Using AdaBoost-based Multiple Functional Neural Fuzzy Classifiers Fusion for Classification Applications

Jyun-Yu Jhang[1] Chin-Ling Lee[2] Cheng-Jian Lin[3*] Chin-Teng Lin[4,5] Kuu-Young Young[4]

[1] Institute of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu 300, Taiwan
[2] National Taichung University of Science and Technology, Taichung 411, Taiwan
[3] Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan
[4] Department of Electrical and Control Engineering National Chiao Tung University, Hsinchu 300, Taiwan
[5] Faculty of Engineering and Information Technology University of Technology Sydney, Sydney 2007, Australia

Abstract. In this study, two intelligent classifiers, the AdaBoost-based incremental functional neural fuzzy classifier (AIFNFC) and the AdaBoost-based fixed functional neural fuzzy classifier (AFFNFC), are proposed for solving the classification problems. The AIFNFC approach will increase the amount of functional neural fuzzy classifiers based on the corresponding error during the training phase; while the AFNFC approach is equipped with a fixed amount of functional neural fuzzy classifiers. Then, the weights of AdaBoost procedure are assigned for classifiers. The proposed methods are applied to different classification benchmarks. Results of this study demonstrate the effectiveness of the proposed AIFNFC and AFFNFC methods.

1 Introduction

The object classification methods, as stated in [1]-[2], are to divide the initially undefined entities into classes. In each class the individuals are close to one another in some sense. These methods can be used to determine the particular symptoms of the diseases in medical science, to classify the species in biology, and to solve common classification problems in engineering and other fields. Most traditional statistical classification approaches, such as discrimination analysis, are built on the Bayesian decision theory [3], and each method generally has its own explicit probability model, but it works well and performs effectively only when the underlying assumptions are correct. For measurement of the similarity, twelve similarity structures were listed in the work of [4]. In supervised machine learning for classification, about 126 relative works are summarized in recent literature [5], which works are based on logical/symbolic techniques, perceptron-based techniques, and Statistics.

The boosting approaches are meta-algorithms for supervised learning, which are originally proposed to improve the accuracy of classification and prediction problem. Derived from machine learning, Valiant [6] presented the probably approximately correct (PAC) architecture and defined the notation of learnability. Furthermore, Kearns and Valiant [7] introduced the idea of weak learnability, and claimed the minimum performance of an algorithm should be better than a random guess. The above guess was proved by Schapire [8]. Schapire stated that the original classification results of dealing with the 2-class classification problem can be improved if a weak classifier could be divided into two new classifiers via a specific process of data filtering and grouping. Such approach was named as Boosting by Schapire. Later, Freund [9] developed a more efficient approach, but it suffered from some practical drawback. Afterwards, Freund and Schapire [10] proposed the AdaBoost algorithm and claimed the method can reduce the learning error and continuously decrease the testing error without over-fitting phenomenon. The traditional AdaBoost approach is hard to deal with the multi-class problems, and its performance highly depends on the design of weak classifiers. Consequently, the general solution for multi-class problem is to deploy hundreds or thousands weak classifiers for the traditional AdaBoost method.

Vast researches in neural classification have shown that neural networks [11-12] are promising alternatives to conventional classification methods. However, the meaning of each neuron and the decision of each weight are difficult to understand in the neural networks. Fuzzy entropy measure [13] is employed to partition the input feature space into decision regions and to select relevant features with good separability for the classification task. Neural fuzzy networks [14] integrate the advantages of both neural networks and fuzzy systems, which bring the low level learning ability, optimization capability and computational power of neural networks into fuzzy systems and give the high-level human-like thinking and

* Corresponding author: cjlin@ncut.edu.tw; Tel.: +886-953-002-825

reasoning of fuzzy systems to neural networks. Besides, the fuzzy inference systems can be adapted into with the ability of self-tuning and getting closer to neural networks.

