'Fuzzy' vs 'Non-fuzzy' in Combining Classifiers Designed by Boosting

Ludmila I. Kuncheva

Abstract— **Boosting is recognized as one of the most successful techniques for generating classifier ensembles. Typically, the classifier outputs are combined by the weighted majority vote. The purpose of this study is to demonstrate the advantages of some fuzzy combination methods for ensembles of classifiers designed by Boosting. We ran 2-fold cross-validation experiments on 6 benchmark data sets to compare the fuzzy and non-fuzzy combination methods. On the "fuzzy side" we used the fuzzy integral and the decision templates with different similarity measures. On the "non-fuzzy side" we tried simple combiners such as the majority vote, minimum, maximum, average, product, and the Naive Bayes combination. Surprisingly, the minimum, maximum, average and product, which have been reported elsewhere to work very well on a variety of problems, appeared to be inadequate for our task. Thus the real contest was among the fuzzy combination methods on the one hand, and the weighted majority vote, the simple majority vote, and the Naive Bayes combiner, on the other hand. In our experiments, the fuzzy methods performed consistently better than the nonfuzzy methods. The weighted majority vote showed a stable performance, though slightly inferior to the performance of the fuzzy combiners. The majority vote and the Naive Bayes combiners had erratic behavior, ranging from the best to the worst contestants for different data sets.**

Keywords— **Classifier combination; Ensembles of classifiers created by Boosting; Adaboost; Fuzzy integral; Decision templates; Weighted Majority Vote; Naive Bayes.**

I. Introduction

By combining the outputs of a team of classifiers, we aim at a more accurate decision than that of the single best member of the team. Many combination methods and algorithms have been developed, including methods based on fuzzy sets [14]. However, treating combining classifiers as a branch of *statistical pattern recognition* sometimes brings about an unwelcome attitude towards using fuzzy combiners. The purpose of this study is to examine experimentally how useful fuzzy combiners are by a comparison with popular non-fuzzy combiners. We look at classifier ensembles generated by Boosting, which is recognized as one of the most successful algorithms for creating classifier ensembles [3, 7, 10, 20]. The ensemble is constructed incrementally, the subsequent classifiers focusing on those objects in the data set, which appeared to be "difficult" for the previous member of the ensemble. The presumption is that this strategy introduces diversity in the ensemble, and therefore enhances the performance.

The text is organized as follows. Section II introduces the formalism of combining classifiers and the non-fuzzy combination methods: majority vote, weighted majority vote (the standard choice for the Boosting algorithm), minimum, maximum, average, product, and the Naive Bayes combiner. The Boosting algorithm for generating an ensemble is also explained there. The "fuzzy competitors" are presented in Section III: Fuzzy Integral [4–6, 11, 22, 23] and Decision Templates (DTs) [15]. Section IV contains the experimental set up and the results. We offer some conclusions in Section V.

II. Classifier combination: non-fuzzy

Let $\mathcal{D} = \{D_1, D_2, \ldots, D_L\}$ be a set of trained classifiers (called also ensemble, team, pool, etc.), and $\Omega = \{\omega_1, \ldots, \omega_c\}$ be a set of class labels. Each classifier gets as its input a feature vector $\mathbf{x} \in \Re^n$ and assigns it to a class label from Ω, i.e., $D_i : \Re^n \to \Omega$, or equivalently, $D_i(\mathbf{x}) \in \Omega$, $i = 1, \ldots c$. Alternatively, the classifier output can be formed as a c-dimensional vector

$$D_i(\mathbf{x}) = [d_{i,1}(\mathbf{x}), \ldots, d_{i,c}(\mathbf{x})]^T, \qquad (1)$$

where $d_{i,j}(\mathbf{x})$ is the degree of "support" given by classifier D_i to the hypothesis that $\mathbf{x}$ comes from class ω_j. Most often $d_{i,j}(\mathbf{x})$ is an estimate of the posterior probability $P(\omega_i|\mathbf{x})$. It is convenient to organize the

output of all L classifiers in a *decision profile* [15]

$$DP(\mathbf{x}) = \begin{bmatrix} d_{1,1}(\mathbf{x}) & \dots & d_{1,j}(\mathbf{x}) & \dots & d_{1,c}(\mathbf{x}) \\ \dots & & & & \\ d_{i,1}(\mathbf{x}) & \dots & d_{i,j}(\mathbf{x}) & \dots & d_{i,c}(\mathbf{x}) \\ \dots & & & & \\ d_{L,1}(\mathbf{x}) & \dots & d_{L,j}(\mathbf{x}) & \dots & d_{L,c}(\mathbf{x}) \end{bmatrix}. \quad (2)$$

Thus, the output of classifier D_i is the i-th row of the decision profile, and the support for class ω_j is the jth column. Without loss of generality we can restrict $d_{i,j}(\mathbf{x})$ within the interval $[0,1]$, $i = 1,\dots,L$, $j = 1,\dots,c$, and call the classifier outputs "soft labels". Combining classifiers means to find a class label for $\mathbf{x}$ based on the L classifier outputs. We look for a vector with c final degrees of support for the classes as a soft label for $\mathbf{x}$, denoted

$$D(\mathbf{x}) = [\mu_1(\mathbf{x}),\dots,\mu_c(\mathbf{x})]^T. \quad (3)$$

If a single (crisp) class label of $\mathbf{x}$ is needed, we use the maximum membership rule: Assign $\mathbf{x}$ to class ω_s iff,

$$\mu_s(\mathbf{x}) \geq \mu_t(\mathbf{x}), \forall t = 1,\dots,c. \quad (4)$$

Ties are resolved arbitrarily. The minimum-error classifier is recovered from (4) when $\mu_i(\mathbf{x}) = P(\omega_i|\mathbf{x})$.

