“Fuzzy” Versus “Nonfuzzy” 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 two-fold cross-validation experiments on six benchmark data sets to compare the fuzzy and nonfuzzy combination methods. On the “fuzzy side” we used the fuzzy integral and the decision templates with different similarity measures. On the “nonfuzzy side” we tried the weighted majority vote as well as simple combiners such as the majority vote, minimum, maximum, average, product, and the Naive–Bayes combination. In our experiments, the fuzzy combination 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.**

***Index Terms*—Adaboost, classifier combination, decision templates, ensembles of classifiers created by Boosting, fuzzy integral, weighted majority vote.**

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. We look at classifier ensembles generated by Boosting, which is recognized as one of the most successful algorithms for creating classifier ensembles [3], [11], [16], [36], [37]. 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 *weighted majority vote* is the standard combination method for ensembles generated by boosting. As explained later in the text, weighted majority vote is optimal for the special case of two classes and classifiers with independent outputs. Practice shows, however, that even independently designed classifiers will hardly have independent outputs [21]. Classifiers designed by Adaboost are dependent because each subsequent member of the ensemble is built on a training set influenced by its predecessor. Yet, weighted majority vote works well regardless of the violations of the optimality assumptions. Therefore, there is no reason why other combination methods should not be successful on ensembles generated by boosting.

Many combination methods and algorithms have been developed, including methods based on fuzzy sets [27]. However, treating combining classifiers as a branch of *statistical pattern recognition* sometimes brings about an unwelcome attitude toward using fuzzy combiners. The purpose of this study is to examine experimentally how useful fuzzy combiners are for boosted ensembles by a comparison with popular nonfuzzy combiners.

The difficulty in choosing a suitable combination method for the problem at hand has been recognized and highlighted numerous times in the literature on combining classifiers. So far, we do not have a sufficient body of theory to explain the success of ensembles compared to single classifiers and match combination strategies and methods to a problem. Pieces of theory developed hitherto rely on simplifications and assumptions, and consider mostly special cases [7], [17], [24], [28], [36], [37], [39]–[41]. However, even a discipline as mature as pattern recognition itself does not offer strict guidelines about how to approach a data set and which classifier to select for it. Along the years, the advantages of various classifier models have been demonstrated across different data sets so that the best contestants have been identified amongst thousands of possibilities [15]. Being a relatively recent offspring of pattern recognition and machine learning, combining classifiers still enjoys many heuristic ideas. Establishing even vague priority among these is, therefore, a matter of importance. Many experimental studies have been published in the search of such guidelines, e.g., [25], [26], and [34]. This study also belongs in the experimental group.

The text is organized as follows. Section II introduces the formalism of combining classifiers and the nonfuzzy combination methods: majority vote, weighted majority vote (the standard choice for the Boosting algorithm), minimum, maximum, average, product, and the Naive–Bayes (NB) combiner. The Boosting algorithm for generating an ensemble is also explained there. The “fuzzy competitors” are presented in Section III: Fuzzy Integral [4], [8], [9], [18], [42], [43] and decision templates (DTs) [23], [29]. Section IV contains the experimental set up and the results. We analyze the results in Section V and offer some conclusions in Section VI.

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., $D_i : \Re^n \rightarrow \Omega$ or, equivalently, $D_i(\mathbf{x}) \in \Omega$, $i = 1, \ldots c$. Alternatively, the classifier output can be formed as a c-dimensional vector

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

Manuscript received February 15, 2002; revised August 28, 2002 and February 24, 2003.

The author is with the School of Informatics, University of Wales, Bangor LL57 1UT, U.K. (e-mail: l.i.kuncheva@bangor.ac.uk).

Digital Object Identifier 10.1109/TFUZZ.2003.819842

```
AdaBoost

1. Initialize all coefficients as $W_1(i) = \frac{1}{N}$, $i = 1,\ldots,N$, $N$ being the size of the data
set. 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

       $\epsilon_k = \sum_{i=1}^{N} W_k(i)(1 - y_{i,k}),$   (5)

  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

       $\beta_k = \sqrt{\frac{1-\epsilon_k}{\epsilon_k}}, \quad \epsilon_k \in (0, 0.5),$   (6)

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

       $W_{k+1}(i) = \frac{W_k(i)\beta_k^{(1-2y_{i,k})}}{\sum_{j=1}^{N} W_k(j)\beta_k^{(1-2y_{j,k})}}.$   (7)

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 $\omega_t$, we calculate the
support for that class by

       $\mu_t(\mathbf{x}) = \sum_{D_k(\mathbf{x})=\omega_t} \ln(\beta_k).$   (8)

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

Fig. 1. General description of AdaBoost for classifier ensemble design

where $d_{i,j}(\mathbf{x})$ is the degree of "support" given by classifier D_i to the hypothesis that **x** comes from class ω_j. Most often $d_{i,j}(\mathbf{x})$ is an estimate of the posterior probability $P(\omega_i|\mathbf{x})$. In fact, the detailed interpretation of $d_{i,j}(\mathbf{x})$ beyond a "degree of support" is not important for the operation for any of the combination methods studied here. Except for the decision templates method (explained later), where similarities between fuzzy sets are calculated, $d_{i,j}(\mathbf{x})$ does not even need to be restricted in the interval [0,1].

