ARTICLE IN PRESS

Available online at www.sciencedirect.com

ScienceDirect

Fuzzy Sets and Systems ▮▮▮ (▮▮▮▮) ▮▮▮–▮▮▮

FUZZY sets and systems

www.elsevier.com/locate/fss

Creating ensembles of classifiers via fuzzy clustering and deflection

Huaxiang Zhang[a,*], Jing Lu[b]

[a] *Department of Computer Science, Shandong Normal University, Jinan 250014, Shandong, China*
[b] *Department of Computer Science, Shandong College of Finance, Jinan 250014, Shandong, China*

Received 17 March 2009; received in revised form 25 November 2009; accepted 27 November 2009

Abstract

Ensembles of classifiers can increase the performance of pattern recognition, and have become a hot research topic. High classification accuracy and diversity of the component classifiers are essential to obtain good generalization capability of an ensemble. We review the methods used to learn diverse classifiers, employ fuzzy clustering with deflection to learn the distribution characteristics of the training data, and propose a novel sampling approach to generate training data sets for the component classifiers. Our approach increases the classification accuracy and diversity of the component classifiers. The approach is evaluated using the base classifier c4.5, and the experimental results show that it outperforms Bagging and AdaBoost on almost all the randomly selected 20 benchmark UCI data sets.

Keywords: Fuzzy clustering; Ensemble classifier; Deflection; Information entropy

1. Introduction

Ensembles of classifiers improve the classification accuracy through combining the decisions made by its component classifiers. Since an ensemble could have better generalization ability than its component classifiers, ensemble methods have been widely used in many areas [1–6], and constructing ensembles of classifiers have become a hot research topic. In the context of multiple classifier systems, diversity among the base classifiers is known to be a necessary condition for improving the performance of the ensemble, and high diversity means the component classifiers should be as diverse as possible.

Diversity contributes positively to the accuracy of an ensemble [7], and many approaches have been introduced to create diverse classifiers. The methods of obtaining diverse component classifiers include using different learning algorithms for the base classifiers [6,8], sampling the training data [9–11], projecting the training data into different feature subspaces [12–14] and disturbing the outputs of the training data by adding noise to the outputs of the training data or randomly switching the labels of the training examples [15,16]. A nonlinear boosting projection approach [17] learns diverse base classifiers by focusing on different subspaces of the training data based on the random subspace method. A survey of diversity creation methods has been given by Gavin Brown et al. [18].

Diversity can be obtained by manipulating the training data and training each learner with a slightly different training data set. This is probably the most widely investigated method of ensemble training. Among all the above diversity construction methods, bagging [9] and boosting [10] are the most widely used ones.

* Corresponding author. Tel.: +86 531 86180244.
E-mail address: huaxzhang@hotmail.com (H. Zhang).

0165-0114/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.fss.2009.11.013

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Bagging is a bootstrap ensemble method that trains each classifier on a randomly drawn training set. In bagging, diversity is obtained by constructing each base classifier with a different set of labeled examples, which is obtained from the original training set by re-sampling with replacement. The training set of each classifier consists of the same number of training data as that of the original training set, and some examples may be selected repeatedly while others may not be selected at all. The final decision in bagging is obtained by combining the decisions of the component classifiers with un-weighted voting. It is believed that bagging improves the classification performance mainly by reducing the variance error. Many variants of bagging with excellent classification performance have been developed [11,12,19–21].

Boosting is another commonly used algorithm for generating ensembles. In boosting, each component classifier is generated based on a weighted training data set. The weights of the training data are updated after a classifier has been trained, and a new classifier is generated based on data re-sampled from the original data set with new weights, i.e., examples that are incorrectly predicted by the previous classifier are instanced more frequently. A weighted vote is used to make the final class assignment in AdaBoost. Variants of AdaBoost have been presented in [21–24].

Both the bagging and boosting approaches generate ensembles by training each classifier on a bespoke data set, and some theoretical works [18,25,26] explain the effectiveness of bagging and boosting based on the bias-variance decomposition of classification error and prove the bounds and the margins on their errors.

A new classifier ensemble method named Rotation Forest has been proposed [27]. The main idea of Rotation Forest is to divide the feature set in subsets of dimension M which is less than the original training data feature number and to apply the principal component analysis (PCA) on the features that belong to a given subset. All the generated projections using each data set with M features are combined to build a projection matrix that builds a modified training set. Some variants of Rotation Forest [2,28] have been proposed, and it is reported that RotBoost works well.

All the boosting type algorithms used to generate diverse classifiers through manipulating the training data implicitly employ the intuitive idea that some examples may be hard to classify and others easy. How can we express this characteristic for each training example? AdaBoost directly uses the classification results of the already learned classifiers to update the weights of the examples. The examples with high weights are considered to be hard ones and those with low weights are considered to be easy ones. What does a weight really mean? We can imagine that if an example is close to the decision boundary of a class, it may be difficult to be classified and if an example is close to the decision boundaries of several classes, classifying it becomes more difficult than those close to a decision boundary shared by a few classes; and if an example is far away from the decision boundary of a class, it may be easy to be classified. The classification difficulty degree of an example that we regard as its distribution characteristic may be expressed by some metric information measuring its degree of closeness to decision boundaries.

Each instance in the training set definitely belongs to one class. Sometimes we may fuzzy the boundaries among data, as we commonly use fuzzy approaches to cluster unlabeled data in unsupervised learning. Even there are clear boundaries among the labeled data of different classes, data in the same class still have some characteristics representing their closeness degrees to the class boundaries. It is easy to understand that the characteristic of a data point located far away from the class boundaries is different from that of a data point located near the class boundaries. This characteristic represents the fuzziness of the training data and can be used when creating ensembles with diverse classifiers.

In this research, we present two novel ensemble construction approaches. The first method applies fuzzy clustering to the training data. The second proposed approach builds an ensemble with deflection techniques. Both approaches achieve higher accuracies than classical approaches, and accelerate the ensemble construction process by learning two different classifiers simultaneously in each iteration step. The predictive accuracies of both ensembles have been evaluated and discussed.

This paper is structured as: Section 2 introduces how to fuzzy the training data; Section 3 describes the base classifier generation approach; the ensemble learning algorithm is given in Section 4 and experimental results on 20 UCI data sets [29] are given in Section 5; conclusions of this research are presented in Section 6.

2. Fuzzy clustering the training data

We need to employ an approach different from that of Bagging and Adaboost to generate a training data set for each base classifier of an ensemble. It is easy to understand that the data close to class boundaries is more difficult to be labeled than those located far away from the boundaries, and this characteristic of data distribution is helpful to the construction of diverse component classifiers of an ensemble.

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Clustering is a popular technique commonly used in unsupervised learning to reveal the inherent data structure of an unlabeled data set, and fuzzy c-means (FCM) clustering [30] is a classical method used in semi-supervised and unsupervised learning. We use FCM to cluster the training data and learn the fuzzy memberships of the training data. The fuzzy memberships of a training instance represent the degree the instance belongs to different classes. If an instance can be easily labeled, the distribution of its memberships should be quite unbalanced. This means that one of its memberships approaches one and all the others approach zero. If it is difficult to be labeled, the instance may be close to the class boundaries. In this case, its memberships should be balanced among these classes. We employ FCM method to calculate the fuzzy memberships for each training instance based on the following objective function:

$$J_o = \sum_{i=1}^{d}\sum_{j=1}^{c} \mu_{ij}^m \|\mathbf{x}_i - \mathbf{v}_j\|^2 \tag{1}$$

where m is any real number greater than one and represents the fuzziness degree, μ_{ij} is the degree of membership of the ith instance $\mathbf{x}_i$ with respect to class j, c is the number of classes of the training data, and d denotes the training data set size. $\mathbf{x}_i$ is the ith training instance, and $\mathbf{v}_j$ is the center of class j. $\|*\|$ denotes the Euclidean distance representing the similarity between an instance and the class center.