To train the classifiers and the combiners we have a labeled data set $\mathbf{Z} = \{\mathbf{z}_1,\dots,\mathbf{z}_N\}$, $\mathbf{z}_t \in \Re^n$, called the training set. The basic (non-fuzzy) classifier combination methods are described below.

A. Non-trainable combiners

In this subsection we detail the combiners that are ready to operate as soon as the classifiers are trained, i.e., they do not require any further training of the ensemble as a whole.

The **Majority vote (MAJ)** assigns $\mathbf{x}$ to the class label most represented among the (crisp) classifier outputs. To derive a formal expression, assume that the *label* outputs of the classifiers are given as c-dimensional binary vectors $[d_{i,1},\dots,d_{i,c}]^T \in \{0,1\}^c$, $i = 1,\dots,L$, where $d_{i,j} = 1$ if D_i labels $\mathbf{x}$ in ω_j, and 0, otherwise. The *plurality vote* will pick class ω_k if

$$\sum_{i=1}^{L} d_{i,k} = \max_{j=1}^{c} \sum_{i=1}^{L} d_{i,j}. \quad (5)$$

Ties are resolved arbitrarily. This rule is often called in the literature *the majority vote*. It will indeed coincide with the simple majority (50% of the votes +1) in the case of two classes ($c = 2$). Various studies are devoted to the majority vote for classifier combination [1, 2, 16, 17, 19], etc.

The remaining simple combination methods require soft labels. The **Minimum** simple combiner operates by taking the minimum in each column thereby forming the vector $D(\mathbf{x}) = [\mu_1(\mathbf{x}),\dots,\mu_c(\mathbf{x})]^T$ as

$$\mu_j(\mathbf{x}) = \mathcal{F}\ (d_{1,j}(\mathbf{x}),\dots,d_{L,j}(\mathbf{x})), \quad j = 1,\dots,c, \quad (6)$$

where $\mathcal{F}$ stands for minimum. In a similar way we calculate the class support from the decision profile $DP(\mathbf{x})$ substituting **Maximum**, **Average** and **Product** for $\mathcal{F}$. The way simple combiners work is illustrated in Figure 1.

B. Trainable combiners

The **Naive Bayes (NB)** combination method assumes that the classifiers are mutually independent (this is the reason we use the name "naive"). Denote by s_i the class label assigned to $\mathbf{x}$ by classifier D_i. Let $N(D_i = s_i|\omega_j)$ be the number of points in the training set from class ω_j, for which D_i assigned class label s_i. Then the soft class label for ω_j is calculated as

$$\mu_j(\mathbf{x}) = \frac{1}{N_j^{(L-1)}} \prod_{i=1}^{L} N_c(D_i = s_i|\omega_j), \quad j = 1,\dots,c. \quad (7)$$

If the classifiers in the ensemble are not of identical accuracy, then it is reasonable to attempt to endow the more "competent" classifiers with more power in making the final decision using the **Weighted majority vote (WMAJ)**. We introduce weights or coefficients of importance b_i, $i = 1,\dots,L$, and rewrite (5) as: choose class label ω_k if

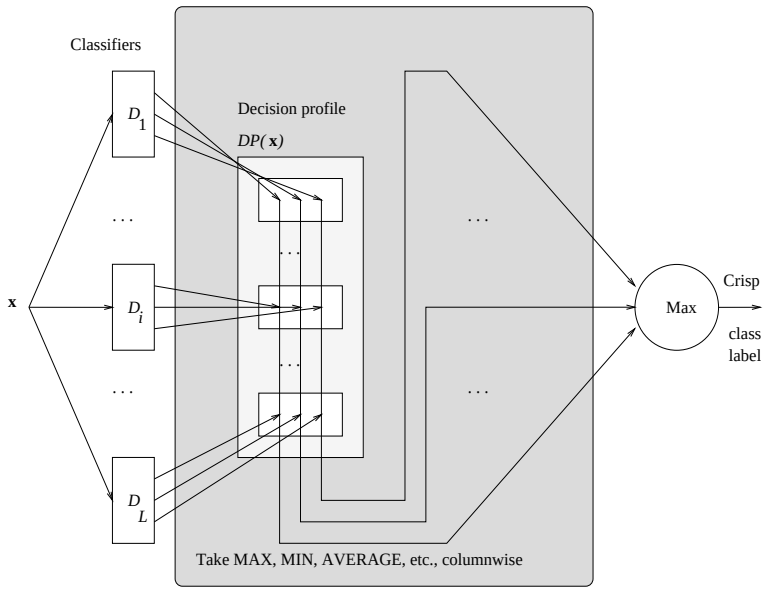
$$\sum_{i=1}^{L} b_i d_{i,k} = \max_{j=1}^{c} \sum_{i=1}^{L} b_i d_{i,j}. \quad (8)$$

One way to select the weights for the classifiers is formalized through the following theorem (paraphrased from [21]), which we state without proof.