It is convenient to organize the output of all L classifiers in a *decision profile* [29]

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

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], $i = 1,\ldots,L$, $j = 1,\ldots,c$, and call the classifier outputs "soft labels." *Combining classifiers* means to find a class label for **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 **x**, denoted

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

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

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

Ties are resolved arbitrarily. The minimum-error classifier is recovered from (4) when $\mu_i(\mathbf{x}) \propto P(\omega_i|\mathbf{x})$. Again, there is no reason why $\mu_i(\mathbf{x})$ should be restricted in the interval [0,1].

A. Boosting for Creating Classifier Ensembles

Boosting algorithms are amongst the most popular methods for constructing classifier ensembles [3], [11], [16], [36]. 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 toward 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 [15], [36], implementing this idea, is shown in Fig. 1.

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

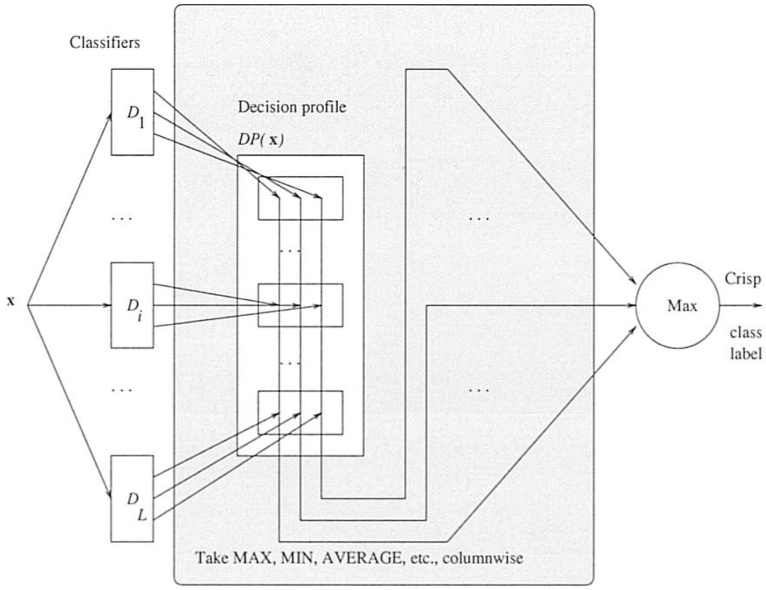


Fig. 2. Operation of the simple combiners.

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

$$DP(\mathbf{x}) = \begin{bmatrix} 0.6 & 0.2 \\ 0.7 & 0.4 \\ 0.4 & 0.4 \end{bmatrix}.$$

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

Method	$\mu_1(\mathbf{x})$	$\mu_2(\mathbf{x})$	Label
Minimum	0.60	0.20	ω_1
Maximum	0.70	0.80	ω_2
Average	0.57	0.47	ω_1
Product	0.17	0.06	ω_1

B. Nontrainable 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 **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 $[d_{i,1}, \ldots, d_{i,c}]^T \in \{0,1\}^c$, $i = 1, \ldots, L$, where $d_{i,j} = 1$ if D_i labels **x** in ω_j, and 0, otherwise, $\sum_{j=1}^{c} d_{i,j} = 1$. The *plurality vote* will pick class ω_k if

$$\sum_{i=1}^{L} d_{i,k} = \max_{j=1}^{c} \sum_{i=1}^{L} d_{i,j}. \quad (9)$$

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, e.g., [1], [2], [31], [32], and [35].

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

$$\mu_j(\mathbf{x}) = \min\left(d_{1,j}(\mathbf{x}), \ldots, d_{L,j}(\mathbf{x})\right), \quad j = 1, \ldots, c. \quad (10)$$

In a similar way, we calculate the class support from the decision profile $DP(\mathbf{x})$ taking **Maximum**, **Average** and **Product** separately for each column. The way simple combiners work is illustrated in Fig. 2.

C. Trainable Combiners

The **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 **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. For example, let the label for **x** suggested by classifier D_i be ω_2. In calculating the support for, say, ω_3, we use $N(D_i = \omega_2|\omega_3)$ which is the entry (3,2) in the confusion matrix for D_i.[1] The support for ω_j is calculated as

$$\mu_j(\mathbf{x}) = \frac{1}{N_j^{(L-1)}} \prod_{i=1}^{L} N(D_i = s_i|\omega_j), \quad j = 1, \ldots, c \quad (11)$$

where N_j is the total number of patterns from ω_j in **Z**.

