In addition to classification task [3], there are many important tasks are exist like association rule mining [4], clustering [5], dependency modelling etc. in the area of data mining. However classification is a fundamental activity in pattern recognition [3, 6, 7], data mining and so forth. Given predetermined disjoint target classes $\{C_1, C_2, ..., C_n\}$, a set of input features $\{F_1, F_2, .., F_m\}$ and a set of training data T with each instance taking the form $<a_1, a_2, ..., a_m>$, where a_i $<i=1, 2, ..., m>$ is in the domain of attribute F_i, i=1, 2, …, m and associated with a unique target class label the task is to build a model that can be used to predict the target category for new unseen data given its input attributes values.

There are many classifiers like Bayesian [8, 9, 10], linear discriminant [11], k-nearest neighbour [12, 13], kernel, multi-layer perceptron [14] and its variant, decision tree [15, 16], and many more exist in the literature. But linear classifiers are of special interest, due to their simplicity and easy expansibility to non-linear classifiers. One of the most powerful classical methods of linear classifiers is the least mean squared error procedure with the Ho and Kashyap modification [17]. Two main disadvantages of this approach are: (i) the use of the quadratic loss function, which leads to a non-robust method, (ii) the impossibility of minimizing the Vapnik-Chervonenkis (VC) [18, 19] dimension of the designed classifier.

The most important feature of the classifier is its generalization ability, which refers to producing a reasonable decision for data previously unseen during the process of classifier design (training). The easiest way to measure the generalization ability is to use a test set that contains data that do not belong to the training set. From statistical

learning theory, we know that in order to achieve good generalization capability, we should select a classifier with the smallest VC dimension (complexity) and the smallest misclassification error on the training set. This principle is called the principle of Structural Risk Minimization (SRM) [19].

In real life, noise and outliers may corrupt samples nominated for the training set. Hence the design of the classifier methods needs to be robust. According to Huber [20], a robust method should have the following properties: (i) reasonably good accuracy at the assumed model, (ii) small deviations from the model assumptions should impair the performance only by a small amount, (iii) larger deviations from the model assumptions should not cause a catastrophe. There are many robust loss functions are discussed in [20]. In this work, due to its simplicity, the absolute error loss function is taken.

The paper by Bellman et al. [21] was the starting point in the application of fuzzy set theory to pattern classification. Since then, researchers have found several ways to apply this theory to generalize the existing pattern classification methods, as well as to develop new algorithms. There are two main categories of fuzzy classifiers: fuzzy if-then rule-based and non if-then rule fuzzy classifiers. The second group may be divided into fuzzy k-nearest neighbours and generalized nearest prototype classifiers (GNPC). Several approaches have been proposed for automatically generating fuzzy if-then rules and tuning parameters of membership functions for numerical data. These methods fall into three categories: neural-network-based methods with high learning abilities, genetic (evolution)-based methods with the Michigan and Pittsburg approaches, and clustering-based methods. There are several methods that combine the above enumerated categories that have proved effective in improving classification performance [22]. Recently, a new direction in the fuzzy classifier design field has emerged: a combination of multiple classifiers using fuzzy sets [25], which may be included into the non if-then fuzzy classifier category. In general, there are two types of the combination: classifier selection and classifier fusion. In the first approach each classifier is an expert in some local area of the feature space. In the second approach all classifiers are trained over the whole feature space. Thus, in this case, we have competition, rather than complementing, among the fuzzy classifiers. Various methods have been proposed for fuzzy classifier design; however, in contrast to statistical and neural pattern classifiers, both theoretical and experimental studies concerning fuzzy classifiers do not deal with the analysis of the influence of the classifier complexity on the generalization error.

In this paper, we have combined the best attributes of fuzzy theory [22] and particle swarm optimization [23, 24] techniques to obtain an optimal fuzzy swarm net. The fuzzy swarm net contains a single layer perceptron. The input data is converted to the fuzzy membership functions. These membership functions are multiplied with the weights and fed to the perceptron. A set of such nets is considered as a swarm and the mean square error (MSE) of each net is considered for training the nets using PSO.

The rest of the paper is organized as follows. In Section 2 the fuzzy net architecture is briefly discussed. Section 3 discusses the basics of particle swarm optimization. Section 4 describes the fuzzy swarm net (FSN) classifier design. Section 5 covers the results obtained during experimental studies. We have concluded the paper in Section 6.