Theorem. Consider an ensemble of L *independent* classifiers $D_1,\dots,D_L$, with individual accuracies $p_1,\dots,p_L$, for solving a 2-class pattern recognition problem by the weighted majority vote. Then, using (8), the accuracy of the ensemble is maximized by assigning weights

$$b_i \propto \log \frac{p_i}{1-p_i}. \quad (9)$$

This result has been derived independently by several researchers in different fields of science such as



EXAMPLE. Let $L = 3$ and $c = 2$, and

$$DP(\mathbf{x}) = \begin{bmatrix} 0.3 & 0.7 \\ 0.6 & 0.4 \\ 0.5 & 0.5 \end{bmatrix}.$$

The soft labels for $\mathbf{x}$ are

Method	$\mu_1(\mathbf{x})$	$\mu_2(\mathbf{x})$	Label
Minimum	0.30	0.40	ω_2
Maximum	0.60	0.70	ω_2
Average	0.47	0.57	ω_2
Product	0.09	0.14	ω_2

Fig. 1. Operation of the simple combiners.

democracy studies, pattern recognition and automata theory, leading to the earliest reference [18] according to [1, 21]. Curiously, the optimal weights do not take into account the performance of other members of the team but only magnify the relevance of the individual classifier based on its accuracy. The weighted majority vote is the standard choice for combining the classifiers in ensembles designed by Boosting.

C. Boosting for creating classifier ensembles

Boosting algorithms are amongst the most popular methods for constructing classifier ensembles [3, 7, 10, 20]. They develop the classifier ensemble $\mathcal{D}$ by adding one classifier at a time. The classifier that joins the ensemble at step k is trained on a data set selectively sampled from the training data set Z. The sampling distribution starts from uniform, and progresses towards increasing the likelihood of "difficult" data points. Thus the distribution is updated at each step, increasing the likelihood of the objects misclassified by the classifier at step $k-1$. The basic algorithm, called AdaBoost [9, 20], implementing this idea, is shown in Figure 2.

III. CLASSIFIER COMBINATION: FUZZY

Two methods have been chosen to represent this group: fuzzy integral and decision templates.

Being an aggregation connective [12,13], **Fuzzy Integral (FI)** has been applied to classifier combination in a number of contexts [4–6, 11, 22, 23].

Let H be a fuzzy set on $\mathcal{D}$ expressing the support for class ω_j. We use a fuzzy measure to take into account the importance of any subset of classifiers from $\mathcal{D}$ with respect to ω_j. Two basic types of fuzzy integrals have been proposed: Sugeno type and Choquet type. The *Sugeno fuzzy integral* with respect to a fuzzy measure g is obtained by

$$\mathcal{A}_g^{FI} = \max_{\alpha}\{\min(\alpha, g(H_\alpha))\}, \tag{14}$$

where H_α is the α-cut of H.

EXAMPLE. Let $\mathcal{D} = \{D_1, D_2, D_3\}$, and let the fuzzy measure g be defined as shown in Table I

Let $H = [0.1, 0.7, 0.5]^T$ be a fuzzy set on $\mathcal{D}$ accounting for the support for class ω_j by D_1, D_2, and D_3, respectively (e.g., the jth column of $DP(\mathbf{x})$). The α-cuts of H are

$$\begin{array}{rclrcl} \alpha &=& 0, & H_0 &=& \{D_1, D_2, D_3\}; \\ \alpha &=& 0.1, & H_{0.1} &=& \{D_1, D_2, D_3\}; \\ \alpha &=& 0.5, & H_{0.5} &=& \{D_2, D_3\}; \\ \alpha &=& 0.7, & H_{0.7} &=& \{D_2\}; \\ \alpha &=& 1, & H_0 &=& \emptyset. \end{array}$$

Then

$$\begin{aligned} \mu_j(\mathbf{x}) &= \mathcal{A}_g^{FI} \\ &= \max\{\min(0,1), \min(0.1,1), \min(0.5,0.8), \\ &\quad \min(0.7,0.1), \min(1,0)\} \\ &= \max\{0, 0.1, 0.5, 0.1, 0\} = 0.5. \end{aligned}$$ ■

The fuzzy measure g can be calculated from a set of L values g^i, called *fuzzy densities*, representing the individual importance of D_i. We can find a λ-fuzzy measure which is consistent with these densities. The

AdaBoost

1. Initialize all coefficients as $W_1(i) = \frac{1}{N}$, $i = 1, \ldots, N$. Start with an empty classifier ensemble $\mathcal{D} = \emptyset$.
2. For $k = 1, \ldots, L$
 2.1. Take a sample S_k from Z using distribution W_k.
 2.2. Build a classifier D_k using S_k as the training set.
 2.3. Calculate the weighted ensemble error at step k by

$$\epsilon_k = \sum_{i=1}^{N} W_k(i)(1 - y_{i,k}), \tag{10}$$

where $y_{i,k} = 1$, if D_k correctly recognizes $\mathbf{z}_i \in \mathbf{Z}$, and $y_{i,k} = 0$, otherwise. If $\epsilon_k = 0$ or $\epsilon_k \geq 0.5$, the weights $W_k(i)$ are reinitialized to $\frac{1}{N}$.
 2.4. Calculate the coefficient

$$\beta_k = \sqrt{\frac{1 - \epsilon_k}{\epsilon_k}}, \quad \epsilon_k \in (0, 0.5), \tag{11}$$

to be used in the weighted voting.
 2.5. Update the individual weights for $i = 1, \ldots, N$.

$$W_{k+1}(i) = \frac{W_k(i)\beta_k^{(1-2y_{i,k})}}{\sum_{j=1}^{N} W_k(j)\beta_k^{(1-2y_{j,k})}}. \tag{12}$$

End k.
3. The final decision for a new object $\mathbf{x}$ is made by weighted voting between the L classifiers. First, all classifiers give labels for $\mathbf{x}$ and then for all D_k that gave label ω_t, we calculate the support for that class by

$$\mu_t(\mathbf{x}) = \sum_{D_k(\mathbf{x})=\omega_t} \ln(\beta_k). \tag{13}$$

The class with the maximal support is chosen for $\mathbf{x}$.