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, $i = 1, \ldots, L$, and rewrite (9) as: Choose class label ω_k if

$$\sum_{i=1}^{L} b_i d_{i,k} = \max_{j=1}^{c} \sum_{i=1}^{L} b_i d_{i,j}. \quad (12)$$

One way to select the weights for the classifiers is formalized through the following theorem (paraphrased from [38]), which we state without proof.

Theorem: Consider an ensemble of L *independent* classifiers $D_1, \ldots, D_L$, with individual accuracies $p_1, \ldots, p_L$, for solving a two-class pattern recognition problem by the weighted majority vote. Then, using (12), the accuracy of the ensemble is maximized by assigning weights

$$b_i \propto \log \frac{p_i}{1 - p_i}. \quad (13)$$

This result has been derived independently by several researchers in different fields of science such as democracy studies, pattern recognition, and automata theory, leading to the earliest reference [33] according to [1] and [38]. Curiously, the optimal weights do not take into account the performance of other members of the team but only magnify the relevance

[1] Recall that the confusion matrix is calculated on the training set so that its (i, j)th entry in the number of objects with true label ω_i, labeled by the classifier as ω_j.

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.

III. Classifier Combination: Fuzzy

If we restrict $d_{i,j}(\mathbf{x})$ into [0,1], we can use numerous aggregation connectives defined for fuzzy sets [5], [13], [19], [46], [47]. Which connective or rather which class of connectives are the most appropriate ones could be related to the semantics of the degree of support, i.e., whether they can compensate one another, etc. It is perhaps more difficult to interpret the classifier outputs in the semantic frameworks suggested in the literature [14] than to pick an aggregation connective once the context has been clarified. The most common aggregation connectives, perceived sometimes as trademarks of fuzzy set theory, are already in use: minimum, maximum, simple average, and product. We placed them as the nonfuzzy nontrainable combiners. Variation of these with different level of "optimism" in the aggregation are also among the possible choices.

Here we selected two methods to represent this group: fuzzy integral (reported to give good results) and decision templates (simple and intuitive).

Fuzzy integral (FI) [19], [20] has been applied to classifier combination in a number of contexts [4], [8], [9], [18], [42], [43].

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.

TABLE I
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

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 (the jth column of $DP(\mathbf{x})$). The α-cuts of H are

$$\begin{array}{lllll}
\alpha & = & 0, & H_0 & = & \{D_1, D_2, D_3\} \\
\alpha & = & 0.1, & H_{0.1} & = & \{D_1, D_2, D_3\} \\
\alpha & = & 0.5, & H_{0.5} & = & \{D_2, D_3\} \\
\alpha & = & 0.7, & H_{0.7} & = & \{D_2\} \\
\alpha & = & 1, & H_1 & = & \emptyset.
\end{array}$$

Then

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

The fuzzy measure g can be calculated from a set of L values g^i, called fuzzy densities,[2] representing the individual importance of D_i. We can find a λ-fuzzy measure which is consistent with these densities. The 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 Fig. 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 **x**. Two fuzzy measure vectors will be the same only if the ordering of the classifier support is the same. The algorithm in Fig. 3 calculates a Sugeno fuzzy integral. For the Choquet fuzzy integral with the same λ-fuzzy measure, the last formula should be

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

The idea of the **decision templates (DTs)** 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 **x**. Fig. 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 [6], [12]. 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 or the index, e.g., DT(S1). Let A and B be fuzzy sets on some universal set U.

The following measures of similarity were used [12]

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

where $\| \zeta \|$ is the relative cardinality of the fuzzy set ζ on U, $\cap$ denotes minimum and $\cup$ denotes maximum

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

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

The following indexes of inclusion of A (the decision profile $DP(\mathbf{x})$ in our case) in B (the decision template DT_j) were used [12]:

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

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

[2] The term "fuzzy densities" appears in the literature as a convenient short-hand for "the point-wise values of the fuzzy measure."

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 (or to values proportional to the accuracies [9,42,43]).
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

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

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 DTs method.

where $\ominus$ is the bounded difference

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

Example: Let $c = 3$, $L = 2$, and the decision templates for ω_1 and ω_2 be, respectively

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

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(I2) are shown in Table II. ■

We note that both fuzzy integral and decision templates are trainable combiners. For the fuzzy integral, the only quantities that have to be estimated are the g^i's (L parameters) whereas DTs require c decision templates of size $L \times c$ each. This suggests that decision templates might be more prone to overtraining than fuzzy integral.

TABLE II
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. Except for the Cone-torus data, the other data sets have been extensively used as benchmarks in the recent literature including that on combining classifiers. It is difficult to establish a good estimate of the accuracy from past usage because of the difference in the experimental protocols (two-fold cross-validation, ten-fold cross-validation, single hold-out results, etc.) Since the aim of this study is to compare fuzzy and nonfuzzy combiners, both of which will be examined under the same experimental protocol, we shall not be overly concerned with the absolute value of the

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

accuracy. The largest class proportion is given in the table as a lower bound of the classification accuracy, i.e., the accuracy when labeling any object in the the most probable class.