2. Fuzzy Net Architecture

The fuzzy net is basically a flat net and the need of the hidden layer is removed and hence, the learning algorithm used in this network becomes very simple. The architecture of this artificial neural network model uses a single perceptron. The input vector is expanded into three different membership functions. The expansion input vectors effectively increases the dimensionality of the input vector and hence the hyper planes generated by the fuzzy net provides greater discrimination capability in the input pattern space. Figure 1 shows an example of configuration of the fuzzy net [26].

Each unit in the input layer has a triangular Gaussian membership function as its inner function, given by

$$O_{i,1}(x_i, sm_i, th_i) = e^{-\frac{1}{2}\left(\frac{x_i - sm_i}{th_i}\right)^2}, \quad (1)$$

$$O_{i,2}(x_i, me_i, th_i) = e^{-\frac{1}{2}\left(\frac{x_i - me_i}{th_i}\right)^2}, \quad (2)$$

$$O_{i,3}(x_i, bg_i, th_i) = e^{-\frac{1}{2}\left(\frac{x_i - bg_i}{th_i}\right)^2}, \quad (3)$$

where x_i is the i[th] input feature, sm_i is the low value of the i[th] input feature, bg_i is the high value of the i[th] input feature, me_i is the medium value of the i[th] input feature and th_i is taken as $(bg_i - sm_i)/3$.

Each input data is the input of these membership functions (low, medium, high) and the outputs of the units O_{ij} (i = 1, - n, j = 1,2,3) are the grades of the membership functions. The inputs of the unit in the output layer are $w_{ij}O_{ij}$. The output unit has a sigmoid function f given by

$$o = f(s) = \frac{1}{1 + e^{-s}}, \quad (4)$$

where s is the sum of the inputs of the output unit i.e.

$$s = \sum_{i=1}^{N}\sum_{j=1}^{3} O_{i,j} W_{i,j}. \quad (5)$$

o is the output of this network. The connection weights w_{ij} are modified by the δ rule.

3. Particle Swarm Optimization

The particle swarm algorithm is an optimization technique inspired by the metaphor of social interaction observed among insects or animals. The kind of social interaction modeled within a PSO is used to guide a population of individuals (called particles) moving towards the most promising area of the search space. In a PSO algorithm, each particle is a candidate solution equivalent to a point in a d-dimensional space, so the i^{th} particle can be represented as

$x_i = (x_{i1}, x_{i2} \ldots x_{id})$. Each particle "flies" through the search space, depending on two important factors, $p_i = (p_{i1}, p_{i2} \ldots p_{id})$, the best position found so far by the current particle has and $p_g = (p_{g1}, p_{g2} \ldots p_{gd})$, the global best position identified from the entire population (or within a neighborhood).

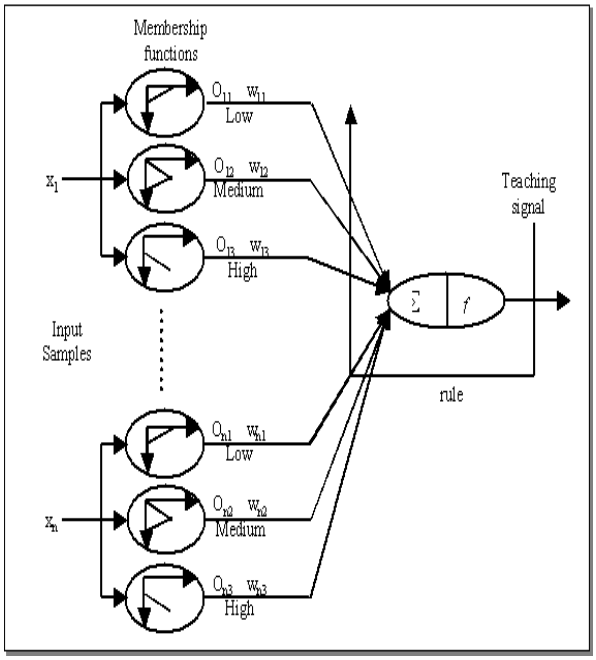

Figure 1. A simple model of fuzzy net

The rate of position change of the i^{th} particle is given by its velocity $v_i = (v_{i1}, v_{i2} \ldots v_{id})$. Equation (6) updates the velocity for each particle in the next iteration step, whereas equation (7) updates each particle's position in the search space [23]

$$v_{id}(t) = \tau(v_{id}(t-1) + \phi_1(p_{id} - x_{id}(t-1)) + \phi_2(p_{gd} - x_{id}(t-1))) \quad (6)$$

$$x_{id}(t) = x_{id}(t-1) + v_{id}(t) \quad (7)$$

where

$$\tau = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}, \ \phi = \phi_1 + \phi_2, \phi > 4.0, \quad (8)$$

τ is referred to as the constriction coefficient.