The rest of this paper is organized as follows. In section 2, we introduce the traditional Adaboost method, functional neural fuzzy classifier proposed early, and our proposed AdaBoost-based approaches, and section III to their performance applying on three classification benchmarks. Finally, conclusions are drawn in Section 4.

2 The Proposed AdaBoost-based Approaches

This section describes the proposed AdaBoost-based incremental functional neural fuzzy classifier (AIFNFC) and AdaBoost-based fixed functional neural fuzzy classifier (AFFNFC). Besides, the AFFNFC can be divided into two different models, named the AFFNFC-1 and AFFNFC-2, based on the influence factor c. The general architecture of the proposed classifiers is shown in Figure 1.

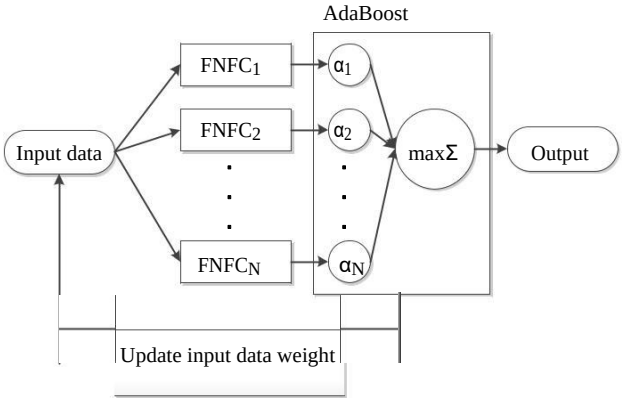

Figure 1. General architecture of the proposed classifiers

2.1. The Original AdaBoost Method

The AdaBoost method constructs a complex nonlinear strong classifier as a linear combination of simpler weak classifier. In every epoch, the method assigns the weights to the weak classifiers for committee voting, and a weak classifier with lower error rate will be assigned a larger weight. In the following, we show the details of the AdaBoost algorithm in the form of pseudo codes [15].

Algorithm: AdaBoost

1. Given N input examples $\{(X_1,Y_1),\ldots,(X_N,Y_N)\}$, with labels $y_i \in Y = \{0,1\}$.
2. Initialize the observation weights, $w_i = \frac{1}{N}$ for $i = 1,2,\ldots,N$.
3. For $t = 1,\ldots,T$
   (1) Fit a classifier $h_t(x)$ to the training data using weights w_i;
   (2) Calculate the training error
$$\varepsilon_t = \sum_{i=1}^{N} w_i I(h_t(x_i) \neq y_i) / \sum_{i=1}^{N} w_i$$
   (3) Calculate hypothesis weight
$$\alpha_t = log\frac{1-\varepsilon_t}{\varepsilon_t}$$
   (4) Update the observation weight
$$w_i \leftarrow w_i \cdot exp\left(\alpha_t \cdot I(h_t(x_i) \neq y_i)\right), i = 1,2,\ldots N$$
   (5) Renormalize w_i
4. Output:
$$H(x) = arg\,\max_k \sum_{t=1}^{T} \alpha_t \cdot I(h_t(x_i) = k)$$

In step 3, $I(h_t(x_i) \neq y_i)$ represents an identical function, when $h_t(x_i) \neq y_i$ it returns 1, otherwise is 0; ε_t is the error rate, α_t is the log-odds rate of ε_t and the weighted value of weak classifier at the t^{th} epoch. It can be easily observed that a smaller ε_t will increase the value of α_t; while ε_t is closed to 1/2, α_t will approach to zero.

2.2 The Functional Neural Fuzzy Classifier

Before introducing the proposed AdaBoost-based classifiers, the weak classifiers - functional neural fuzzy classifier (FNFC) [16], is briefly reviewed in this subsection. The FNFC was a traditional TSK-type functional neural fuzzy network integrated with the functional-link neural network (FLNN) [17] generating complex nonlinear combinations of input variables to the consequent part of the fuzzy rules. As shown in Figure 2, the network architecture can be divided into five layers.