Cone-torus is 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. For the Cleveland Heart Disease data[3] 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.

We performed two-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 Fig. 1 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 accuracy during the AdaBoost iterates for all combination methods described in Sections II and III.

B. Results

Fig. 5 shows the testing accuracies for three selected methods during the progressive ensemble generation: Weighted majority vote (best from the nonfuzzy group), fuzzy integral, and decision templates with Euclidean distance.

Table IV shows the testing accuracies of the combination methods for the six 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 nonfuzzy combination methods (MAJ, NB, and simple combiners), 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 14) 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 six 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.

To find out whether these results are due to chance or reveal a steady pattern on the preference, we apply a difference-of-proportions test for every pair of combiners at level of significance 0.05. Since the assumption of independence between the two samples of interest is generally not true (the same data was used to test both combiners), the results might be on the conservative side [10]. This means that there might be more true differences at the same significance level, undetected hereby. Table VI gives the results from the test. Since we used six data sets, there will be six comparison results for every pair of combination methods. The entries in the table should be read as "[better same worse]" for the six comparisons. Thus, the entry for (MIN,NB), "213," means that the minimum combination method has been found significantly better than Naive Bayes method in two of the six comparisons, the same in one comparison, and worse in three comparisons.

V. ANALYSIS AND DISCUSSION

A. General Remarks

At a first sight, the results favor the fuzzy combination methods. We note that the overall accuracy of the ensembles is not particularly high, compared to the results reported elsewhere. This could be due to 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 two-fold cross-validation, so only 50% of the data was used for training. With a ten-fold cross-validation, the classifiers are trained on 90% of the data, hence a higher accuracy could be expected. In

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

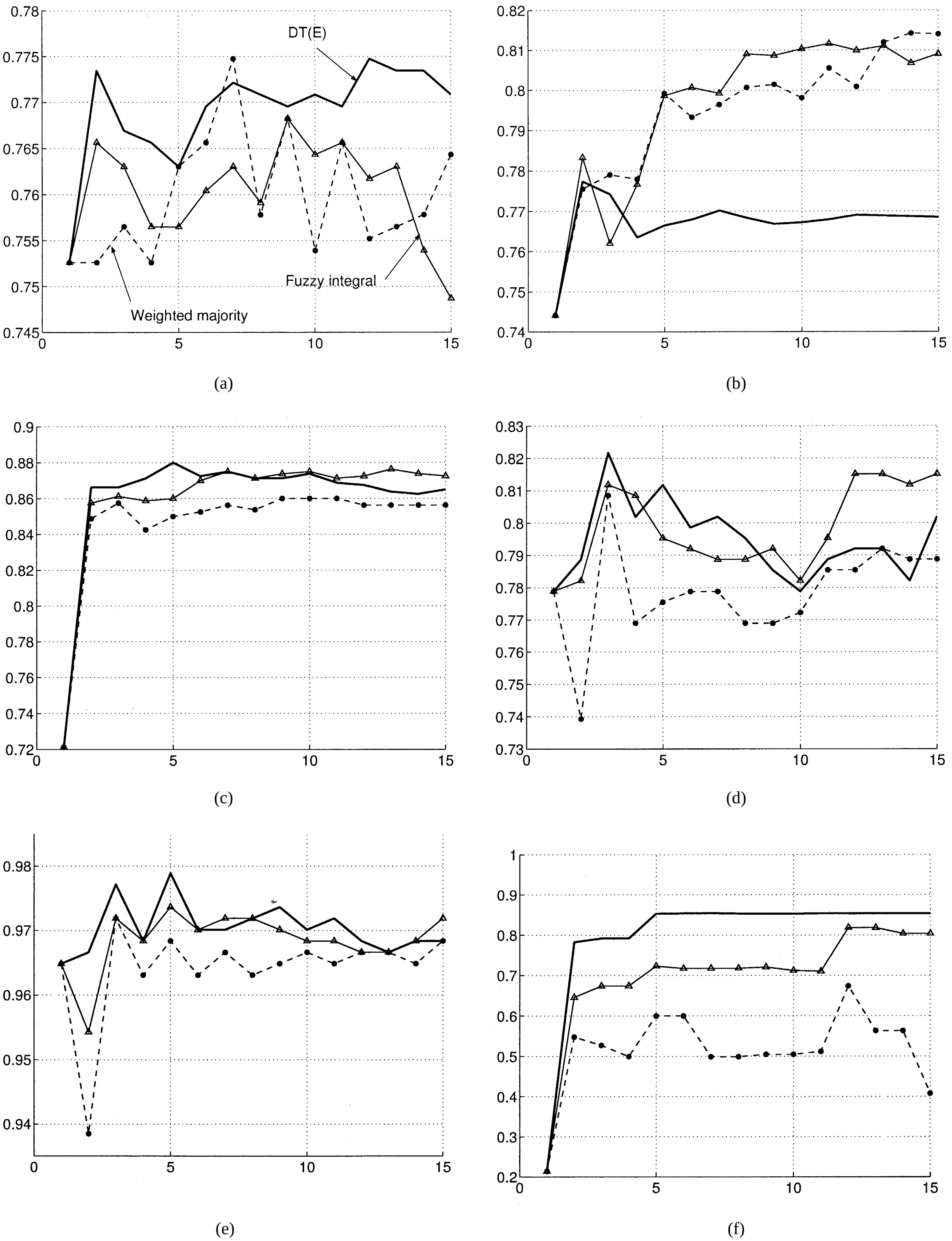


Fig. 5. Testing accuracy of the combination methods during the incremental design of the ensemble by AdaBoost. All methods are displayed with the same lines as explained in subplot (a).