Two common approaches of choosing p_g are known as *gbest* and *lbest* methods. In the *gbest* approach, the position of each particle in the search space is influenced by the best-fit particle in the entire population; whereas the *lbest* approach only allows each particle to be influenced by a fitter particle chosen from its neighborhood. Kennedy and Mendes studied PSOs with various population topologies [8], and have shown that certain population structures could give superior performance over certain optimization functions.

Further, the role of the *inertia weight* ϕ, in Equation (8), is considered critical for the PSO's convergence behaviour. Improved performance can be achieved through the application of an inertia weight applied to the previous velocity

$$v_{id}(t) = \phi \cdot v_{id}(t-1) + \phi_1(p_{id} - x_{id}(t-1)) + \phi_2(p_{gd} - x_{id}(t-1)). \quad (9)$$

The inertia weight is employed to control the impact of the previous history of velocities on the current one. Accordingly, the parameter ϕ regulates the trade-off between the global (wide-ranging) and local (nearby) exploration abilities of the swarm. A large inertia weight facilitates global exploration (searching new areas), while a small one tends to facilitate local exploration, i.e., fine-tuning the current search area. A suitable value for the inertia weight ϕ usually provides balance between global and local exploration abilities and consequently results in a reduction of the number of iterations required to locate the optimum solution. Initially, the inertia weight was constant. However, experimental results indicated that it is better to initially set the inertia to a large value, in order to promote global exploration of the search space, and gradually decrease it to get more refined solutions. Thus, an initial value around 1.2 and a gradual decline towards 0 can be considered as a good choice for ϕ.

4. Fuzzy Swarm Net Classifier

To implement the fuzzy net with swarm intelligence, initially we take a set of fuzzy nets. Each net is treated as a particle and the set of fuzzy nets are treated as swarm. Each net shares the same memory in a distributed environment. At any instance of time all the nets are supplied with one input record and the respective target. All the nets in the distributed environment are initialized to random weights w_{ij} in the range [0,1].

Let us take that the input–output data are given as

$(X_i, y_i) = (x_{1i}, x_{2i}, \ldots\ldots, x_{ni}, y_i)$,

where i = 1, 2, 3, ….., N

The input–output relationship of the above data can be described in the following manner:

$y = f(x_i, x_2, \ldots\ldots, x_N)$.

The estimated output O produced by the nets in the distributed environment can be represented as

$$O = f(x_1, x_2, \ldots\ldots, x_N) = \frac{1}{1 + e^{-s}}$$

where

$$s = \sum_{i=1}^{N} \sum_{j=1}^{3} O_{i,j} W_{i,j}$$

The error is estimated as $err = (Y - O)^2$

Fuzzy Swarm Net architecture has been shown in Figure 2.

We calculate the error for all the nets in the distributed environment after each iteration. The net giving minimum error is treated as the leader or gbest among all the nets. The nets also preserve the best value achieved by the respective nets during all the iterations in their local memory, which is treated as the personal best or pbest of it. Each nets uses both gbest and pbest values to update its weights for the next iteration as follows.

$$v_{kd}(t) = \tau(v_{kd}(t-1) + \phi_1(p_{kd} - x_{kd}(t-1)) + \phi_2(p_{gd} - x_{kd}(t-1)),$$

$$x_{kd}(t) = x_{kd}(t-1) + v_{kd}(t),$$

where k is the swarm size, $d=i*j$, w_{ij} obtains its value from x_{kd}.

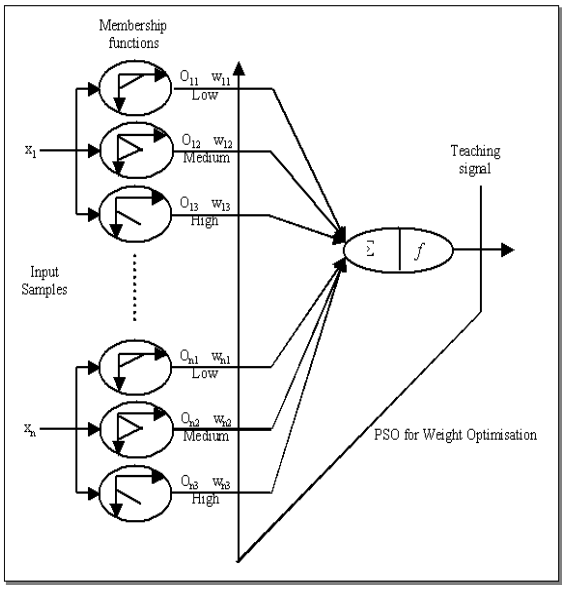

Figure 2. Architecture of fuzzy swarm net

The stopping criterion may be allowing the nets to iterate till they converge to a single decision. However in this process, the nets get over trained leading to poor performance of the classifier. From different simulations of a dataset, a suitable range of iteration can be fixed for it. Different dataset require different range of iterations, one range may not be suitable for all the datasets. The following high-level pseudocode gives more insight view of the proposed model.