The FNFC implements a fuzzy IF-THEN rule as following,

Rule-j:

IF x_1 is A_{1j} and x_2 is A_{2j} ... and x_i is A_{ij} ... and x_N is A_{Nj}

$$\text{THEN } \hat{y}_j = \sum_{k=1}^{M} w_{kj}\Phi_k$$

where x_i and $\hat{y}_j$ are the input and local output variables, respectively; N is the number of input variables; Rule-j is the j^{th} fuzzy rule; A_{ij} is the linguistic term of precondition part with Gaussian membership function; w_{kj} is the link weight of the local output; Φ_k is the basis trigonometric function of input variables and M is the number of basis function.

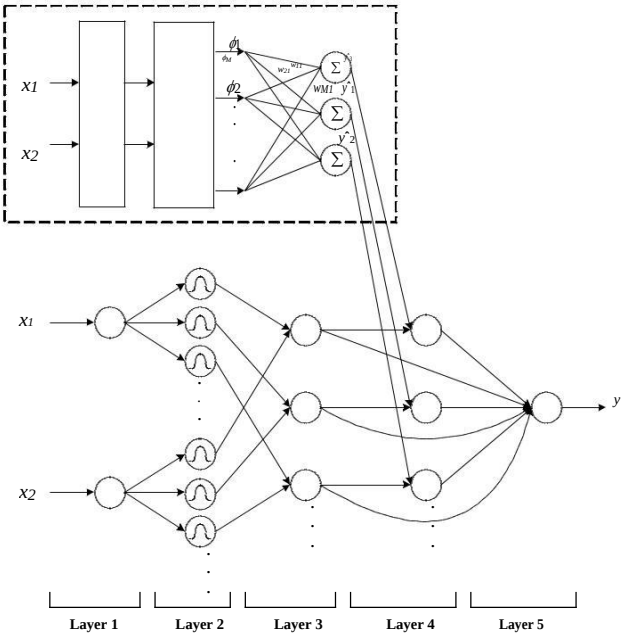

Figure 2. General Architecture of the FNFC classifier.

Layer 1: Each node in this layer only transmits input values to layer 2 without computation.

$$u_i^{(1)} = x \quad (1)$$

Layer 2: The calculated membership value represents the degree which an input value belongs to a fuzzy set $\mathbf{A_{ij}}$ in layer 2. We use the Gaussian membership function here, which is defined as

$$u_{ij}^{(2)} = exp\left(\frac{-\left[u_i^{(1)} - m_{ij}\right]}{\sigma_{ij}^2}\right) \quad (2)$$

where m_{ij} and σ_{ij} are the mean and variance of Gaussian membership function, respectively.

Layer 3: The product operator is adopted to perform the precondition part of the fuzzy rules. The corresponding equation of each inference node is presented as,

$$u_j^{(3)} = \prod_i u_{ij}^{(2)} \quad (3)$$

Layer 4: Node in layer 4 received the output from layer 3 and the FLNN output. As a result, the equation can be summarized as following,

$$u_j^{(4)} = u_j^{(3)} \cdot \sum_{k=1}^{m} w_{kj} \cdot \Phi_k \quad (4)$$

where w_{kj} is the corresponding link weight of the FLNN

and Φ_k is the functional expansion of input variables. The functional expansion uses a trigonometric polynomial basis function, given by

$x_1 \sin(\pi x_1) \cos(\pi x_1) x_2 \sin(\pi x_2) \cos(\pi x_2)$ for two-dimensional input variables. Therefore, M is the number of basis functions,

$\mathbf{M = 3 \times N}$, where N is the number of input variables.

Layer 5: The out node in layer 5 acts as a defuzzifier with

$$y = u^{(5)} = \frac{\sum_{j=1}^{R} u_j^{(4)}}{\sum_{j=1}^{R} u_j^{(3)}} = \frac{\sum_{j=1}^{R} u_j^{(3)} \left(\sum_{i=1}^{M} w_{ij} \Phi_i\right)}{\sum_{j=1}^{R} u_j^{(3)}} \quad (5)$$

where R is number of fuzzy rules and y is the output of FNFC model.