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 nonfuzzy methods.

The overall rank score in Table V also places fuzzy methods before the nonfuzzy ones. The best combiner in our experiments appeared to be the DTs method based on Euclidean distance (DT(E)) followed by the fuzzy integral (FI). Average was the best from the nontrainable group, confirming the findings from other studies [24].

Table VI identifies the decision templates with $S3$ (DT(S3)) as the only nondominated combination method. The first digit of all the entries in its column is 0, indicating that in none of the comparisons another method has been found significantly better. This suggests that although DT(E) and fuzzy integral

TABLE IV
TESTING ACCURACIES FOR THE COMBINATION METHODS AND THE SIX DATA SETS

Method	Pima	Phoneme	Cone-torus	Cleveland	Wisconsin	Satimage
Weighted majority	76.4	81.4	85.6	78.9	96.8	40.8
Majority	72.8	78.4	76.8	79.2	97.0	27.4
Naive Bayes	65.1	70.7	82.0	78.5	96.0	70.7
Average	75.0	80.1	79.5	80.2	96.7	48.6
Minimum	73.3	75.9	51.6	82.2	93.0	16.2
Maximum	75.0	75.8	84.7	82.2	92.8	19.7
Product	75.4	77.1	60.4	81.2	97.0	22.2
Fuzzy integral	74.9	80.9	87.2	81.5	97.2	80.3
Decision templates (I1)	76.8	72.7	87.0	79.5	96.7	83.0
Decision templates (I2)	76.8	72.7	87.0	79.5	96.7	83.0
Decision templates (S1)	76.8	72.5	86.4	79.5	96.7	84.8
Decision templates (S2)	76.8	72.5	86.4	79.5	96.7	85.1
Decision templates (S3)	76.8	80.2	87.0	79.5	96.7	85.1
Decision templates (E)	77.1	76.9	86.5	80.2	96.8	85.4

TABLE V
TESTING RANKS FOR THE COMBINATION METHODS AND THE SIX DATA SETS (THE HIGHER THE RANK, THE BETTER THE METHOD)

Method	Pima	Phoneme	Cone-torus	Cleveland	Wisconsin	Satimage	Total
Weighted majority	8.0	14.0	7.0	2.0	10.0	5.0	46.0
Majority	2.0	10.0	3.0	3.0	12.5	4.0	34.5
Naive Bayes	1.0	1.0	5.0	1.0	3.0	7.0	18.0
Average	5.5	11.0	4.0	9.0	6.5	6.0	42.0
Minimum	3.0	7.0	1.0	13.5	2.0	1.0	27.5
Maximum	5.5	6.0	6.0	13.5	1.0	2.0	34.0
Product	7.0	9.0	2.0	11.0	12.5	3.0	44.5
Fuzzy integral	4.0	13.0	14.0	12.0	14.0	8.0	65.0
Decision templates (I1)	12.0	4.5	11.5	6.0	6.5	9.5	50.0
Decision templates (I2)	12.0	4.5	11.5	6.0	6.5	9.5	50.0
Decision templates (S1)	9.5	2.5	8.5	6.0	6.5	11.0	44.0
Decision templates (S2)	9.5	2.5	8.5	6.0	6.5	13.0	46.0
Decision templates (S3)	12.0	12.0	13.0	6.0	6.5	12.0	61.5
Decision templates (E)	14.0	8.0	10.0	10.0	11.0	14.0	67.0

TABLE VI
RESULTS FROM A PAIRWISE STATISTICAL COMPARISON OF COMBINATION METHODS. THE ENTRIES MEAN "[BETTER,SAME,WORSE]" OUT OF SIX COMPARISONS

	WMAJ	MAJ	NB	AVR	MIN	MAX	PRO	FI	DT(I1)	DT(I2)	DT(S1)	DT(S2)	DT(S3)	DT(E)
WMAJ	-	330	321	141	420	330	330	051	141	141	141	141	051	141
MAJ	033	-	222	042	420	321	240	033	132	132	132	132	033	042
NB	123	222	-	132	312	222	222	024	024	024	024	024	024	024
AVR	141	240	231	-	420	321	330	042	132	132	132	132	042	132
MIN	024	024	213	024	-	042	033	024	123	123	123	123	024	033
MAX	033	123	222	123	240	-	132	033	132	132	132	132	033	042
PRO	033	042	222	033	330	231	-	033	132	132	132	132	033	042
FI	150	330	420	240	420	330	330	-	141	141	141	141	051	141
DT(I1)	141	231	420	231	321	231	231	141	-	060	051	051	042	042
DT(I2)	141	231	420	231	321	231	231	141	060	-	051	051	042	042
DT(S1)	141	231	420	231	321	231	231	141	150	150	-	060	051	051
DT(S2)	141	231	420	231	321	231	231	141	150	150	060	-	051	051
DT(S3)	150	330	420	240	420	330	330	150	240	240	150	150	-	150
DT(E)	141	240	420	231	330	240	240	141	240	240	150	150	051	-

(FI) had higher ranks, the differences where they outperformed DT(S3) were not found to be significant.