Pseudocode

1) Determine system's input variables
2) Form training and testing data
3) Initialise the weights to the nets in the swarm
4) Calculate the output of the nets and determine their error.
5) Update gbest and pbest if required.
6) If stopping criterion not met go to step 4.

5. Simulations and Result

In this Section the performance of the FSN model is evaluated using the real world benchmark classification databases. The most frequently used in the area of neural networks and of neuro-fuzzy systems are Wisconsin Breast Cancer (WBC), PIMA, WINE and BUPA Liver Disorders. All these databases are taken from the UCI machine repository [27].

5.1. Description of the Datasets

Let us briefly discuss the datasets, which we have taken for our experimental setup.

Wisconsin Breast Cancer Dataset: The dataset consists of d=9 features made on each of the 699 clinical cases of class c=2. The two distinct categories correspond to two different classes such as Benign, and Malignant. The problem is to classify each test point to its correct cases based on the nine tests.

PIMA Indians Diabetes Database: This dataset consists of n = 8 numerical medical attributes and c=2 classes (tested positive or negative for diabetes). There are n=768 instances. Further, data set related to the diagnosis of diabetes in an Indian population that lives near the city of Phoenix, Arizona.

BUPA Liver Disorders: This data set related to the diagnosis of liver disorders and created by BUPA Medical Research, Ltd. The dataset consists of d=5 attributes and c=2 number of classes. There are n=345 number of instances.

WINE Database: This is a three-class problem. It has 178 numbers of instances and 13 attributes. The instances are distributed into three classes. Class 1 contains 59, class 2 contains 71 and class three contains 48 numbers of patterns.

Table 1 presents a summary of the main features of each database that has been used in this study.

5.2. Classification Performance

For the case of the Wisconsin Breast Cancer, PIMA Indians Diabetes and BUPA Liver Disorders datasets, the total set of patterns was randomly divided into two equal parts (database1.dat and database2.dat). Each of these two sets was alternately used either as a training set or as a test set. As we use a stochastic method for data classification, result may vary from simulation to simulation. Each set has been simulated for 50 times. The results considered here is the average of 50 simulations. Table 2 summarizes the results obtained in the classification of these four data sets with the use of the FSN model.

From Table 2 it has been observed that for the case of Wisconsin Breast Cancer dataset on an average the hit percentage in the training set is at par with the hit percentage in test set. In the case of PIMA and BUPA datasets, on an average, the hit percentage of training set is significantly better than the hit percentage of test set.

Table 1. Descriptions of the datasets

	Number of Patterns	Number of Attributes	Number of Classes	Number of Patterns in Class 1	Number of Patterns in Class 2	Number of patterns in Class 3
PIMA	768	8	2	500	268	
BUPA	345	6	2	145	200	
WBC	699	10	2	458	241	
WINE	178	13	3	59	71	48

Further Table 3 shows the comparison of the proposed FSN with RBF and MLP. The comparative performance of FSN with RBF is superior, however it is competitive with MLP. Figure 3 shows the performance comparison of FSN with RBF by considering the hit percentage of training sets. The result obtained from FSN is dominating the results obtained from RBF. Similarly Figure 4 shows the performance comparisons of FSN with RBF by considering the hit percentage of test set and are very promising. However in the case of FSN with MLP Figure 5 and 6 shows a very close performance. In other words we can claim the obtained result form FSN is competitive with MLP.

Table 2. Results obtained from the FSN model

Dataset	Hit Percentage in the training set	Hit percentage in the test set
WBC1.dat	97.0858	95.5714
WBC2.dat	97.9656	97.1346
Average WBC	97.5257	**96.353**
Pima1.dat	81.0156	75.9376
pima2.dat	79.4532	75.1302
Average PIMA	80.2344	**75.5309**
liver1.dat	75.3488	70.1745
liver2.dat	76.9368	68.1502
Average LIVER	76.1428	**69.1476**
Wine1.dat	96.46065	98.3707
Wine2.dat	93.31455	98.3145
Average WINE	94.8876	**98.3426**

5.3. Dependability of the Classifier

In addition to the classification accuracy obtained from a classifier, dependability of a model resides on the consistency of the results obtained. To verify the dependability of the model standard deviation of the results obtained in 50 simulations are taken and presented in Table 4.