The learning algorithm of the FNFC includes the structure learning phase and parameter learning phase. The division of input space and the establishment of fuzzy rules are performed automatically in structure learning phase; meanwhile, the link weights in consequent part and the parameters of the membership function are adjusted in the parameter learning phase. The details of the FNFC can be found in [16].

2.3 The Proposed AIFNFC Approach

When handling the multi-class classification problem, most boost-based methods restricted the problem into multiple two-class problems. Without this restriction, we proposed a new AdaBoost-based incremental functional neural fuzzy classifier (AIFNFC), which uses the FNFCs as the weak classifiers. By using the FNFC as the weak classifier, AIFNFC approach improves the disadvantage of boost-based methods for multi -class classification. In each epoch, the weights will be updated by AdaBoost approach; the better classifier is assigned a higher weight to improve the final classification result. The pseudo code of AIFNFC is listed in the following.

Algorithm: *AIFNFC*

1. With *C* classes, given *N* training examples, $\{(x_1, y_1, z_1), \ldots, (x_i, y_i, z_i), \ldots, (x_N, y_N, z_N)\}$, where x_i represents the given input attribute, label y_i, $y_i \in Y = \{1,2,\ldots,C\}$, is the desired class, and the label z_i, $z_i \in Z = \{0,1\}$, is the predicted outcome.
2. Initialize the data weights w_i, $z_i = \frac{1}{N}$ for i=1,2,…,N, where N is amount of the training data amount, T for amount of the epoch amount, and t for epoch index.
3. For $t = 1, \ldots, T$;
   (1) If $z_i = 0$, create a weak classifier $h_t(x_i)$ by FNFN;
   (2) Calculate the classification error with all data by
   $$\varepsilon_t = \sum_{i=1}^{N} w_i I(h_t(x_i) \neq y_i) / \sum_{i=1}^{N} w_i \quad (6)$$
   where $I(\cdot)$ represents an identical function
   (3) If $\varepsilon_t > 0.5$, go to step 4;
   (4) Calculate the weight of this weak classifier by
   $$\alpha_t = log \frac{1-\varepsilon_t}{\varepsilon_t} \quad (7)$$
   (5) Update the i[th] data weight by
   $$w_i \leftarrow w_i \cdot exp\left(\alpha_t \cdot I(h_t(x_i) \neq y_i)\right) \quad (8)$$
   (6) Update the predicted outcome label by

$$z_i = \begin{cases} 1, if & h_t(x_i) = y_i \\ 0, if & h_t(x_i) \neq y_i \end{cases} \quad (9)$$

(7) Renormalize w_i;