Fuzzily partitioning the training data is carried out through iteratively optimizing the objective function J_o, and during each iteration step, the membership μ_{ij} and the class center $\mathbf{v}_j$ is updated as

$$\mu_{ij} = \left[\sum_{k=1}^{c}\left(\frac{\|\mathbf{x}_i - \mathbf{v}_j\|^2}{\|\mathbf{x}_i - \mathbf{v}_k\|^2}\right)^{1/(m-1)}\right]^{-1} \tag{2}$$

$$\mathbf{v}_j = \frac{\sum_{i=1}^{d} \mu_{ij}^m \mathbf{x}_i}{\sum_{i=1}^{d} \mu_{ij}^m} \tag{3}$$

The iteration process stops when the objective function J_o converges.

In order to process data with non-numerical attributes, we define the nominal attribute difference between two instances to be 0 if the attribute values are the same or 1 if they are different. When calculating the class center, we process the data set and count the number for each value of the nominal attribute, and denote the numbers as nominal attribute value numbers. We replace the nominal attribute value in (3) with 1 when its corresponding value number is maximized or 0 if not.

As FCM converges to one minimum of the objective function J_o which may have multiple minimizers, we adopt a technique to get more minimizers of J_o. After a minimum of J_o has been obtained, we reformulate it with a transformation, such that the reformulated objective function will not obtain minima at previous minimal points but keeps all the other minima of the original objective function locally unchanged. This property is called the deflection property [31]. Each time, we replace the objective function with the reformulated objective function and apply FCM approach to the new objective function to learn the fuzzy memberships. Given the kth reformulated function $J_{o,k}$ and the fuzzy memberships learned using the FCM, the function transformation is obtained

$$J_{o,k+1} = J_{o,k} \cdot \mathrm{Tanh}(\lambda_k \|\boldsymbol{\mu} - \boldsymbol{\mu}_k\|) \tag{4}$$

where $\boldsymbol{\mu}$ is the fuzzy membership vector, $\boldsymbol{\mu}_k$ is the kth group of fuzzy memberships learned by applying FCM to $J_{o,k}$, and λ_k is a relaxing parameter.

3. Generating training data set for component classifiers

After the fuzzy memberships have been calculated, we employ the concept of entropy commonly used in information theory to characterize the fuzziness of each training instance, and calculate the fuzzy information it contains

$$\mathrm{Info}(\mathbf{x}_i) = -\sum_{j=1}^{c} \mu_{ij} \log_2 \mu_{ij} \tag{5}$$

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

We notice that $\mathrm{Info}(\mathbf{x}_i)$ is zero if one μ_{ij} equals 1 and all the others equal 0. This may imply that $\mathbf{x}_i$ can be labeled easily. If each μ_{ij} of $\mathbf{x}_i$ equals $1/c$, $\mathrm{Info}(\mathbf{x}_i)$ is maximized, and in this case, labeling $\mathbf{x}_i$ is difficult. If we calculate a weight for each training instance according to $\mathrm{Info}(\mathbf{x}_i)$, then we can build an individual classifier h_1 on weighted bootstrap samples from the original data set.

The reciprocal of the fuzzy information is

$$\overline{\mathrm{Info}}(\mathbf{x}_i) = \frac{1}{\mathrm{Info}(\mathbf{x}_i)} \tag{6}$$

$\overline{\mathrm{Info}}(\mathbf{x}_i)$ can also be used to calculate another weight for each training instance, and a base classifier h_1^* can be constructed based on the newly weighted bootstrap samples obtained from the original training data set.

Definitions of $\mathrm{Info}(\mathbf{x}_i)$ and $\overline{\mathrm{Info}}(\mathbf{x}_i)$ imply that if the weighted data generated based on $\mathrm{Info}(\mathbf{x}_i)$ are difficult labeled training data, then the weighted data generated based on $\overline{\mathrm{Info}}(\mathbf{x}_i)$ should be easily labeled ones. These two data sets focus on different parts of the original data set, so they are diverse data sets and h_1 and h_1^* are diverse classifiers.

After h_1 and h_1^* have been trained, we update $\mathrm{Info}(\mathbf{x}_i)$ and $\overline{\mathrm{Info}}(\mathbf{x}_i)$ so as to generate new training data sets.

4. Algorithm description

A weighting scheme is a group of weights over the training data. The main idea of our algorithm is to find a set of weights used to learn different component classifiers. For each training instance $\mathbf{x}_i$, we calculate $\mathrm{Info}(\mathbf{x}_i)$ and $\overline{\mathrm{Info}}(\mathbf{x}_i)$, and obtain two groups of data $\{\mathrm{Info}(\mathbf{x}_i)|\mathbf{x}_i \in D\}$ and $\{\overline{\mathrm{Info}}(\mathbf{x}_i)|\mathbf{x}_i \in D\}$ (D is the training data set). Then we calculate two weights for each training instance as

$$w(\mathbf{x}_i) = \frac{\mathrm{Info}(\mathbf{x}_i)}{\sum_{\mathbf{x}_i \in D} \mathrm{Info}(\mathbf{x_i})} \tag{7}$$

$$\overline{w}(\mathbf{x}_i) = \frac{\overline{\mathrm{Info}}(\mathbf{x}_i)}{\sum_{\mathbf{x}_i \in D} \overline{\mathrm{Info}}(\mathbf{x}_i)} \tag{8}$$

We form two weights $W_1 = \{w(\mathbf{x}_i)|\forall \mathbf{x}_i \in D\}$ and $\overline{W}_1 = \{\overline{w}(\mathbf{x}_i)|\forall \mathbf{x}_i \in D\}$, which satisfy the following conditions:

$$\sum_{i=1}^{d} w(\mathbf{x}_i) = 1, \quad \forall w(\mathbf{x}_i) \in [0, 1] \tag{9}$$

$$\sum_{i=1}^{d} \overline{w}(\mathbf{x}_i) = 1, \quad \forall \overline{w}(\mathbf{x}_i) \in [0, 1] \tag{10}$$

We feed the learning algorithm h_1 and h_1^* with a training data set that consists of d examples drawn from the original data set based on the weights W_1 and $\overline{W}_1$, respectively, using the weighted bootstrap approach.

We need to calculate the weights W_{t+1} and $\overline{W}_{t+1}$ at the $(t+1)$th round in order to generate classifiers h_{t+1} and h_{t+1}^*. Given a data distribution line on which the training data are distributed according to their fuzzy information, and let the most easily labeled and most difficult labeled data locate at the left and the right end of the line, respectively, then data become gradually more and more difficult to be labeled from the left end to the right end. Data misclassified by the generated classifiers are the ones that are difficult to be labeled and should be moved to the right along the training data distribution line, and data correctly classified by the generated classifiers should be moved to the left along the training data distribution line. After the data have been moved along the distribution line, a new data distribution can be obtained. As we know, the fuzzy memberships of each training example determine its location on the distribution line, so we update the fuzzy memberships of the training data based on the classification results of the previously trained classifiers, and calculate the fuzzy information using the updated memberships.