Table 3. Performance comparison of FSN, RBF and MLP

Dataset Used	FSN		RBF		MLP	
	Hit Percentage in Training Set	Hit Percentage in Test Set	Hit percentage in Training Set	Hit Percentage in Test Set	Hit percentage in Training Set	Hit Percentage in Test Set
WBCD	97.0858	95.5714	89.714	90.831	97.714	96.562
	97.9656	97.1346	91.977	89.714	98.0514	95.429
Average	97.5257	96.353	90.8455	90.2725	97.8827	95.9955
PIMA	81.0156	75.9376	74.479	78.125	80.729	76.823
	79.4583	75.1302	76.823	77.604	74.74	79.167
Average	80.2349	75.5309	75.651	77.8645	77.7345	77.995
BUPA	75.3488	70.1745	70.349	68.208	72.093	74.566
	76.9368	68.1502	71.676	65.698	71.512	73.41
Average	76.1428	69.1476	71.0125	66.953	71.8025	73.988
WINE	96.46065	98.3707	80.899	84.27	98.876	96.629
	93.31455	98.3145	76.404	85.393	100	95.506
Average	94.8876	98.3426	78.6515	84.8315	99.438	96.0675

5.4. Efficiency of the Classifier

Efficiency of the classifier may be evaluated in different ways. Here we present the error curves at Figure 7, 8, 9, and 10 for uniform 500 iterations. The rate of convergence of the models clearly depicts the efficiency of the models. The Mean Square Error (MSE) values have been evaluated from the average of the squared error of 30 nets distributed in the search space.

In this experiment, our objective is not to minimize the error, rather to select suitable weights for the net, which will yield competitive results. In addition to this we have to ensure that the net is not over trained. So the iterations suitable for breast cancer can be taken from the range [50-150] and in case of other thee dataset is in the range [100-200].

Table 4. Standard deviations obtained from 50 simulations with FS

Dataset	Standard Deviation in the training set	Standard Deviation in the test set
WBC1.dat	0.2256	0.4312
WBC2.dat	0.3432	0.3307
Average WBC	0.2844	**0.381**
pima1.dat	0.4331	1.2351
pima2.dat	0.7613	0.7287
Average PIMA	0.5972	**0.9819**
Liver1.dat	1.1693	0.9901
liver2.dat	0.4265	1.3746
Average LIVER	0.7979	**1.1824**
Wine1.dat	1.424589	0.679323
Wine2.dat	1.689402	1.062752
Average Wine	1.5570	**0.8710**