4. Output the resulted strong classifier by

$$H(x) = arg\ \max_k \sum_{t=1}^{T} \alpha_t \cdot I(h_t(x_i) = k) \quad (10)$$

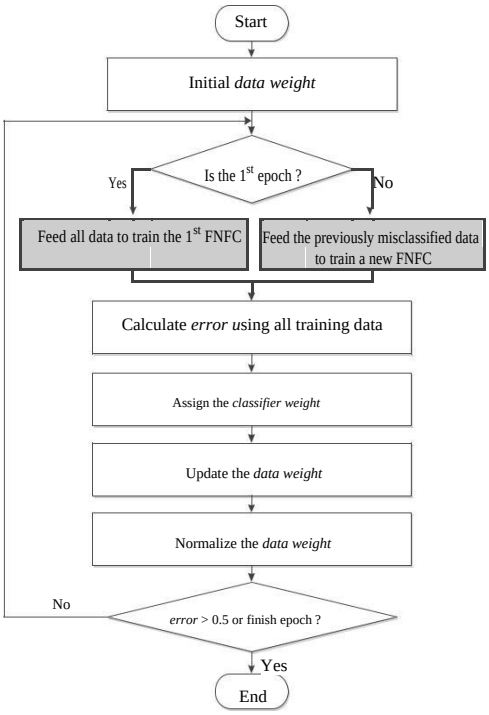

Figure 3. Flowchart of the proposed AIFNFC approach

2.4 The Proposed AFFNFC Approach

We observe that the poor classifier in the AIFNFC approach is generally followed by a relatively better classifier. Hence, one modification of AIFNFC is needed to force all classifiers on contributing themselves with greater effectiveness. The proposed AdaBoost-based fixed functional neural fuzzy classifier (AFFNFC) is characterized by a customized amount of weak classifiers, and the AdaBoost method is used to update the weight of these weak classifiers that lastly contribute the final outcome via their weights. In addition, the amount of weak classifiers should be defined before executing the epoch process. The pseudo code of AIFNFC is listed in the following.

Algorithm: *AFFNFC*

1. With C classes, given N training examples $\{(x_1, y_1), \ldots, (x_i, y_i), \ldots, (x_N, y_N)\}$ where x_i represents the given input attribute, label $y_i \in Y = \{1, 2, \ldots, C\}$ the desired class.
2. Initialize the data weight $w_i = \frac{1}{N}$ for $i = 1, 2, \ldots, N$, epoch amount T, epoch index t, the amount of classifiers M , and the ratio c.
3. Train M FNFN classifiers.
4. For $t = 1, \ldots, T$
   (1) For $m = 1, \ldots M$
   Fit a classifier $h_m(x_i)$;
   Calculate the classification error by

$$\varepsilon_t = \sum_{i=1}^{N} w_i I(h_m(x_i) \neq y_i) / \sum_{i=1}^{N} w_i \quad (11)$$

   (2) Calculate the weight of this weak classifier by

$$\alpha_{t,m} = log \frac{1-\varepsilon_{t,m}}{\varepsilon_{t,m}} \cdot c \quad (12)$$

   where the influence factor $c = (M - r)/M$

   (3) Update the i^{th} data weight by

$$w_i \leftarrow w_i \cdot exp(\alpha_{t,m} \cdot I(h_m(x_i) \neq y_i) \cdot c) \quad (13)$$

   (4) Re-normalize w_i

5. Calculate the sum of T weights for the m^{th} classifier by

$$\beta_m = \sum_{t=1}^{T} \alpha_{t,m} \quad (14)$$

6. Output the resulted strong classifier by

$$H(x) = arg\ \max_k \sum_{m=1}^{M} \beta_m \cdot I(h_m(x_i) = k) \quad (15)$$

After a certain training epoch conducted, all error rates of classifiers are sorted in descending order, and the number of order is used to determine the influence factor c. The factor c is defined as $(M - r) / M$, where M represents the amount of fixed classifiers, and positive integer $r \in [0, M - 1]$ represents the ranking order of the classifier. A specific classifier with $r = 0$ means this classifier is with the lowest error rate among all the classifiers.

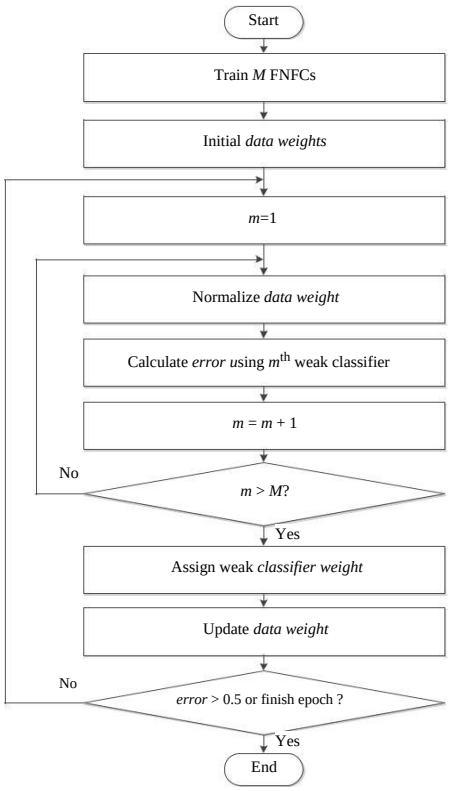

Figure 4. Flowchart of the proposed AFFNFC approach

There are two design schemes in this paper for different influence factor c. When there are some classifiers of the same error rate, the ranking factor r would be set according to their processing order, and this scheme is named AFFNFC -1. The other design scheme, called the AFFNFC-2, will assign the same ranking factor r to all the classifiers of the identical error rate.