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Table 1
The FuzzyBoost algorithm.

Given the original training data set, the base classifier r
- Using fuzzy c-means technique to fuzzy cluster the training data and get the fuzzy memberships for each training instance
- Calculating the two weights for each example based on formula (7) and (8), respectively forming weights W_1 and $\overline{W_1}$ and constructing classifier h_1^* and h_1 using the weighted bootstrap instances drawn from the original data set according to weights $\overline{W_1}$ and W_1, respectively
- for $t = 1, \ldots, r-1$
  Updating the fuzzy memberships for the data correctly classified by mvh_t (the majority vote result of classifiers $h_1 \ldots h_t$.) and mvh_t^* according to (11); updating the fuzzy memberships for the data misclassified by mvh_t and mvh_t^* based on (12)
  Using the updated fuzzy memberships to calculate the information entropy based on (5) and the weight using (7) for each sample to obtain W_{t+1}; using the updated fuzzy memberships to calculate the reciprocal of information entropy for each sample based on (6), updating the reciprocal again according to (13), and then calculating the weight using (8) for each sample to form $\overline{W}_{t+1}$; training classifiers h_{t+1} and h_{t+1}^* with weighted bootstrap samples using weights W_{t+1} and $\overline{W}_{t+1}$, respectively
- Calculating the error rate and a parameter alpha for each base classifier c_i according to $\varepsilon_i = (1/N)\sum_j I(c_i(x_j)\neq y_j)$ and $\alpha_i = \frac{1}{2}\ln(1-\varepsilon_i)/\varepsilon_i$, and outputting the final decision of the ensemble according to $C^*(x) = \text{argmax}_{y\in C}\sum_{i=1}^{r}\alpha_i I(c_i(x) = y)$

Given $\mu_{ij}(t)$ denoting the membership of $\mathbf{x}_i$ at the tth round, if $\mathbf{x}_i$ belongs to class j and is correctly classified, we update the fuzzy memberships of $\mathbf{x}_i$ as follows:

$$\begin{cases} \mu_{ij}(t+1) = \mu_{ij}(t) + \alpha(1-\mu_{ij}(t)) \\ \mu_{ik}(t+1) = \mu_{ik}(t) - \alpha\mu_{ik}(t), \quad k\neq j \end{cases} \tag{11}$$

The constant α is a reward parameter that lies in (0,1).

If $\mathbf{x}_i$ belongs to class j and is misclassified at the tth round, we update its membership as

$$\mu_{ik}(t+1) = \mu_{ik}(t) - \alpha\left(\mu_{ik}(t) - \frac{1}{c}\right) \quad (k = 1, 2, \ldots, j, \ldots, c) \tag{12}$$

According to the updating rule (11), if a training example is correctly classified, its membership to the right class increases and the memberships to all the other classes decrease. As this process goes on, the membership to the right class gradually approaches one, and the memberships to the other classes gradually approaches zero. The training example is then moved gradually to the left farthest end of the distribution line.

Updating rule (12) is performed if a training example is misclassified, its memberships gradually approaches $1/c$ after updating, and the example moves gradually to the far most right end of the distribution line.

In the above updating process, the correctly classified data move to the left side and the misclassified data move to the right side along the distribution line. This process updates the distribution of the training data along the distribution line.

New fuzzy memberships are obtained after updating the fuzzy memberships of the training data, and can be used to calculate new $\text{Info}(\mathbf{x}_i)$ and $\overline{\text{Info}}(\mathbf{x}_i)$ for each training data. As we want to make the constructed classifiers gradually focus on those not easily classified, the weights of the correctly classified data should decrease. We adopt the following approach to update $\overline{\text{Info}}(\mathbf{x}_i)$ of the data correctly classified according to the majority vote result of the t classifiers $h_1^* \ldots h_t^*$

$$\overline{\text{Info}}(\mathbf{x}_i) = \overline{\text{Info}}(\mathbf{x}_i)/(e^{I(y_i = mvh^*(\mathbf{x}_i))\cdot\overline{\text{Info}}(\mathbf{x}_i)}) \tag{13}$$

$\overline{\text{Info}}(\mathbf{x}_i)$ in the right side of (13) is calculated using (6), and y_i is the real class label of the training example $\mathbf{x}_i$. $I(*) = 1$ if $*$ is true or 0 if $*$ is false, and $mvh_t^*(\mathbf{x}_i)$ is the majority vote result of the t classifiers $h_1^* \ldots h_t^*$.

We calculate two weights for each example based on (7) and (8) at the $(t+1)$th round, and obtain two weights W_{t+1} and $\overline{W}_{t+1}$. Classifiers h_{t+1} and h_{t+1}^* can be generated, respectively, based on the weighted bootstrap instances drawn from the original data set according to weights W_{t+1} and $\overline{W}_{t+1}$.

We use the constructed base classifiers to classify a new sample, and obtain the final classification result in the same way as that of AdaBoost. We name the algorithm FuzzyBoost and describe it in Table 1.

We use the deflection approach given in formula (4) to obtain several objective functions, and get one group of fuzzy memberships by minimizing each objective function. Then we can use FuzzyBoost algorithm to generate a number

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Table 2
Deflected FuzzyBoost Algorithm (DFA).

Given the deflection number s and the base classifier number r
- Using FCM to get a group of fuzzy memberships of the objective function
- Looping s times. For each time, transforming the objective function according to (4), applying FCM to the transformed objective function, and calculating a new group of fuzzy memberships. Finally, we obtain $s+1$ groups of fuzzy memberships
- Using FuzzyBoost approach to generating $\lfloor r/(s+1)\rfloor$ classifiers based on each group of fuzzy Memberships. If $(s+1)*\lfloor r/(s+1)\rfloor < r$, we train $r - s*\lfloor r/(s+1)\rfloor$ classifiers using the finally obtained group of memberships. r base classifiers can be constructed in the end
- Calculating the error rate and a parameter alpha for each base classifier c_i according to $\varepsilon_i = (1/N)\sum_j I(c_i(x_j)\neq y_j)$ and $\alpha_i = \frac{1}{2}\ln(1-\varepsilon_i)/\varepsilon_i$, and outputting the final decision of the ensemble according to $C^*(x) = \text{argmax}_{y\in C}\sum_{i=1}^{r}\alpha_i I(c_i(x)=y)$

of base classifiers based on each group of the fuzzy memberships, and build the ensemble using all the generated classifiers. The algorithm is named Deflected FuzzyBoost Algorithm (DFA), and is described in Table 2.

5. Experimental results

Experiments on 20 benchmark data sets have been conducted to evaluate the classification performance of the constructed ensemble, and the learning capability of FuzzyBoost and DFA is to be measured by the accuracy of the generated ensemble. Performance analysis and comparison with AdaBoosting and Bagging are also given.

5.1. Data sets

Twenty data sets of varying complexity from the UCI Repository of Machine learning Databases are used. These data sets have been extensively employed for benchmark study. Detailed information about the data sets (attribute number, class number, record number) is summarized in Table 3. These data sets differ from each other in the training size, the dimension of feature space and the number of classes.