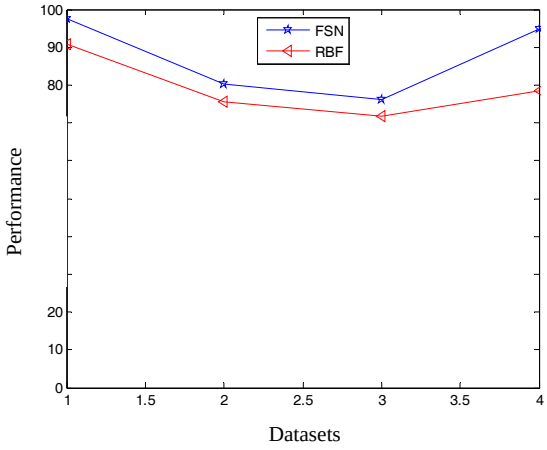

Figure 3. Performance of FSN Vs RBF in training Set

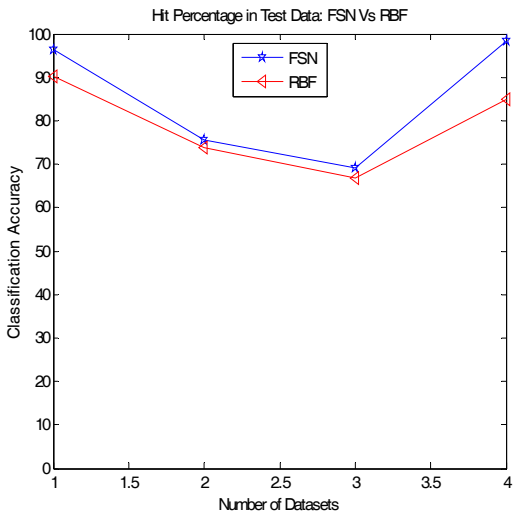

Figure 4. Performance of FSN Vs RBF in test set

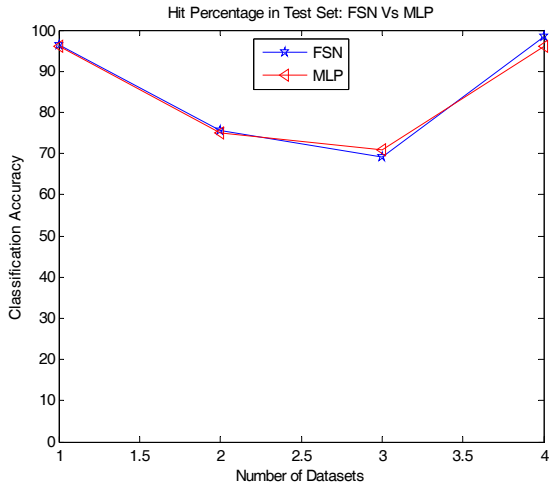

Figure 5. Hit percentage of training set performance comparison of FSN with MLP

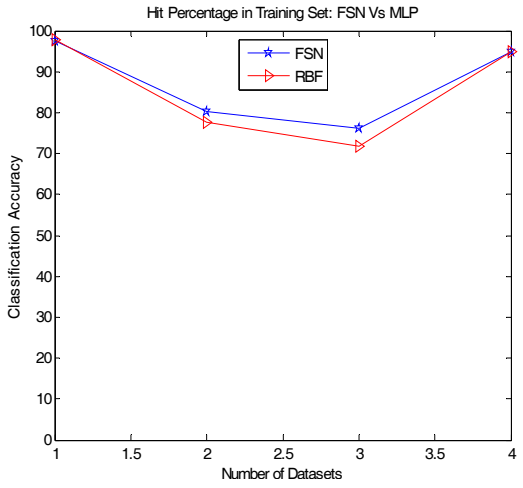

Figure 6. Hit percentage of test set performance comparison of FSN with MLP

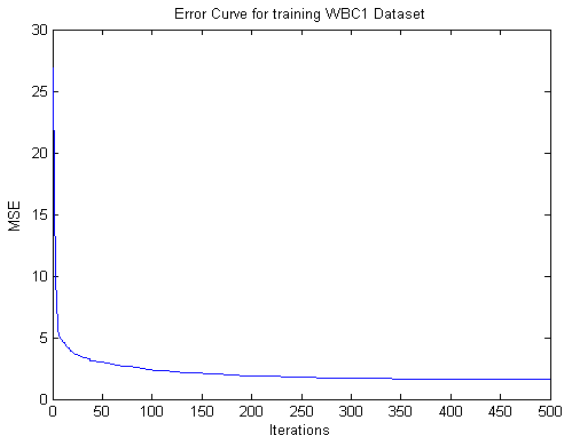

Figure 7. Error obtained from WBCD during training set

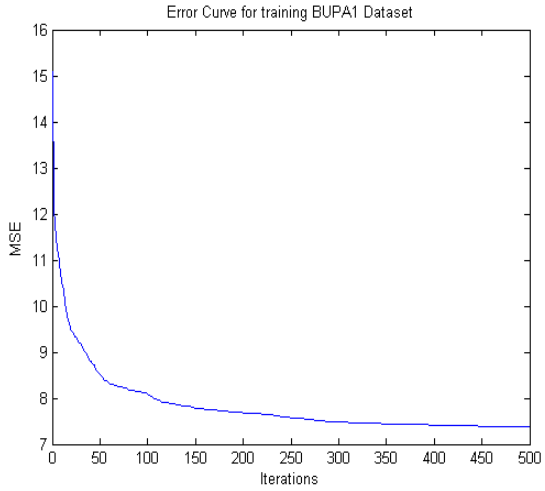

Figure 8. Error obtained from BUPA during training set

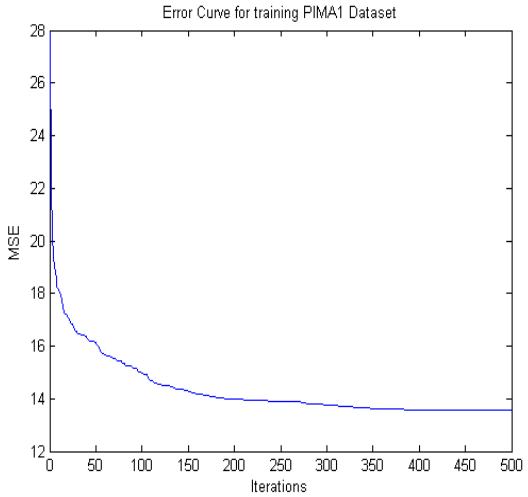

Figure 9. Error obtained from PIMA during training Set

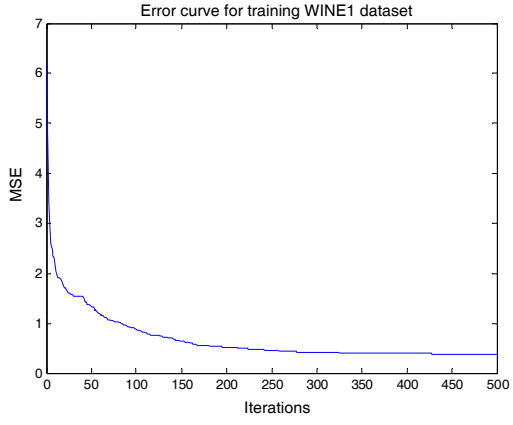

Figure 10. Error obtained from WINE during training Set

6. Conclusions

In this paper, we have evaluated the fuzzy swarm net (FSN) model for the task of classification in data mining. The FSN model expands the given set of inputs into three categories low, medium and high. These inputs are fed to the single layer feed forward artificial neural network. A set of such nets is being taken to spread in a distributed environment. Swarm intelligence technique is used to train these nets. The experimental studies demonstrated that the FSN model performs the pattern classification task quite well. The efficiency of the FSN models is competitive in terms of its rate of convergence.

Acknowledgement

One of the authors Dr. Satchidananda Dehuri would like to thank Department of Science and Technology, Govt. of India vide letter number SR/BY/E-07/2007 dated 03-01-2008 for financial support under the BOYSCAST fellowship 2007-2008 and Prof. S.-B. Cho, Department of Computer Science, Yonsei University, Seoul, Korea for his useful comments and providing the Soft Computing Laboratory facilities.

References

[1] G. Piatetsky-Shapiro, P. Symith, and R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, 1996.

[2] J. Han, and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2001.

[3] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, New York , Wiley, 2001.