3 Experimental Results

In this study, three benchmark dataset, Iris, Wine, and Breast Cancer are measured to evaluate the performance of the proposed methods. These three datasets come from the Donald Bren School of Information and Computer Science University of California at Irvine (UCI), the well-known Machine Learning Repository website [18]; Iris classification is a low-dimensional problem, and Breast Cancer and Wine classifications are multi-dimensional problems. Table 1 summarizes the amounts of samples, attributes and classified classes. Since this website recoded many benchmark datasets for machine learning, researchers can easily apply these benchmarks to verify their classification approaches, and conduct the necessary comparison of the corresponding training and testing results. In the rest of this section, we compare the performances of AIFNFC, AFFNFC-1, AFFNFC-2, and other methods combined with FNFC model, such as voting method with multiple FNFCs and single FNFC.

Table 1. Statistics of the chosen benchmark datasets.

Dataset	#Samples	#Attributes	#Classes
Iris	150	4	3
Wine	178	13	3
Brest Cancer	683	9	2

To compare with other methods, all the commonly used parameters are set identical initially, the learning rate η is set to 0.01, the entropy is set to 0.3, the amount of training epochs is set to 100 times, and the numbers of weak classifiers FNFCs are respectively set to 5, 10 and 15 for different experiment. In these experiments, half patterns of dataset are selected randomly as the training set, and the other patterns are used as testing data. For example, 75 instances of original 150 examples in Iris dataset will be used as training data and the rest 75 instances are applied for the testing. In addition, the randomized selecting process will be carried out five times for each benchmark dataset.

Example 1: Classification of the Iris Data

The Iris dataset is by Fisher [19], which is frequently chosen to verify newly designed algorithms in machine learning. The classification of the iris dataset is to differentiate the species based on the iris sepal (length and width) and petal (length and width). The Iris data comprises 150 actual instances that are distributed into three species: Iris Setosa, Iris Versicolout and Iris Virginica. Each iris dataset has four input features and one predicted attribute, , shown in Table 2.

Table 2. Attribute information of the Iris data

Attribute Type	No.	Attribute Description
Input features	1	Sepal Length
	2	Sepal Width
	3	Petal Length
	4	Petal Width
Predicted Attribute		Class of Iris Plant: (1) Iris Setosa (2) Iris Versicolout (3) Iris Virginica

We repeated the experiment on 5 different training-testing datasets that were obtained via a random selecting process from the original Iris data. Table 3 summarizes the five-run averaged experimental accuracy rates of the AIFNFC, AFFNFC-1, AFFNFC -2, single FNFC, and voting method with multiple FNFCs. Compared to the other approaches, the proposed AFFNFCs are generally with better performances.

Table 3. Average Accuracy on Iris Data

	Classifiers Number	Training (%)	Testing (%)
AIFNFC	-	97.60	92.27
AFFNFC-1	5	98.67	95.20
	10	99.47	96.27
	15	99.73	96.53
AFFNFC-2	5	98.93	96.53
	10	99.47	96.27
	15	99.73	96.53
Voting method with multiple FNFCs	5	97.60	93.87
	10	98.67	96.27
	15	98.93	95.73
FNFC (single)	-	95.36	92.96

Example 2: Classification of the Wine Data

The Wine dataset [20] is the results of a chemical analysis of 178 wines which are brewed in the same area in Italy but derived from three different cultivars. The classification of wine type is based on 13 constituents found in the wines. The information of constituents and predicted attribute are presented in Table 4.