Fig. 2. A general description of AdaBoost for classifier ensemble design

TABLE I
AN EXAMPLE OF THE VALUES OF A FUZZY MEASURE g OVER A SET OF THREE CLASSIFIERS $\mathcal{D} = \{D_1, D_2, D_3\}$.

Subset	D_1	D_2	D_3	D_1, D_2	D_1, D_3	D_2, D_3	D_1, D_2, D_3
g	0.3	0.1	0.4	0.4	0.5	0.8	1

value of λ is obtained as the unique real root greater than -1 of the polynomial

$$\lambda + 1 = \prod_{i=1}^{L}(1 + \lambda g^i), \quad \lambda \neq 0. \tag{15}$$

The operation of fuzzy integral as a classifier combiner is shown in Figure 3.

The support for ω_k, $\mu_k(\mathbf{x})$, can be thought of as a "compromise" between the *competence* (represented by the fuzzy measure g) and the *evidence* (represented by the k-th column of the decision profile $DP(\mathbf{x})$). Notice that the fuzzy measure vector $[g(1), \ldots, g(L)]^T$ might be different for each class, and is also specific for the current $\mathbf{x}$. Two fuzzy measure vectors will be the same only if the ordering of the classifier support is the same. The algorithm in Figure 3 calculates a Sugeno fuzzy integral. For the Choquet fuzzy integral with

Fuzzy integral for classifier fusion

1. Fix the L *fuzzy densities* $g^1, \ldots, g^L$, e.g., by setting g^i to the estimated probability of correct classification of D_i.
2. Calculate $\lambda > -1$ from (15).
3. For a given $\mathbf{x}$ sort the kth column of $DP(\mathbf{x})$ to obtain $[d_{i_1,k}(\mathbf{x}), d_{i_2,k}(\mathbf{x}), \ldots, d_{i_L,k}(\mathbf{x})]^T$, $d_{i_1,k}(\mathbf{x})$ being the highest degree of support, and $d_{i_L,k}(\mathbf{x})$, the lowest.
4. Arrange the fuzzy densities correspondingly, i.e., $g^{i_1}, \ldots, g^{i_L}$ and set $g(1) = g^{i_1}$.
5. For $t = 2$ to L, calculate recursively

$$g(t) = g^{i_t} + g(t-1) + \lambda g^{i_t} g(t-1).$$

6. Calculate the final degree of support for class ω_k by

$$\mu_k(\mathbf{x}) = \max_{t=1}^{L} \left\{ \min\{d_{i_t,k}(\mathbf{x}), g(t)\} \right\}.$$

Fig. 3. Fuzzy integral for classifier fusion

1. Decision templates (training)

For $j = 1, \ldots, c$, calculate the mean of the decision profiles $DP(\mathbf{z}_k)$ of all members of ω_j from the data set $\mathbf{Z}$. Call the mean *a decision template* DT_j

$$DT_j = \frac{1}{N_j} \sum_{\substack{\mathbf{z}_k \in \omega_j \\ \mathbf{z}_k \in \mathbf{Z}}} DP(\mathbf{z}_k),$$

where N_j is the number of elements of $\mathbf{Z}$ from ω_j.

2. Decision templates (operation)

Given the input $\mathbf{x} \in \Re^n$, construct $DP(\mathbf{x})$. Calculate the **similarity** $\mathcal{S}$ between $DP(\mathbf{x})$ and each DT_j, $j = 1, \ldots, c$ as the components of the soft label of $\mathbf{x}$

$$\mu_j(\mathbf{x}) = \mathcal{S}(DP(\mathbf{x}), DT_j).$$

Fig. 4. Operation of the Decision templates method

the same λ-fuzzy measure, the last formula should be

$$\mu_k(\mathbf{x}) = d_{i_1,k}(\mathbf{x}) + \sum_{j=2}^{L} \Big(d_{i_{j-1},k}(\mathbf{x}) - d_{i_j,k}(\mathbf{x}) \Big) g(j-1).$$

The idea of the **Decision Templates (DT)** model is to "remember" the most typical decision profile for each class, called the *decision template*, DT_j, for that class, and then compare it with the current decision profile $DP(\mathbf{x})$. The closest match will label $\mathbf{x}$. Figure 4 describes the operation of the decision templates model.

As both $DP(\mathbf{x})$ and DT_j can be regarded as fuzzy sets on $\mathcal{D} \times \Omega$, any measure of similarity between fuzzy sets can be used. Here, based on our previous experience, we use the Euclidean distance, three similarity measures and two inclusion indices. The decision template combiners are named in the experiments as **DT(xx)**, where 'xx' stands for the measure index, e.g., DT(S1). Let A and B be fuzzy sets on some universal set U.