5.2. Evaluation approach

C4.5 algorithm is chosen as the base learning scheme and the performance of Bagging, AdaBoost, FuzzyBoost and DFA on each data set is compared. The performance of each ensemble classifier is evaluated using a stratified 10-fold cross validation procedure. In this cross validation procedure, the data set is partitioned randomly into 10 parts, and in each part, the class is represented in approximately the same proportions as in the full data set. One part is held out in each turn and the component classifiers are trained on the left nine-tenths. The error rate of the constructed ensemble is evaluated on the holdout set. In order to get a best estimation of the classification error rate, we averaged the results over 3 times 10-fold cross validation for each data set. The number of base classifiers is set to be 50 for all algorithms. Although this number may not be the optimal one for the algorithms, it is fair to each algorithm, because it is in general difficult to give a reliable estimate of the optimal number of base classifiers to get the best generalization accuracies.

The setup of the parameters is given as: the reward parameter $\alpha = 0.5$; the base classifier number $r = 50$; the deflection number $s = 2$; the relaxing parameters λ equals 2 and keeps fixed; the fuzziness number $m = 2$.

5.2.1. Performance comparison

For each algorithm, the mean classification error rate with its standard deviation on each data set is reported in Table 4. A lower error rate indicates higher classification accuracy. The relaxing parameter determines the steepness of the transformation function and affects the function deflection result. We set the relaxing parameter for each transformation function to be 2 based on our evaluation on all the data sets. The reward parameter determines the moving steps of the training data along the virtual distribution line and influence the classification accuracy. The relationship between the reward parameter and the classification error rate is evaluated, and experimental results show that the optimal reward parameter is different from one data set to another. There is not an optimal one for all the data sets, and most of the optimal reward parameters lie in the interval (0.25, 0.75), so we set the reward parameter to be 0.5. Given the number of the classifiers composing an ensemble, there is a compromise between the deflection number and the base classifier number. As we aim on evaluating the generalization capability of the constructed classifiers, the number of classifiers

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Table 3
Summary of data sets.

Data set	Training set size	Attribute number	Class number
Artificial	5109	7	10
Audiology	226	69	24
Automobile	205	25	7
Balance-scale	625	4	3
Breast cancer-w	699	9	2
Cean1	476	166	2
Diabetes	768	8	2
German	1000	20	2
Glass	214	9	7
Heart-statlog	270	13	2
Hepatitis	155	19	2
Horse colic	300	27	3
Ionosphere	351	34	2
Labor	57	16	2
Sonar	208	60	2
Soybean	683	35	19
Vehicle	846	18	4
Voting	435	16	2
Waveform	5000	40	3
Wine	178	13	3

generated according to each group of fuzzy memberships should not be too small, so we set the deflection number to be 2, and obtain three groups of fuzzy memberships. Given the number of the base classifier, the number of classifiers that should be generated based on each group of fuzzy memberships can be calculated. In this paper, we set the number of base classifiers to be 50, so we set the numbers of the classifiers to 16, 16 and 18 for the sequentially obtained three groups of fuzzy memberships.

We seek to build one base classifier using easy examples and another using difficult examples in order to improve the classification accuracy of the ensemble, these base classifiers may be diverse as training on different parts of the original data set. Adaboost focuses only on the hard examples but our experimental results show that the base classifiers in the ensemble focus on the easy examples really produce difference in performance. So we do experiments to show the generalization error of FuzzyBoost using: all the classifiers, only the ones that focus on the easy parts (denoted as Alle) and only the ones that focus on the hard pats of the training data set (denoted as Alld).

The winner in each data set is given in bold style in Table 4. From the results shown in Table 4, we make observations listed in Table 5.

(1) Compared with AdaBoost, FuzzyBoost wins on 14 data sets, draws on 1 data set and loses on the left 5 data sets according to the mean classification error rate. On average, we can say that FuzzyBoost outperforms AdaBoost on the 20 benchmark data sets. Further, according to the mean classification error rate, FuzzyBoost outperforms Bagging on 15 data sets and loses on 1. They draw on the other four data sets. Generally speaking, FuzzyBoost also works better than Bagging on the 20 benchmark data sets.

(2) DFA employs a deflection technique to improve the classification performance of the generated ensemble. We observe from the result that DFA outperforms AdaBoost on 13 data sets and loses to AdaBoost on seven data sets according to the mean classification error rate. Similar observations can be made based on the mean classification error rate that DFA outperforms Bagging on 17 data sets and loses to Bagging on three data sets. Similar conclusions can be made that DFA generally outperforms AdaBoost and Bagging on the 20 data sets.

(3) DFA outperforms FuzzyBoost in eight data sets, loses on six data sets and draw on the left six data sets according to the results of the mean classification error rate.

(4) Based on the results of the mean classification error rate, DFA performs best 9 times and performs worst 1 time on all the 20 data sets, FuzzyBoost does best 5 times and worst 2 time, AdaBoost does best 5 times and worst 8 times, and Bagging outperforms the other three algorithms only on 1 data set and does worst 12 times.

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Table 4
The mean error rates (expressed in %) and standard deviation of classification on 20 data sets.

Data set	FuzzyBoost	DFA	AdaBoost	Bagging	Alle	Alld
Artificial	39.09 ± 0.28	**38.91** ± 0.28	39.61 ± 0.24	39.37 ± 0.65	40.28 ± 0.28	40.64 ± 0.28
Audiology	13.71 ± 0.11	**13.27** ± 0.11	15.92 ± 0.11	19.17 ± 0.33	15.60 ± 0.11	15.60 ± 0.11
Automobile	**13.66** ± 020	14.63 ± 0.20	13.71 ± 0.18	15.12 ± 0.56	16.59 ± 0.22	15.66 ± 0.20
Balance-scale	19.22 ± 0.36	18.24 ± 0.35	23.30 ± 0.39	**17.76** ± 0.28	19.92 ± 0.34	22.08 ± 0.38
Breast cancer-w	3.43 ± 0.18	3.72 ± 0.19	**3.43** ± 0.56	3.81 ± 0.16	4.86 ± 0.20	4.58 ± 0.19
Cean1	10.08 ± 0.31	10.08 ± 0.32	**7.98** ± 0.27	11.20 ± 0.27	10.50 ± 0.32	7.98 ± 0.28
Diabetes	**23.18** ± 0.50	24.35 ± 0.49	25.43 ± 0.47	23.44 ± 0.40	25.22 ± 0.49	25.22 ± 0.49
German	**24.1** ± 0.49	24.20 ± 0.49	26.00 ± 0.50	25.43 ± 0.41	25.20 ± 0.50	24.90 ± 0.49
Glass	24.30 ± 0.25	23.83 ± 0.26	**21.49** ± 0.25	26.48 ± 0.23	25.23 ± 0.27	24.83 ± 0.26
Heart-statlog	**18.15** ± 0.43	18.15 ± 0.42	21.85 ± 0.44	18.27 ± 0.36	19.78 ± 0.42	18.89 ± 0.43
Hepatitis	15.48 ± 0.42	**14.84** ± 0.39	16.13 ± 0.39	19.78 ± 0.36	17.84 ± 0.38	18.55 ± 0.36
Horse colic	27.42 ± 0.42	**26.76** ± 0.42	27.98 ± 0.42	27.09 ± 0.35	29.10 ± 0.43	28.42 ± 0.42
Ionosphere	5.84 ± 0.26	**5.84** ± 0.25	5.89 ± 0.24	7.88 ± 0.23	6.84 ± 0.26	6.55 ± 0.26
Labor	14.04 ± 0.35	14.04 ± 0.37	**9.94** ± 0.32	12.87 ± 0.35	15.79 ± 0.39	17.54 ± 0.41
Sonar	17.79 ± 0.41	17.79 ± 0.41	**15.22** ± 0.38	19.75 ± 0.34	19.71 ± 0.44	18.27 ± 0.41
Soybean	6.59 ± 0.08	**6.03** ± 0.08	7.17 ± 0.22	8.87 ± .022	7.88 ± 0.08	6.30 ± 0.08
Vehicle	24.22 ± 0.34	**22.10** ± 033	22.38 ± 0.33	24.22 ± 0.28	22.81 ± 0.33	22.93 ± 0.34
Voting	3.68 ± 0.19	**3.68** ± 0.21	4.90 ± 0.63	3.72 ± 0.17	4.37 ± 0.20	4.37 ± 0.21
Waveform	**15.76** ± 32	16.62 ± 0.32	15.85 ± 0.32	17.02 ± 0.84	16.66 ± 0.33	16.20 ± 0.32
Wine	2.81 ± 0.14	**2.27** ± 0.15	4.49 ± 0.15	2.62 ± 0.12	3.37 ± 0.15	3.37 ± 0.15