Table 4. Attribute information of the wine data

Attribute Type	No.	Attribute Description
Input Attribute	1	Alcohol
	2	Malic Acid
	3	Ash
	4	Alcalinity of Ash
	5	Magnesium
	6	Total Phenols
	7	Flavanoids
	8	Nonflavanoid Phenols
	9	Proanthocyanins
	10	Color Intensity
	11	Hue
	12	OD280/OD315 of Diluted Wines

	13	Proline
Predicted Attribute		Class of Wine Identifier (1-3) (1)Barolo(2) Grignolino (3) Barbera

Table 5 lists the five-run averaged accuracy rates experimented on the Wine dataset. In this experiment, the AIFNFC method with fewer classifiers is a bit weak on performance since the applied number of AIFNFC classifiers is not fixed. Therefore, there are fluctuations of accuracy rate in classification results of the AIFNFC method. But, the proposed AFFNFCs still have better performance than other methods.

Table 5. Average accuracy on wine data

Algorithms	Classifiers Number	Training (%)	Testing (%)
AIFNFC	-	95.96	87.19
AFFNFC-1	5	98.65	91.69
	10	**100.0**	**93.48**
	15	100.0	93.26
AFFNFC-2	5	98.65	91.69
	10	**100.0**	**93.48**
	15	100.0	93.26
Voting method with multiple FNFCs	5	98.43	91.24
	10	99.10	93.03
	15	99.33	93.48
FNFC (single)	-	94.65	85.71

Example 3: Classification of the Breast Cancer Data

This breast cancer dataset was created by Dr. William H. Wolberg at the Wisconsin University [21]. The total number of the original samples is 699. Since there are 16 samples contain missing values, only 683 patterns were used in this experiment. The columns of sample id and class label in original dataset are removed, so each pattern only consists of nine input features and one predicted attribute, as shown in Table 6.

Table 6. Attribute information of the breast cancer data

Attribute Type	No.	Attribute Description
Input Attribute	1	Clump Thickness
	2	Uniformity of Cell Size
	3	Uniformity of Cell Shape
	4	Marginal Adhesion
	5	Single Epithelial Cell Size
	6	Bare Nuclei
	7	Bland Chromatin
	8	Normal Nucleoli
	9	Mitoses
Predicted Attribute		Class of Breast Cancer: (1) Benign (2) Malignant

Better testing classification results are with the AIFNFC method, since the AIFNFC is great at improving the best classifier which finally influences the final classification results. The AFFNFC always has good results in training phase, but it did not conduct a superior performance in this case. The experimental results are listed in Table 7.

Table 1. Average accuracy on breast cancer data

Algorithms	Classifiers Number	Training (%)	Testing (%)
AIFNFC	-	97.94	**96.51**
AFFNFC-1	5	98.34	96.06
	10	98.40	96.00
	15	**98.45**	96.06
AFFNFC-2	5	98.34	96.06
	10	98.40	96.00
	15	98.40	96.11
Voting method with multiple FNFCs	5	98.05	95.89
	10	97.77	95.43
	15	97.94	95.89
FNFC (single)		98.09	95.45

When massive voting decisions are applied, the experimental results show that the wrong decisions of most of the classifiers will increase the misclassification rate for the voting method with multiple FNFCs. On the other hand, the proposed AdaBoost-based approaches have the benefit of arranging larger associated weights for the classifiers of higher accuracies, which release the influence of low-accuracy classifiers. Besides, if a training instance is misclassified by a classifier of greater weight, the final classification result can be improved with the other classifiers when the summation of their weights is bigger than the one of misguided classifier. For example, for the 34th instance in Iris dataset, both the AFFNFC and voting method are with 7 misclassified classifiers, then the voting method makes a wrong decision, but the final classification of the AFFNFC remained correct.

4 Conclusions

In this paper, we propose two intelligent AdaBoost-based approaches, AIFNFC and AFFNFC, for the classification applications. The AIFNFC approach will increase the amount of functional neural fuzzy classifiers based on the corresponding error during the training phase; while the AFNFC approach is equipped with a fixed amount of functional neural fuzzy classifiers. Then, the weights of AdaBoost procedure are assigned for classifiers. Experimental results show that the proposed AIFNFC and AFFNFC approaches can effectively improve the classification performances.