Measures of similarity [8]:

$$S_1(A,B) \equiv \frac{\| A \cap B \|}{\| A \cup B \|}, \tag{16}$$

where $\| \zeta \|$ is the relative cardinality of the fuzzy set ζ on U

$$S_2(A,B) \equiv 1 - \| A \nabla B \|, \tag{17}$$

where $A \nabla B$ is the symmetric difference defined by the Hamming distance $\mu_{A \nabla B}(u) = |\mu_A(u) - \mu_B(u)|$.

$$S_3(A,B) \equiv 1 - \| A \Delta B \|, \tag{18}$$

where $\mu_{A \Delta B}(u) = \max\{\mu_{A \cap \bar{B}}(u), \mu_{\bar{A} \cap B}(u)\}$.

Indices of inclusion of A (the decision profile $DP(\mathbf{x})$ in our case) in B (the decision template DT_j) [8]

$$I_1(A,B) \equiv \frac{\| A \cap B \|}{\| A \|}. \tag{19}$$

$$I_2(A,B) \equiv 1 - \| A| - |B \|. \tag{20}$$

where $|-|$ is the bounded difference

$$\mu_{A|-|B}(u) = \max\{0, \mu_A(u) - \mu_B(u)\}. \tag{21}$$

EXAMPLE. Let $c = 3$, $L = 2$, and

$$DT_1 = \begin{bmatrix} 0.6 & 0.4 \\ 0.8 & 0.2 \\ 0.5 & 0.5 \end{bmatrix} \quad \text{and} \quad DT_2 = \begin{bmatrix} 0.3 & 0.7 \\ 0.4 & 0.6 \\ 0.1 & 0.9 \end{bmatrix}.$$

Assume that for an input $\mathbf{x}$, the following decision profile has been obtained

$$DP(\mathbf{x}) = \begin{bmatrix} 0.3 & 0.7 \\ 0.6 & 0.4 \\ 0.5 & 0.5 \end{bmatrix}.$$

The similarities and the class labels using DT(E) to DT(I3) are shown in Table II.

■

TABLE III

SUMMARY OF THE DATA SETS USED

Database	n	c	N	$\hat{P}_{max}$	Availability
Pima Indians Diabetes	8	2	768	65.10 %	UCI[c]
Phoneme	5	2	5404	70.65 %	ELENA[b]
Cone-torus	2	3	800	50.00 %	Private[a]
Cleveland Heart Disease	13	2	303	54.48 %	UCI[c]
Wisconsin Diagnostic Breast Cancer	30	2	569	62.74 %	UCI[c]
Satimage data	36	6	6435	23.82 %	ELENA[b]

Notations:

n: number of features
c: number of classes
N: number of cases in the database
$\hat{P}_{max}$: the largest class proportion

[a] http://www.bangor.ac.uk/∼mas00a/Z.txt and Zte.txt
[b] ftp ftp.dice.ucl.ac.be, directory pub/neural/ELENA,
[c] http://www.ics.uci.edu/∼mlearn/MLRepository.html

TABLE II

THE SIMILARITIES AND THE CLASS LABELS USING THE DECISION TEMPLATES COMBINATION METHOD.

DT(xx)	$\mu_1(\mathbf{x})$	$\mu_2(\mathbf{x})$	Label
DT(E)	0.9567	0.9333	ω_1
DT(S1)	0.7143	0.6667	ω_1
DT(S2)	0.8333	0.8000	ω_1
DT(S3)	0.5000	0.5333	ω_2
DT(I1)	0.8333	0.8000	ω_1
DT(I2)	0.9167	0.9000	ω_1

IV. EXPERIMENTS

A. Experimental setup

We used six data sets as summarized in Table III:

- Pima Indians Diabetes data: A population of women living near Phoenix, Arizona, USA of Pima Indian heritage were tested for diabetes according to World Health Organization criteria.
- Phoneme data: Five-dimensional vectors characterizing two classes of phonemes: nasals (70.65 %) and orals (29.35 %).
- Cone-torus data: A three-class dataset with 400 2-d points generated from three differently shaped distributions: a cone, half a torus, and a normal distribution with prior probabilities 0.25, 0.25, and 0.5, respectively. A separate data set for testing with 400 more points generated from the same distribution is also available as the file Zte.txt.
- Cleveland Heart Disease data[1]: The presence or absence of heart disease is predicted based on 13 features. There are a few missing values in the data. In our experiments these were replaced by the average of the column (feature) regardless of the class labels.
- Wisconsin Diagnostic Breast Cancer data: Features are computed from a digitized image of a fine needle aspirate of a breast mass. The mass is classed as benign or malignant.
- Satimage data: Generated from the Landsat Multi-Spectral Scanner image data; consists of 6435 patterns (pixels) with 36 attributes (4 spectral bands × 9 pixels in neighborhood). In this series of experiments we used features 5 to 9. Pixels are classified in 6 classes, and are presented in random order in the database.

We performed 2-fold cross-validation with all data sets, taking at random one half of the data for training and the other half for testing, and then swapping the two sets. All the choices of the parameters and the classifier training was done on the training sets only.

All data sets were normalized in the following way. A linear transformation was used, separately for each feature, to bring its values within the interval $[0, 1]$. The *training set* was used to find the minimum and the maximum of the feature values. The testing set was transformed using these same constants.