Table 5
Performance win/lose comparisons between two algorithms on 20 data sets.

DFA/FuzzyBoost (Win/lose/draw)	FuzzyBoost/AdaBoost (Win/lose/draw)	FuzzyBoost/Bagging (Win/lose/draw)	DFA/AdaBoost (Win/lose/draw)	DFA/Bagging (Win/lose/draw)	FuzzyBoost/Alle (Win/lose/draw)	FuzzyBoost/Alld (Win/lose/draw)
8/6/6	14/5/1	15/4/1	13/7/0	17/3/0	19/0/1	18/0/2

From the above observations, we may conclude that DFA does best than the other three algorithms on most of the 20 data sets. Even it performs worst on 1 data set (Labor), it still draws with Fuzzyboost. We see that Labor is a data set with small size of data and large attribute number.

From the results shown in Table 4, we also make observations that Fuzzyboost wins almost all the times on the 20 data sets compared with Alle and Alld. It loses to Alle on data set Vehicle, and loses to Alld on data sets Vehicle and Soybean. This may be shown that the addition of the "easy" classifiers is helpful to the classification accuracy of the ensemble.

5.2.2. Weight initialization evaluation

Fuzzy clustering approach is employed to estimate the classification difficulty of labeled instances. Adaboost focuses on the "difficult" instances gradually as the iteration number increases, but Fuzzyboost can focus on the "difficult" instances at the start-up iteration. In order to evaluate whether this initialization approach is helpful to reduce the generalization error, we plot the evolution of the classification errors with respect to the base classifier number for DFA, FuzzyBoost and FuzzyBoost with initializing the weights to $1/d$ (d is the training data number). Experiments are performed on six randomly selected data sets, and the evolution curves are shown in Fig. 1.

Figure one shows that, when the number of the base classifiers is small, the classification error rates on all the 10 data sets are high. The error rates change with the number of the base classifiers, and the whole changing trend is that they decrease as the number of base classifiers increases. It is known that if the number of classifiers increases to some extent, the error rate will increase. We use 50 base classifiers in our experiments, which may not be an optimal choice. The figure also shows that the evolution speeds of the error rates for these three algorithms on different data sets are different, but they seem to be similar on a same data set. On each data set, the error rate of FuzzyBoost initialized the

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

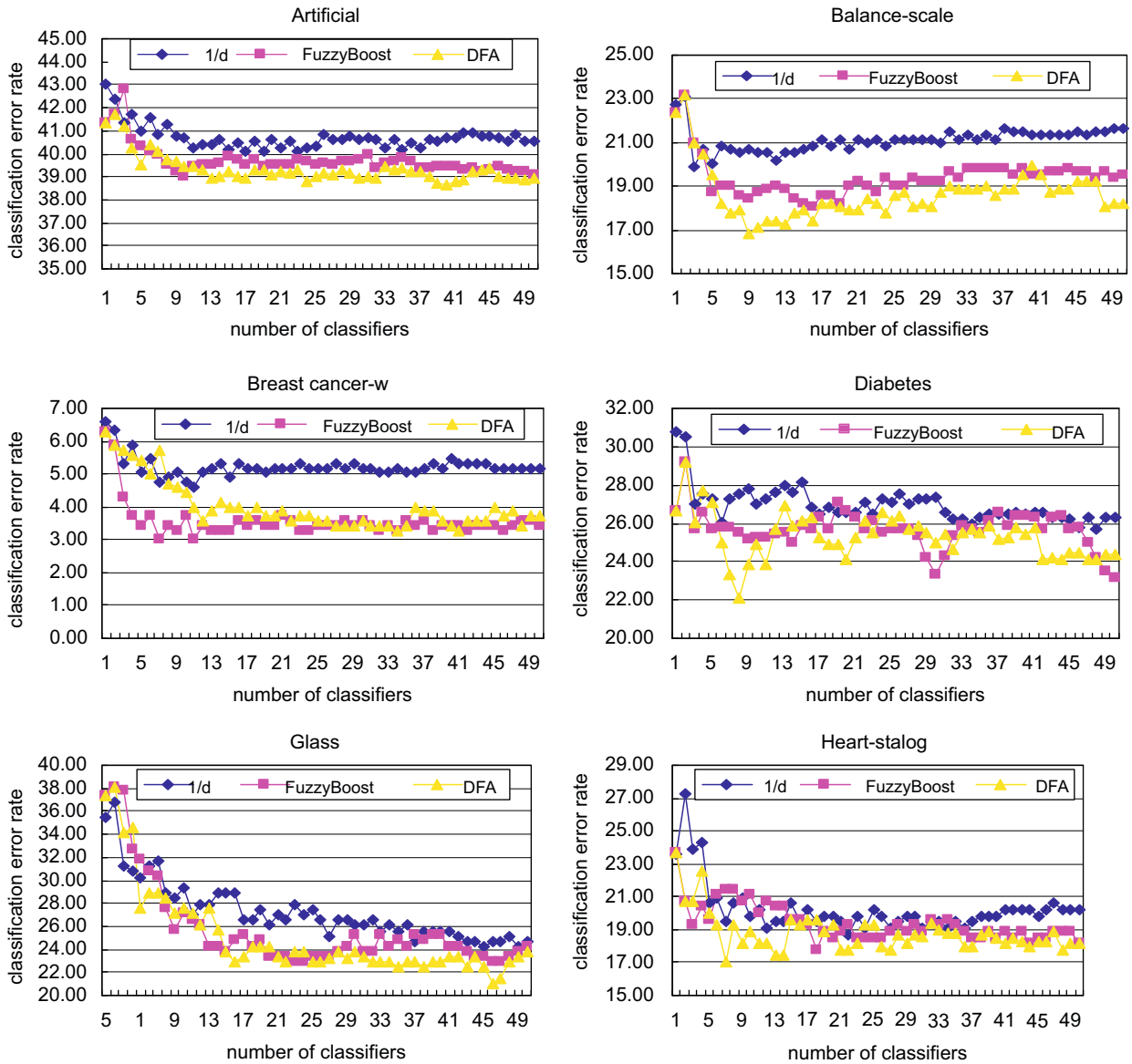

Fig. 1. Classification error rate with respect to the number of base classifiers.

weight with $1/d$ is bigger than that of FuzzyBoost and DFA on most occasions with different number of classifiers, and this may be caused by the diversity difference of the base classifiers.