References

1. R. M. Cormack, “A Review of Classification,” Journal of the Royal Statistical Society (Series A), Vol. 134, No. 3, pp. 321-367, 1971.
2. J. M. CAMIN and R. S. SOKAL, “A Method for Deducing Branching Sequences in Phylogeny,” Evolution, Vol. 19, No. 3, pp. 311-326, 1965.
3. P. O. Duda and P. E. Hart, “Pattern classification and scene analysis,” Wiley, NY, USA, 1973.
4. J. A. Hartigan, “Representation of Similarity Batrices by Trees,” Journal of the American Statistical Association, Vol. 62, No. 320, pp. 1140-1158, 1967.

5. S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques," Informatica, Vol. 31,pp. 249-268, 2007.
6. L. G. Valiant, "A Theory of the Learnable," Communications of the ACM, Vol. 27, pp. 1134-1142, 1984.
7. M. Kearns and L. Valiant, "Learning Boolean Formulae or Finite Automata is as Hard as Factoring," Technical Report, TR-14-88, Aiken Computation Laboratory, Karrard University, Cambridge, MA.
8. R. E. Schapire, "The Strength of Weak Learnability," Machine Learning, Vol. 5, pp. 197-227, 1990
9. Y. Freund, "Boosting a Weak Learning Algorithm by Majority," Information and Computation, Vol. 121, pp. 256-285, 1995.
10. Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," Journal of Computer and System Sciences, Vol. 55, No. 1, pp. 119-139, 1997.
11. G. Peter Zhang, "Neural Networks for Classification: A Survey," IEEE Transactions on Systems, Man, and Cybernetics:Applications and Reviews, Vol. 30, No. 4, pp. 451-462, Nov. 2000.
12. K. Devulapalli, "Neural Networks for Classification and Regression. Biom. Biostat. Int. J.,Vol. 2, Issue 6, DOI: 10.15406/bbij.2015.02.00046S, 2015.
13. L. A. Zadeh, "Fuzzy Sets," Information and Control, Vol. 8, pp. 338-353, 1965.
14. C. J. Lin and H.-Y. Lin, "Mobile Robot Wall-following Control Using a Fuzzy CMAC with Group-based Strategy Bacterial Foraging Optimization," International Journal of Advanced Robotic Systems, Vol. 14, Issue 4, DOI: https://doi.org/10.1177/1729881417720872, July 14, 2017.
15. J. Zhu, H. Zou, S. Rosset and T. Hastie, "Multi-class AdaBoost," Statistics and Its Interface, Vol. 2, pp.349–360, 2009.
16. C. H. Cheng, M. T. Su, C. J. Lin, and C. T. Lin, "A Hybrid of Bacterial Foraging Optimization and Particle Swarm Optimization for Evolutionary Neural Fuzzy Classifier," International Journal of Fuzzy Systems, Vol. 16, No. 3, September 2014, pp. 422-433.
17. J. C. Patra, R. N. Pal, B. N. Chatterji and G. Panda, "Identification of Nonlinear Dynamic Systems Using Functional Link Artificial Neural Networks," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 29, No. 2, pp. 254-262, 1999.
18. UCI Machine Learning Repository, 2013, Available from URL: http://archive.ics.uci.edu/ml/
19. R. A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems," Annual Eugenics, Vol. 7, No.2, pp. 179-188, 1936.
20. M. Forina, R. Leardi, C. Armanino, and S. Lanteri, "PARVUS - An Extendible Package for Data Exploration," Journal of Chemometrics, Vol. 4, No. 2, pp. 191-194, 1990.
21. W. H. Wolberg and O. L. Mangasarian, "Multisurface method of pattern separation for medical diagnosis applied to breast cytology," Proceedings of the National Academy of Sciences of the United States of America, Vol. 87, pp. 9193-9196, 1990.