The AdaBoost algorithm in Figure 2 was implemented to build ensembles of $L = 15$ classifiers with each data set. The individual classifiers were multilayer perceptron (MLP) neural networks with one hidden layer consisting of 15 nodes, trained for 300 epochs by fast backpropagation (Matlab Neural Network Toolbox). We recorded the training and testing

[1] Dr. Robert Detrano collected the data base; V.A. Medical Center, Long Beach and Cleveland Clinic Foundation.

accuracy during the AdaBoost iterates for all combination methods described in Sections II and III.

B. Results

The results found with the simple combination methods: average, minimum, maximum and product were surprisingly inadequate, so we left these methods out of the comparison. Figure 5 shows the testing accuracies for the remaining methods during the progressive ensemble generation.

Table IV shows the testing accuracies of the combination methods for the 6 data sets at the end of the training, i.e., when the ensemble consisted of $L = 15$ classifiers. The lines separate the standard combination methods for AdaBoost, the Weighted majority vote (WMAJ), the non-fuzzy combination methods (MAJ and NB), and the fuzzy methods (FI and DTs).

To facilitate the comparison we also calculated the relative performance of each method with respect to the others. The columns with the accuracies were sorted individually and each combination model was assigned a rank with respect to its place among the others. The highest rank (value 10) was assigned to the best model and the lowest rank (value 1) was assigned to the worst model. The ranks are shown in Table V. The 6 ranks for each combination model were then added up to give a measure of the overall dominance among the models. The total ranks are displayed in the last column of the table.

Figure 6 shows a scatterplot of the ranks of the combination methods on the axes: training rank / testing rank. The higher the ranks, the better the method. Shown also is the diagonal line corresponding to equal ranks in training and testing. Naturally, the testing ranks (see Table V) are much more important, and the methods should be judged on these. The figure shows that the Weighted majority vote had the perfect score on the training (out-ranking all the other methods) but on the testing data fuzzy combination methods were better.

C. Discussion

The results show that the fuzzy combination methods give stable results for all data sets. The only data set where the standard combination outperformed the rest of the methods was the Phoneme data. With the other 5 data sets, fuzzy combination methods were better, markedly so for the Satimage data. Perhaps the room for improvement in the Satimage data comes from the fact that there are 6 classes, and so the recognition problem is more complicated than for the other

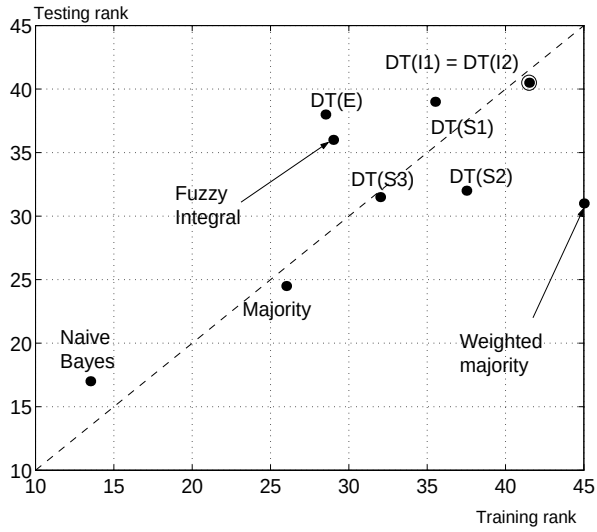


Fig. 6. Testing ranks versus training ranks of the fuzzy and non-fuzzy combination methods. The higher the ranks, the better the method.

data sets.

The overall accuracy of the ensembles is not particularly high, compared to the results reported elsewhere. This could be a results from a poor selection of the parameters of the individual classifiers, i.e., the MLP configuration and training protocol. Another reason is that we used a 2-fold cross-validation, so only 50 % of the data was used for training. With a 10-fold cross-validation, the classifiers are trained on 90 %, hence a higher accuracy could be expected. In any case, the purpose of this study was to explore the potential of some fuzzy combination methods compared to the standard choice and some popular non-fuzzy methods.

Surprisingly, the simple combiners such as average, minimum, maximum and product gave poor results, so much so, that we dropped them off the comparison. This might be due to the fact that these methods expect the classifiers in the ensemble to be at a "reasonable" standard. AdaBoost may generate members of the team with arbitrarily low performance rate, and the simple combiners are not equipped to cope with this imbalance.

The overall score favored fuzzy methods over the non-fuzzy ones. The best combiner in our experiments appeared to be the Decision Templates methods with similarity based on I_1 and I_2. All DT versions together with the fuzzy integral had approximately identical consistently good behavior. The two non-fuzzy methods: majority vote and Naive Bayes showed erratic patterns across the data sets, probably sharing