5.2.3. The influence of the parameters

As mentioned above, the reward parameter influences the performance of FuzzyBoost and DFA. We fix the deflection parameter and evaluate the influence of the reward parameter on the classification performance of the ensemble by implementing experiments on the 20 data sets and calculating the mean classification error rates. The reward parameter is set to be 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, respectively, and the deflection number is fixed to be 0, 1, 2, 3, 4, 5 each time, then we get 45 mean error rates for each data set. The results on ten data sets are given in Table 6.

The experimental results given in Table 6 show the joint influence of the reward parameter and the deflection number on the ensemble performance. Results obtained under different deflection numbers reveal that when the reward parameter and the deflection number are properly chosen for a specific data set, the classification the ensemble performance can

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Table 6
Mean error rates under different values of the reward parameter and deflection number (values in bracket are the optimal ones).

	0	1	2	3	4	5	0	1	2	3	4	5
	Artificial						Audiology					
0.1	39.13	39.21	39.48	38.87	38.64	39.03	15.93	17.26	18.14	18.58	19.47	19.47
0.2	39.56	39.15	39.30	39.09	38.77	**38.60**	14.60	16.37	17.26	18.58	19.03	18.14
0.3	39.36	39.26	38.91	39.36	39.22	39.38	14.16	15.04	15.93	15.93	17.70	17.70
0.4	39.36	38.87	38.76	39.44	39.09	39.11	13.27	13.27	14.60	14.16	16.37	17.26
0.5	39.09	39.50	38.91	38.93	39.44	38.89	13.71	13.72	13.27	15.04	15.04	15.49
0.6	39.58	39.03	39.01	38.81	39.40	39.50	14.16	13.72	13.72	14.16	14.16	16.37
0.7	39.09	39.54	39.11	39.22	39.07	39.36	**12.83**	13.72	14.60	14.16	14.60	14.16
0.8	39.26	39.79	39.19	39.42	38.91	39.01	13.27	14.60	13.72	14.16	15.04	15.04
0.9	39.44	39.24	39.66	39.64	38.81	39.56	13.27	15.04	13.27	13.27	14.60	14.60
	Automobile						Balance-scale					
0.1	14.63	16.10	15.12	15.12	16.10	15.12	18.40	17.92	16.96	16.96	**16.48**	16.64
0.2	14.15	13.17	13.17	14.15	17.07	17.07	19.20	18.72	17.76	17.60	17.12	17.28
0.3	14.15	15.61	14.63	14.63	15.12	14.15	20.00	18.88	19.04	17.76	17.60	17.28
0.4	14.63	13.17	14.63	15.12	15.12	15.12	19.68	19.84	19.52	18.88	18.24	17.76
0.5	13.66	13.66	14.63	14.63	14.63	15.12	19.22	19.68	18.24	18.56	18.24	17.44
0.6	14.63	15.12	15.61	15.12	13.66	15.12	19.36	19.84	18.88	19.20	18.88	18.40
0.7	**12.68**	**12.68**	14.63	14.15	14.63	15.61	20.00	19.84	19.36	19.04	19.20	17.60
0.8	13.17	13.17	14.63	13.17	14.63	13.17	19.36	19.84	18.72	19.20	19.20	17.60
0.9	15.61	15.12	14.63	14.15	15.61	15.12	18.88	19.20	18.56	18.56	18.72	18.24
	Breast cancer-w						Clean1					
0.1	3.72	3.72	4.15	4.01	4.15	3.86	9.24	9.03	9.66	9.45	8.61	9.03
0.2	3.43	3.86	3.86	3.86	3.58	3.72	8.82	9.03	8.40	9.03	8.82	8.19
0.3	4.15	3.86	3.72	4.15	3.72	4.15	**7.98**	9.66	9.87	9.03	8.82	9.03
0.4	3.43	3.43	3.86	4.01	4.01	3.86	9.45	8.19	**7.98**	8.61	8.82	10.08
0.5	3.43	3.43	3.72	3.72	3.86	3.72	10.08	10.08	10.08	8.61	9.03	8.61
0.6	4.01	3.86	3.72	3.58	3.72	3.86	9.45	10.08	9.66	9.03	8.61	8.40
0.7	**3.29**	3.43	4.01	3.72	**3.29**	4.01	9.45	9.45	**7.98**	8.82	9.87	9.03
0.8	4.01	3.43	3.86	4.29	3.72	4.01	10.92	10.29	9.45	8.61	8.61	9.03
0.9	3.86	3.58	3.72	4.29	4.15	3.86	9.66	9.87	9.24	10.50	10.08	9.03
	Diabetes						German					
0.1	24.35	24.35	23.70	24.87	25.39	24.61	25.40	24.90	25.20	24.80	25.50	25.10
0.2	24.35	23.96	24.35	24.74	25.13	24.74	25.40	25.90	24.90	25.00	25.60	25.70
0.3	24.22	25.00	24.61	25.00	24.35	23.83	24.90	25.80	26.00	25.80	24.70	26.00
0.4	24.48	23.96	24.35	24.87	25.78	24.48	25.60	25.50	25.40	25.80	25.70	26.40
0.5	**23.18**	25.91	24.35	25.65	25.26	25.00	24.10	24.00	24.20	24.50	25.10	25.10
0.6	25.52	25.00	24.22	25.91	25.13	**23.70**	23.80	24.60	24.60	25.20	23.60	25.00
0.7	23.96	24.74	24.35	23.96	25.00	25.65	24.00	23.20	23.80	24.90	24.70	25.20
0.8	24.22	25.13	24.22	25.26	25.39	24.87	24.40	23.80	23.60	23.90	24.60	**22.00**
0.9	24.22	25.91	24.35	23.83	25.52	25.00	22.80	22.80	23.30	23.30	23.80	23.60
	Glass						Heart-statlog					
0.1	23.83	24.30	24.77	24.77	22.90	24.30	18.15	18.15	17.78	18.15	18.89	19.26
0.2	24.30	25.70	23.36	22.90	24.77	23.83	18.15	18.52	18.15	18.15	18.89	19.63
0.3	22.90	22.90	24.77	23.83	22.43	24.30	18.52	18.89	20.00	18.89	18.89	18.52
0.4	21.96	22.43	21.03	22.43	24.30	23.36	18.89	17.41	19.26	18.52	18.89	18.52
0.5	24.30	21.96	23.83	21.96	22.43	24.30	18.15	17.04	18.15	18.15	20.37	18.89
0.6	24.30	21.96	25.23	23.83	23.83	21.96	18.89	18.89	19.26	18.89	18.89	18.52
0.7	21.96	21.03	22.90	21.50	22.43	**20.56**	20.00	17.78	19.26	19.26	18.89	19.63
0.8	22.43	22.90	23.36	23.83	24.30	21.03	20.00	20.00	18.15	18.52	18.52	19.26
0.9	23.83	21.96	23.36	23.83	21.96	21.96	18.15	**16.67**	19.26	18.52	18.89	17.41

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Table 7
The mean error rate (expressed in %) comparison of DFA-ORD with FuzzyBoost, DFA, Fuzzyboost, AdaBoost and Bagging (values in bracket are the optimal alpha and deflection number).