[4] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules Between Sets of Items in Large Databases," *In ACM SIGMOD International Conference on Management of Data*, pp. 201-216, 1993.

[5] A. Jains, and R. Dubes, *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs NJ, USA, 1988.

[6] S. Theodoridis, and K. Koutroumbas, *Pattern Recognition*, Elsevier, 2006.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[8] C. P. Robert, *The Bayesian Choice*, New York, Springer Verlag, 2001.

[9] R. L. Winkler, Introduction to Bayesian Inference and Decision, *Gainesville, FL: Probabilistic Publications*, 450 pp, 2003.

[10] A. Gelman, J. Carlin, H. Stern, and D. Rubin, *Bayesian Data Analysis*, Chapman and Hall/CRC, 1995.

[11] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, Wiley-Inter Science, 2004.

[12] B. S. Dasarathy, *Nearest Neighbor Pattern Classification Techniques*, Los Aamitos, Calif: IEEE Computer Society Press, 1990.

[13] R. Herbrich, *Learning Kernel Classifiers: Theory and Algorithms*, MIT Press, 2001.

[14] C. M. Bishop, *Neural Networks for Pattern Recognition*, Clarendon Press, 1995.

[15] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wordsworth, Belmont, 1984.

[16] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Series in machine Learning, 1992.

[17] Y. C. Ho, and R. L. Kashyap, "A Class of Iterative Procedures for Linear Inequalities," *J. SIAM Contr.*, Vol. 4, No. 2, pp. 112–115, 1966.

[18] V. Vapnik, and A. Chervonenkis, "On the Uniform Convergence of Relative Frequencies of Events to their Probabilities," *Theory of Probability and its Applications*, Vol. 16, No. 2, pp. 264-280, 1971.

[19] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, Vol. 2, No. 2, pp. 955-974, 1998.

[20] P. J. Huber, *Robust Statistics*, New York, Wiley, (1981).

[21] R. Bellman, K. Kalaba, and L. A. Zadeh, "Abstraction and Pattern Classification," J. *Math. Anal. Appl.*, Vol. 13, No. 1, pp. 1–7, 1966.

[22] E. Czogala, and J. M. Leski, *Fuzzy and Neuro-Fuzzy Intelligent Systems*, Heidelberg: Physica-Verlag, 2000.