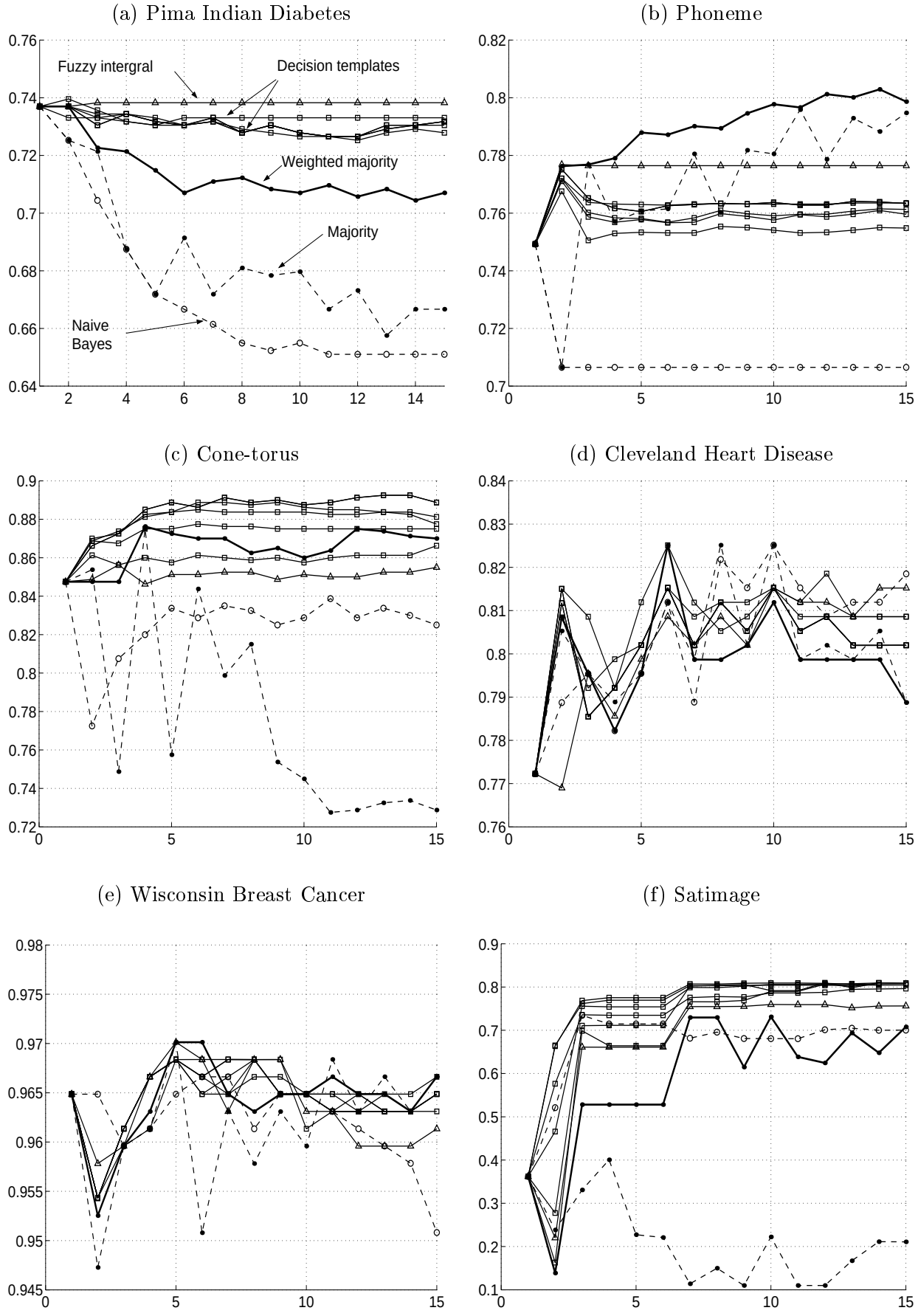


Fig. 5. The testing accuracy of the combination methods during the incremental design of the ensemble by AdaBoost. All methods are displayed with the same lines as explained in subplot (a).

TABLE IV

TESTING ACCURACIES FOR THE COMBINATION METHODS AND THE 6 DATA SETS.

Method	Pima	Phoneme	Cone-torus	Cleveland	Wisconsin	Satimage
Weighted majority	70.7	79.9	87.0	78.9	96.7	70.8
Majority	66.7	79.5	72.9	78.9	96.7	21.1
Naive Bayes	65.1	70.7	82.5	81.8	95.1	70.0
Fuzzy integral	73.8	77.6	85.5	81.5	96.1	75.6
Decision templates (I1)	73.2	76.3	88.9	80.2	96.5	80.8
Decision templates (I2)	73.2	76.3	88.9	80.2	96.5	80.8
Decision templates (S1)	73.2	76.1	88.1	80.2	96.5	80.8
Decision templates (S2)	73.0	76.0	87.8	80.2	96.5	80.5
Decision templates (S3)	72.8	75.5	86.6	80.9	96.7	79.6
Decision templates (E)	73.3	76.4	87.5	80.9	96.3	80.5

TABLE V

TESTING ACCURACIES AND RANKS FOR THE COMBINATION METHODS AND THE 6 DATA SETS.

Method	Pima	Phoneme	Cone-torus	Cleveland	Wisconsin	Satimage	Total
Weighted majority	3.0	10.0	5.0	1.5	8.5	3.0	31.0
Majority	2.0	9.0	1.0	1.5	10.0	1.0	24.5
Naive Bayes	1.0	1.0	2.0	10.0	1.0	2.0	17.0
Fuzzy integral	10.0	8.0	3.0	9.0	2.0	4.0	36.0
Decision Templates (I1)	7.0	5.5	9.5	4.5	5.5	8.5	40.5
Decision Templates (I2)	7.0	5.5	9.5	4.5	5.5	8.5	40.5
Decision Templates (S1)	7.0	4.0	8.0	4.5	5.5	10.0	39.0
Decision Templates (S2)	5.0	3.0	7.0	4.5	5.5	7.0	32.0
Decision Templates (S3)	4.0	2.0	4.0	8.0	8.5	5.0	31.5
Decision Templates (E)	9.0	7.0	6.0	7.0	3.0	6.0	38.0

the inadequacy of the simple combiners towards imbalanced classifier ensembles.