Data set	DFA-ORD (α,s)	Fuzzyboost	DFA	AdaBoost	Bagging
Artificial	**38.60** $\pm$ 0.27(0.2, 5)	39.09 $\pm$ 0.28	38.91 $\pm$ 0.28	39.61 $\pm$ 0.24	39.37 $\pm$ 0.65
Audiology	**12.83** $\pm$ 0.10(0.7, 0)	13.71 $\pm$ 0.11	13.27 $\pm$ 0.11	15.92 $\pm$ 0.11	19.17 $\pm$ 0.33
Automobile	**12.68** $\pm$ 0.19(0.7, 1)	13.66 $\pm$ 020	14.63 $\pm$ 0.20	13.71 $\pm$ 0.18	15.12 $\pm$ 0.56
Balance-scale	**16.48** $\pm$ 0.33(0.1, 4)	19.22 $\pm$ 0.36	18.24 $\pm$ 0.35	23.30 $\pm$ 0.39	17.76 $\pm$ 0.28
Breast cancer-w	**3.29** $\pm$ 0.18(0.7, 0)	3.43 $\pm$ 0.18	3.72 $\pm$ 0.19	3.43 $\pm$ 0.56	3.81 $\pm$ 0.16
Clean1	**7.98** $\pm$ 0.28(0.3, 0)	10.08 $\pm$ 0.31	10.08 $\pm$ 0.32	**7.98** $\pm$ 0.27	11.20 $\pm$ 0.27
Diabetes	**23.17** $\pm$ 0.50(0.5, 0)	23.18 $\pm$ 0.50	24.35 $\pm$ 0.49	25.43 $\pm$ 0.47	23.44 $\pm$ 0.40
German	**22** $\pm$ 0.47(0.8, 5)	24.1 $\pm$ 0.49	24.20 $\pm$ 0.49	26.00 $\pm$ 0.50	25.43 $\pm$ 0.41
Glass	**20.56** $\pm$ 0.24(0.7, 5)	24.30 $\pm$ 0.25	23.83 $\pm$ 0.26	21.49 $\pm$ 0.25	26.48 $\pm$ 0.23
Heart-statlog	**16.67** $\pm$ 0.41(0.9, 1)	18.15 $\pm$ 0.43	18.15 $\pm$ 0.42	21.85 $\pm$ 0.44	18.27 $\pm$ 0.36
Hepatitis	**13.54** $\pm$ 0.36(0.9, 1)	15.48 $\pm$ 0.42	14.84 $\pm$ 0.39	16.13 $\pm$ 0.39	19.78 $\pm$ 0.36
Horse colic	**24.75** $\pm$ 0.40(0.9, 0)	27.42 $\pm$ 0.42	26.76 $\pm$ 0.42	27.98 $\pm$ 0.42	27.09 $\pm$ 0.35
Ionosphere	**5.13** $\pm$ 0.23(0.3, 0)	5.84 $\pm$ 0.26	5.84 $\pm$ 0.25	5.89 $\pm$ 0.24	7.88 $\pm$ 0.23
Labor	10.53 $\pm$ 0.32(0.7, 4)	14.04 $\pm$ 0.35	14.04 $\pm$ 0.37	**9.94** $\pm$ 0.32	12.87 $\pm$ 0.35
Sonar	15.87 $\pm$ 0.39(0.7, 2)	17.79 $\pm$ 0.41	17.79 $\pm$ 0.41	**15.22** $\pm$ 0.38	19.75 $\pm$ 0.34
Soybean	**5.86** $\pm$ 0.08(0.9, 2)	6.59 $\pm$ 0.08	6.03 $\pm$ 0.08	7.17 $\pm$ 0.22	8.87 $\pm$ .022
Vehicle	**21.16** $\pm$ 0.32(0.4, 0)	24.22 $\pm$ 0.34	22.10 $\pm$ 033	22.38 $\pm$ 0.33	24.22 $\pm$ 0.28
Voting	**3.22** $\pm$ 0.18(0.8, 3)	3.68 $\pm$ 0.19	3.68 $\pm$ 0.21	4.90 $\pm$ 0.63	3.72 $\pm$ 0.17
Waveform	**15.38** $\pm$ 0.32(0.9, 0)	15.76 $\pm$ 32	16.62 $\pm$ 0.32	15.85 $\pm$ 0.32	17.02 $\pm$ 0.84
Wine	**2.25** $\pm$ 0.12(0.8, 0)	2.81 $\pm$ 0.14	2.27 $\pm$ 0.15	4.49 $\pm$ 0.15	2.62 $\pm$ 0.12

be optimal. We name the algorithm DFA with the Optimal Reward parameter and Deflection number DFA-ORD. The mean error rate comparison of DFA-ORD with other algorithms is described in Table 7.

Some observations can be made from the results in Table 7: DFA-ORD does best on 18 of the 20 data sets except on data set Labor and Sonar with small size of training instance and large attribute number. The results show that the classification performance of the ensemble can be improved through concurrently selecting an optimal reward parameter and a deflection number for a specific data set.

We also examine whether the deflection approach improves diversity among base classifiers in the ensemble. Q-statistics is one of the averaged pairwise measures used to measure diversity among classifiers in an ensemble, for smaller Q values, better diversity can be obtained. We employ the formulae four from Kuncheva et al.'s paper [32], and the results are given in Table 8. We observe from the results that the Q values of DFA are smaller than the Q values of Fuzzyboost on 16 data sets, and they are the same on two data sets. This shows that the deflection approach improves the diversity among the base classifiers.

6. Conclusions and discussions

We have presented a novel ensemble creation approach that incorporates a deflection technique based on fuzzy clustering the training data. The approach can construct accurate and diverse base classifiers using weighted training data. A concept of information entropy is defined and used to calculate the weights of the training data. Different groups of weightings are obtained by a deflection technique and adjusting the fuzzy memberships is executed according to proposed updating rules. As fuzzy c-means can only learn one local optimal solution to its objective function, we adopt a deflection technique to learn more than one optimum of the objective function and obtain a number of groups of fuzzy memberships. The main contributions of this work are as follows: (1) we adopt FCM to cluster the training data in order to get the distribution attribute of the training data, and propose fuzzy entropy to evaluate the labeling difficulty of each training example. (2) FCM converges only to one local optima of the objective function, and each local optimal point corresponds to an optimal fuzzy labeling of the training data. We employ a deflection technique to learn different optimal points of the objective function, and get different optimal fuzziness of the training data. (3) Experimental results show that there is an optimal learning rate for each data set. The optimal learning rate is different from one data set to another, and it reveals the internal closeness of the data.

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

Table 8
Q statistics values of Fuzzyboost and DFA.