B. *Effects of the Individual Accuracies, Their Variability, and Ensemble Diversity*

Can we decide which fuzzy or nonfuzzy combiner to use on a given data set? A closer look into the characteristics of the final ensembles designed by Boosting reveals the difficulty in doing so. Looking for clues, we summarized in Table VII some characteristics of the final ensembles of 15 classifiers (averaged over two cross-validation runs) for the six data sets.

The single best classifier was identified as the one with the highest training accuracy. Column 1 shows the testing accuracies of the best classifiers in the ensembles. Next to it, we show in brackets the iteration number in the boosting algorithm where

TABLE VII
CHARACTERISTICS OF THE FINAL ENSEMBLES OF 15 CLASSIFIERS DESIGNED BY BOOSTING

Data set	Single best [%] (iteration)	Ensemble average [%]	Ensemble standard deviation [%]	Q
Pima	75.26 (1)	59.03	9.95	0.0870
Phoneme	74.41 (1)	63.77	7.13	0.1535
Cone-torus	84.88 (2)	68.13	7.33	0.2992
Cleveland	77.89 (1)	70.74	5.47	0.4438
Wisconsin	96.49 (1)	89.57	4.94	0.6430
Satimage	64.70 (2)	27.49	14.05	-0.0685

this classifier was found. It is not surprising that the best overall accuracy is achieved by the first or the second classifier in the ensemble where we draw a training sample almost uniformly from the training set at hand. Later classifiers are more specialized on difficult part of the training data rendering low overall accuracy.

We calculated the averaged accuracy of the ensemble looking for a pattern on the improvement. The most pronounced improvement of fuzzy combiners over the nonfuzzy ones was found on the least accurate ensemble, the Satimage data. Using 36 features and accounting for six classes, makes the training of an MLP a difficult task. Judging by the best accuracy of 64.70%, the training has been trapped in local optima for all ensemble members.[4] By resulting in low-accuracy classifiers, however, this experiment highlights an interesting finding: Fuzzy combiners are particularly useful when the classifiers forming the ensemble are poorly trained or calibrated.

Fumera and Roli [17] suggest that the *imbalance* of the ensemble plays a role in its success. To account for this, we show in Table VII the standard deviations within the 15 classifiers. The highest variation in accuracy is exhibited by the Satimage data where the best improvement happens to be as well. However, this pattern is not consistent with the other data sets. Higher variance is not a guarantee that fuzzy combiners will be better than nonfuzzy combiners. Fig. 6(a) shows the scatter of the individual classification accuracies of the ensembles for the six data sets and Fig. 6(b) gives the sequence of individual accuracies across the AdaBoost iterations. Not surprisingly, the performances deteriorate. However, there is no principal difference between the deterioration pattern for the Phoneme data from the others which can explain the failure of DTs on this data set.

Diversity is another characteristic that is perceived to be of primary importance for the success of the ensemble. Based on our previous research [30] we chose to show the measure of diversity Q of the final ensembles. The lower the value of Q, the higher the diversity. Although best improvement was fonud at the lowest Q (greatest diversity), no consistent pattern can be observed which can indicate where fuzzy methods should be preferred.