V. CONCLUSIONS

We studied the potential of fuzzy combination methods for ensembles of classifiers designed by Boosting. Since Boosting induces diversity in classifier performances, which is beneficial to the team but might lead to great imbalances of the accuracies, simple combination methods such as average, minimum, maximum and product, failed to produce a sensible ensemble output. The best among the non-fuzzy combination methods appeared to be the Naive Bayes and the simple majority vote but they also were inferior to the fuzzy combiners.

There are different variants of Boosting, most of them called commonly AdaBoost. For example, the version implemented here can be thought of as "aggressive" in the sense that the weights of all data points are actively modified at each iteration: the weights of the misclassified objects are increased and these of the correctly classified objects are decreased. There is a more "conservative" implementation where the weights change in only one way: either increasing the weights of the mislabeled objects or decreasing these of the correctly labeled objects. Such versions are called again AdaBoost in [3] and [10]. It is possible that the ranking of the methods might change depending on which version we used. Also, the choice of the base classifier models might have an effect on the preference of the combiners. Thus the claim here is not that "the fuzzy combiners are better". It is a well known postulate in pattern recognition that there is no "best" classifier or "best" combination method. What this study suggests is to keep fuzzy combiners on the list of options, and assign them high priority when choosing a combiner.

References

[1] Y. L. Barabash. *Collective Statistical Decisions in Recognition*. Radio i Sviaz', Moscow, 1983. (In Russian).

[2] R. Battiti and A.M. Colla. Democracy in neural nets: Voting schemes for classification. *Neural Networks*, 7:691–707, 1994.

[3] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–142, 1999.

[4] J.A. Benediktsson, J.R. Sveinsson, J. I. Ingimundarson, H. Sigurdsson, and O.K. Ersoy. Multistage classifiers optimized by neural networks and genetic algorithms. *Nonlinear Analysis, theory, Methods & Applications*, 30(3):1323–1334, 1997.

[5] S.-B. Cho and J.H. Kim. Combining multiple neural networks by fuzzy integral and robust classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 25:380–384, 1995.

[6] S.B. Cho and J.H. Kim. Multiple network fusion using fuzzy logic. *IEEE Transactions on Neural Networks*, 6:497–501, 1995.

[7] T.G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15, Cagliari, Italy, 2000. Springer.

[8] D. Dubois and H. Prade. *Fuzzy Sets and Systems: Theory and Applications*. Academic Press, NY, 1980.

[9] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, NY, second edition, 2001.

[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*, 55(1):119–139, 1997.

[11] P.D. Gader, M.A. Mohamed, and J.M. Keller. Fusion of handwritten word classifiers. *Pattern Recognition Letters*, 17:577–584, 1996.

[12] M. Grabisch. On equivalence classes of fuzzy connectives - the case of fuzzy integrals. *IEEE Transactions on Fuzzy Systems*, 3(1):96–109, 1995.

[13] M. Grabisch and M. Sugeno. Multi-attribute classification using fuzzy integral. In *IEEE International Conference on Fuzzy Systems*, pages 47–54, San Diego, California, 1992.

[14] L.I. Kuncheva. Combining classifiers: Soft computing solutions. In S.K. Pal and A. Pal, editors, *Pattern Recognition, From Classical to Modern Approaches*, chapter 15, pages 427–452. World Scientific, 2001.

[15] L.I. Kuncheva, J.C. Bezdek, and R.P.W. Duin. Decision templates for multiple classifier fusion: an experimental comparison. *Pattern Recognition*, 34(2):299–314, 2001.

[16] L. Lam and A. Krzyzak. A theoretical analysis of the application of majority voting to pattern recognition. In *12th International Conference on Pattern Recognition*, pages 418–420, Jerusalem, Israel, 1994.

[17] L. Lam and C.Y. Suen. Application of majority voting to pattern recognition: An analysis of its behavior and performance. *IEEE Transactions on Systems, Man, and Cybernetics*, 27(5):553–568, 1997.

[18] W. Pierce. *Improving reliability of digital systems by redundancy and adaptation*. PhD thesis, Electrical Engineering, Stanford University, 1961.

[19] D. Ruta and B. Gabrys. A theoretical analysis of the limits of majority voting errors for multiple classifier systems. Technical Report 11, ISSN 1461-6122, Department of Computing and Information Systems, University of Paisley, December 2000.

[20] R.E. Schapire. Theoretical views of boosting. In *Proc. 4th European Conference on Computational Learning Theory*, pages 1–10, 1999.

[21] L. Shapley and B. Grofman. Optimizing group judgemental accuracy in the presence of interdependencies. *Public Choice*, 43:329–343, 1984.

[22] A. Verikas, A. Lipnickas, K. Malmqvist, M. Bacauskiene, and A. Gelzinis. Soft combination of neural classifiers: A comparative study. *Pattern Recognition Letters*, 20:429–444, 1999.

[23] D. Wang, J. M. Keller, C.A. Carson, K.K. McAdoo-Edwards, and C.W. Bailey. Use of fuzzy-logic-inspired features to improve bacterial recognition through classifier fusion. *IEEE Transactions on Systems, Man, and Cybernetics*, 28B(4):583–591, 1998.