Data set	DFA	Fuzzyboost
Artificial	0.28	0.31
Audiology	0.39	0.44
Automobile	0.22	0.22
Balance-scale	0.56	0.62
Breast cancer-w	0.42	0.41
Clean1	0.07	0.09
Diabetes	0.26	0.28
German	0.28	0.35
Glass	0.18	0.17
Heart-statlog	0.28	0.34
Hepatitis	0.33	0.38
Horsecolic	0.34	0.39
Ionosphere	0.16	0.17
Labor	0.13	0.15
Sonar	0.04	0.05
Soybean	0.47	0.61
Vehicle	0.23	0.24
Vote	0.69	0.87
Waveform	0.16	0.21
Wine	−0.32	−0.32

It is worth noting that fuzzy c-means is commonly used in clustering problems, but we use it to cluster the training data in order to obtain the data distribution in classes. Our approach is time consuming compared with AdaBoost and Bagging in constructing the component classifiers, this is caused by the fuzzy c-means method.

The results of the experiments conducted on the benchmark data sets demonstrate that the proposed approaches perform better than those of AdaBoost and Bagging, and the result comparison reported in Table 7 show that the proposed methods are more effective when optimal reward parameter and deflection number are selected. The experimental results and the observations suggest that selection of the parameters need further discussion and theoretical analysis. We think that the perturbation of the parameters by running Fuzzyboost or DFA several times with different parameters setting and then combining the learned component classifiers may not permit to obtain better performance, as the perturbation of the parameters will balance the instance selection.

Even though FuzzyBoost introduces new limitation because fuzzy clustering depends on the distribution of data heavily, and fuzzy clustering may not learn the correct data distribution, it still works well on most of the tested data sets. This shows us that consideration of the training data distribution characteristic is helpful for classification problems. Further research and theoretical analysis should be done to reveal what kind of clustering approaches is optimal to characterize the distribution of the training data. We only test the base learning algorithm C4.5 in this work, and other base learning algorithms will be tested in our future work.

Acknowledgements

Many thanks for the anonymous reviewers for their helpful comments. This research is partially supported by the National Natural Science Foundation of China (No. 90612003) and the Science and Technology Projects of Shandong Province, China (Nos. 2007ZZ17, 2008GG10001015, 2008B0026, J09LG02).

References

[1] L. Nanni, A. Lumini, Ensemblator: an ensemble of classifiers for reliable classification of biological data, Pattern Recognition Letters 28 (5) (2007) 622–630.

[2] L. Nanni, A. Lumini, Ensemble generation and feature selection for the identification of students with learning disabilities, Expert Systems with Applications 3 (2) (2009) 3896–3900.

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013

[3] N. Garcia-Pedrajas, C. Hervas-Martinez, D. Ortiz-Boyer, Cooperative coevolution of artificial neural network ensembles for pattern classification, IEEE Transactions on Evolutionary Computation 9 (3) (2005) 271–302.
[4] L. Nanni, A novel ensemble of classifiers for protein fold recognition, Neurocomputing 69 (2006) 2434–2437.
[5] W.B. Langdon, S.J. Barrett, B.F. Buxton, Combining decision trees and neural networks for drug discovery, in: Genetic Programming, Proc. 5th European Conf., EuroGP 2002, Kinsale, Ireland, 2002, pp. 60–70.
[6] D. Maio, L. Nanni, Multihashing, human authentication featuring biometrics data and tokenized random number: a case study FVC2004, Neurocomputing 69 (2005) 242–249.
[7] Guest editorial, Diversity in multiple classifier systems, Information Fusion 6 (2005) 3–4.
[8] K. Woods, W. Kegelmeyer, K. Bowyer, Combination of multiple classifiers using local accuracy estimates, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (1997) 405–410.
[9] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.
[10] Y. Freund, R.E. Schapire, Experiments with a new boosting algorithm, in: Proc. 13th Internat. Conf. on Machine Learning, Morgan Kaufmann, Los Altos, CA, 1996, pp. 148–156.
[11] L. Nanni, A. Lumini, FuzzyBagging: a novel ensemble of classifiers, Pattern Recognition 39 (2006) 488–490.
[12] R. Bryll, R. Gutierrez-Osuna, F. Quek, Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets, Pattern Recognition 36 (6) (2003) 1291–1302.
[13] Y. Kim, W.N. Street, F. Menczer, Optimal ensemble construction via meta-evolutionary ensembles, Expert Systems with Applications 30 (2006) 705–714.
[14] L. Rokach, Genetic algorithm-based feature set partitioning for classification problems, Pattern Recognition 41 (2008) 1676–1700.
[15] L. Breiman, Randomizing outputs to increase prediction accuracy, Technical Report 518, Statistics Department, University of California, 1998. URL ⟨http://www.boosting.org/papers/Bre98.pdf⟩.
[16] G. Martínez-Mu noz, A. Suárez, Switching class labels to generate classification ensembles, Pattern Recognition 38 (2005) 1483–1494.
[17] N. García-Pedrajas, C. García-Osorio, C. Fyfe, Nonlinear boosting projections for ensemble construction, Journal of Machine Learning Research 8 (2007) 1–33.
[18] G. Brown, J. Wyatt, R. Harris, X. Yao, Diversity creation methods: a survey and categorization, Information Fusion 6 (2005) 5–20.
[19] T. Hothorn, B. Lausen, Double-bagging: combining classifiers by bootstrap aggregation, Pattern Recognition 36 (6) (2003) 1303–1309.
[20] G. Martínez-Mu noz, A. Suárez, Using all data to generate decision tree ensembles, IEEE Transactions on Systems, Man and Cybernetics C 34 (4) (2004) 393–397.
[21] E. Bauer, R. Kohavi, An empirical comparison of voting classification algorithms: bagging, boosting and variants, Machine Learning 36 (1–2) (1999) 105–139.
[22] N.C. Oza, Boosting with averaged weight vectors, in: Proc. Internat. Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science, Vol. 2709, Springer, Guildford, Surrey, 2003, pp. 15–24.
[23] X. Wang, H. Wang, Classification by evolutionary ensembles, Pattern Recognition 39 (2006) 595–607.
[24] C.-X. Zhang, J.-S. Zhang, A local boosting algorithm for solving classification problems, Computational Statistics and Data Analysis 52 (2008) 1928–1941.
[25] G.I. Webb, Multiboosting: a technique for combining boosting and wagging, Machine Learning 40 (2) (2000) 159–196.
[26] N. Ueda, R. Nakano, Generalization error of ensemble estimators, in: Proc. Internat. Conf. on Neural Networks, 1996, pp. 90–95.
[27] J.J. Rodriguez, L.I. Kuncheva, C.J. Alonso, Rotation Forest: a new classifier ensemble method, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (10) (2006) 1619–1630.
[28] C.X. Zhang, J.-S. Zhang, RotBoost: a technique for combining Rotation Forest and AdaBoost, Pattern Recognition Letters 29 (10) (2008) 1524–1536.
[29] C.L. Blake, C.J. Merz, UCI repository of machine learning databases, 1998.
[30] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum, New York, 1981.
[31] R.D. Mckelvey, A Liapunov function for Nash equilibria, Technical Report, California, Institute of Technical Report, California Institute of Technology, 1991.
[32] L.I. Kuncheva, C.J. Whitaker, Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy, Machine Learning 51 (2003) 181–207.

Please cite this article as: H. Zhang, J. Lu, Creating ensembles of classifiers via fuzzy clustering and deflection, Fuzzy Sets and Systems (2009), doi:10.1016/j.fss.2009.11.013