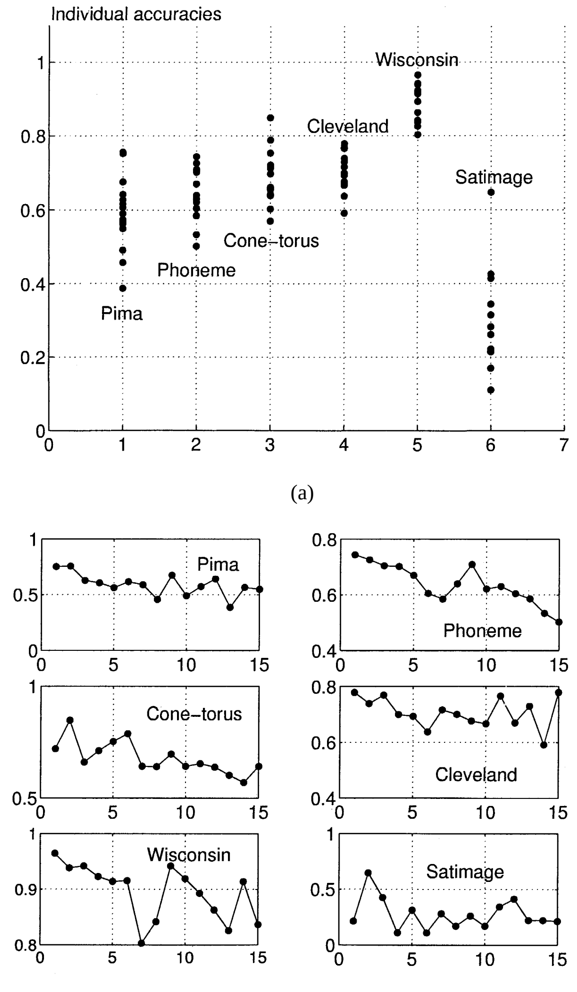


Fig. 6. (a) Scatter of the individual accuracies of the 15 classifiers for the six data sets. (b) Plot of the individual accuracies versus AdaBoost step.

C. Why are DTs Different From the Rest of the Combiners?

Below we try to give more insight into why DTs are expected to work. The combiners which treat the individual outputs separately, (one class at a time) are named in [29] “class-conscious.” Such are all basic combination methods, e.g., minimum, maximum, average, product, etc., and the fuzzy integral. Conversely, DTs are a “class-indifferent” approach because they treat the classifier outputs as a context-free set of features, much as the stacked generalization approach [44]. Thus, by design, all class-conscious combiners are idempotent, i.e., if the ensemble consists of L copies of a classifier D, the ensemble itself will be no different from D. Indeed, the decision profile will contain L identical rows and any of the operations discussed above will lead to the same overall class label as D. Decision templates,

[4] Woods *et al.* report in [45] an MLP with accuracy 83.98% using 5 features from the Satimage data but since no training protocol was specified we were not able to match this result.

however, will not be necessarily identical to D; they might be better or worse.

To illustrate this point, we consider a classifier D for a two-class data set. Denote the outputs for the two classes as $d_1 = \hat{P}(\omega_1|\mathbf{x})$ and $d_2 = 1 - d_1 = \hat{P}(\omega_2|\mathbf{x})$.[5] Taking L copies of D as our ensemble, the decision profile for $\mathbf{x}$ contains L rows of $[d_1 \;\; d_2]$. It is not difficult to verify that all combination methods explained above except decision templates will copy the decision of D as their final decision. However, this is not the case with DTs. Assume that we have obtained the following DTs:

$$DT_1 = \begin{bmatrix} 0.55 & & 0.45 \\ & \dots & \\ 0.55 & & 0.45 \end{bmatrix} \quad \text{and} \quad DT_2 = \begin{bmatrix} 0.2 & & 0.8 \\ & \dots & \\ 0.2 & & 0.8 \end{bmatrix}$$

and the decision of D for $\mathbf{x}$ is $d_1 = 0.4$ and $d_2 = 0.6$. All combination methods except DTs will assign $\mathbf{x}$ to class ω_2. Using, say DT(E), we have the two Euclidean distances

$$\begin{aligned} E_1 &= \sqrt{L \times ((0.55 - 0.40)^2 + (0.45 - 0.60)^2)} \\ &= \sqrt{0.045\, L}; \qquad (22) \\ E_2 &= \sqrt{L \times ((0.2 - 0.40)^2 + (0.8 - 0.60)^2)} \\ &= \sqrt{0.080\, L}. \qquad (23) \end{aligned}$$

Since $E_1 < E_2$ $\mathbf{x}$ will be classed in ω_1. Is this good or bad? The fact that a different classification is possible only supports the thesis that DTs are not an idempotent combiner. Hence, it is possible that the true label of $\mathbf{x}$ was ω_1, in which case DTs are correct where all other combiners, including D itself are wrong. The question is in what experimental scenario should we expect DTs to be more correct than other methods?

Let D be a linear classifier which we ran on the Cleveland data set, using a randomly selected half for training and the remaining half for testing. Every point $\mathbf{x} \in \Re^n$ can be characterized by its output values d_1 and d_2. Being a probabilistic label, (d_1, d_2) can be plotted as a point on the diagonal line of the unit square. A fuzzy label therefore will be a point within the unit square. Let us construct an ensemble taking L distorted copies $\tilde{D}$ of D, with output (d_1^3, d_2). Now the label points are off the diagonal line as shown in Fig. 7. All combination methods except DTs will label the points according to the bisecting line: The points whose label fall in the shaded area will be labeled in ω_2 because for these points $d_2 > d_1^3$, and so the combined result from any idempotent combiner will be $\mu_2 > \mu_1$ (matching that of $\tilde{D}$). The accuracy of the individual classifier in this example is 81.47%. Next, we apply DT(E). The two decision templates consist of L identical rows, therefore, they also can be characterized by points in $[0, 1]^2$. The two templates are depicted as crosses in Fig. 7. Since the support for the classes is now calculated by the distance to the template, a new decision boundary is found, shown by the dashed line. Four points from the original training set, previously mislabeled as ω_2 are now correctly labeled as ω_1 (encircled in the figure). Thus in this example, the training accuracy of DT(E) is 84.11%, exceeding that of the individual classifier and the other combination methods.