[23] J. Kennedy, and R. Eberhart, *Swarm Intelligence*, Morgan Kaufmann Academic Press, 2001.

[24] J. Kennedy, and R. Mendes, "Population Structure and Particle Swarm Performance," *In Proc. of the Congress on Evolutionary Computation*, Vol. 2, pp. 1671-1676, 2002.

[25] L. I. Kuncheva, "Switching Between Selection and Fusion in Combining Classifiers: An Experiment," *IEEE Trans. Syst. Man Cybern.-Part B*, Vol. 32, No. 2, pp. 146–156, 2002.

[26] S. Watanabe, T. Furuhashi, K. Obata, and Y. Uchikawa, "A Study on Feature Extraction Using a Fuzzy Net for Off-Line Signature Recognition," *Proc. of Int. Joint Conf. on Neural Networks*, Vol. 3, pp. 2857- 2860, 1993.

[27] C. L. Blake, and C. J. Merz, "UCI Repository of Machine Learning Databases," http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.

Bijan Bihari Mishra is working as a Professor in the Department of Computer Science and Engineer-ing, College of Engineering Bhu-baneswar, Orissa, INDIA. He received his Batchelor of Technology from Kanpur Uni-versity, India in 1984 and Master of Technology from Utkal University, Bhubaneswar, India in 2002. Presently continuing towards his Ph.D. in Biju Pattanaik University of Technology (BPUT). His research interests include Data Mining and Soft Computing.

E-mail: misra_bijan@yahoo.co.in

Satchidananda Dehuri is a Reader in P.G. Department of Information and Communication Technology, Fakir Mohan University, Vyasa Vihar, Balasore, Orissa. He received his M.Sc. degree in Mathematics from Sambalpur University, Orissa in 1998, and M.Tech. and Ph.D. degrees in Computer Science from Utkal University, Vani Vihar, Orissa in 2001 and 2006, respectively. He was at the Center for Theoretical Studies, Indian Institute of Technology Kharagpur as a Visiting Scholar in 2002. During May-June 2006 he was a Visiting Scientist at the Center for Soft Computing Research, Indian Statistical Institute, Kolkata. He is now in Soft Computing Laboratory, Yonsei University, Seoul, Korea as a post doct. Fellow under the BOYSCAST program of DST, Govt. of India on leave from Parent Institution. His research interests include Evolutionary Computation, Neural Networks, Data Warehousing and Mining. He has already published about 50 research papers in reputed journals and referred conferences, has published two books and edited one book in Springer-Verlag Heidelberg, and is acting as an editorial member of various journals. He chaired the sessions of various International Conferences.

E-mail: satchi.lapa@gmail.com

Ganapati Panda is presently working as a Professor and Head in the Dept. of Electronics & Communication Engineering at National Institute of Technology, Rourkela. Prior to this he was the Director of NIT, Jamshedpur. He already guided 16 Ph.D. Scholars and published more than 220 research papers in various International and National referred journals and conferences. He is a Fellow of National Academy of Engineering (FNAE), Fellow of National Academy of Science (FNASc.) and Senior Member IEEE. His present research interests are DSP, Digital Communication, Intelligent Instrumentation, Soft Computing and Sensor Networks.

E-mail: ganapati.panda@gmail.com

Pradipta Kishore Dash (SM'1990) is working as a Director Silicon Institute of Technology, Bhubaneswar, India. Earlier he was a Professor in the Faculty of Engineering, Multimedia Univer-sity, and Cyberjaya, Malaysia. He also served as a Professor of Electrical Engineering & Chairman, Center for Intelligent Systems, National Institute of Technology, Rourkela, India for more than 25 years. Dr. Dash holds D.Sc., Ph.D., M.E., and B.E. degrees in Electrical Engineering and had his Post-Doctoral education at the University of Calgary, Canada. His research interests are in the area of Power Quality, FACTS, Soft Computing, Deregulation and Energy Markets, Signal processing, and Data mining and Control. He had several visiting appointments in Canada, USA, Switzerland, and Singapore. To his credit he has published more than 150 International Journal papers and nearly 100 in International conferences. Prof. Dash is a Fellow of the Indian National Academy of Engineering and senior member of the IEEE, and Fellow of Institution of Engineers, India.

E-mail: pkdash_india@yahoo.co.in

Paper Handling Data:

Submitted: 10.04.2007
Accepted: 23.04.2008
Corresponding author: Dr. Satchidananda Dehuri, Department of Information and Communication Technology, Fakir Mohan University, Vyasa Vihar, Balasore-756019, India.