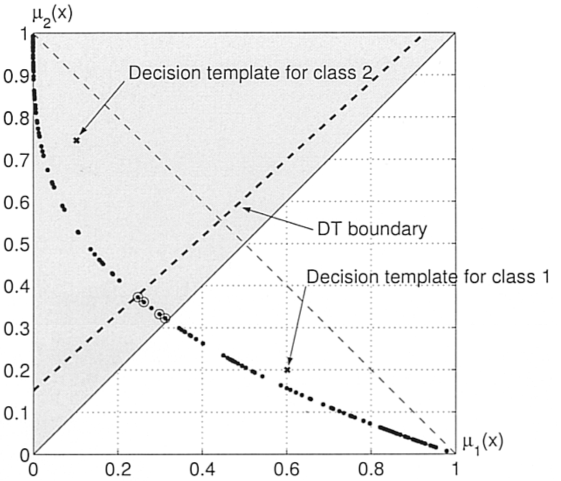


Fig. 7. Illustration of the decision templates operating on an ensemble of identical classifiers using Cleveland data set. Shaded area is the decision region for class ω_2 using the original classifier and any combination method other than DTs. The points in the training set are depicted using their labels from the "cloned" classifier. The two decision templates are shown with crosses and the respective new classification boundary is shown by the dashed line. Previously mislabeled points which are correctly labeled by the DT(E) are encircled.

D. Fewer Classifiers (Early Stopping of Adaboost)

In all previous experiments, we chose the number of classifiers ($L = 15$) arbitrarily. Fig. 5 suggests that reasonable accuracy of the combination is achieved by fuzzy combiners at the early stages of AdaBoost, with only a few classifiers. It could be argued that further increment in the number of classifiers dilutes the differences which are exploited well by the trainable fuzzy combiners such as DTs. To examine this hypothesis we analyzed a "slice" of AdaBoost stopped after the third classifier was built. Tables VIII–X mirror Tables IV–VI, respectively, for $L = 3$ classifiers.

Again, the fuzzy methods dominate the nonfuzzy methods examined. Two interesting observations can be made from the rank score Table IX. First, the average combiner was ranked very high, almost catching up with its best rivals in the comparison. This shows once again its robustness and universality. Second, the total rank scores of fuzzy integral and weighted majority dropped while the rank scores for all the decision template models went up. This indicates that DTs are more beneficial for small number of classifiers. The reason for that could be that, having to estimate $c^2 \times L$ parameters from the training set, DTs are prone to overtraining. Therefore, for larger L, it is advisable to have a separate validation set for calculating the c decision templates. Note that (Table X) now both DT(S3) and DT(E) are nondominated.

Fig. 8 offers an overall picture of the statistical comparisons. We grouped the fuzzy and the nonfuzzy methods and found a total of the number of times fuzzy methods dominated nonfuzzy methods, the number of times they were indistinguishable, and the number of times fuzzy methods were worse than nonfuzzy methods. Since there are seven fuzzy and seven nonfuzzy methods and six data sets, the total number of "fuzzy versus nonfuzzy" comparisons is $7 \times 7 \times 6 = 294$. Plotted

[5] The probabilistic semantic is used for illustration purposes. The example generalizes to any other semantic of the classifier outputs.

TABLE VIII
TESTING ACCURACY FOR ENSEMBLES OF $L = 3$ CLASSIFIERS

Method	Pima	Phoneme	Cone-torus	Cleveland	Wisconsin	Satimage
Weighted majority	75.7	77.9	85.8	80.9	97.2	52.7
Majority	76.2	76.0	85.8	80.9	97.2	63.9
Naive Bayes	72.0	70.7	83.4	80.9	97.2	70.1
Average	76.8	76.8	86.8	82.2	97.5	68.3
Minimum	76.4	75.5	57.4	81.8	96.5	16.5
Maximum	75.8	75.0	86.5	81.5	96.7	68.2
Product	76.8	75.3	67.2	82.5	97.5	26.4
Fuzzy integral	76.3	76.2	86.1	81.2	97.2	67.4
Decision templates (I1)	75.5	76.1	87.2	81.8	97.5	74.4
Decision templates (I2)	75.5	76.1	87.2	81.8	97.5	74.4
Decision templates (S1)	75.7	76.1	87.4	81.8	97.4	79.7
Decision templates (S2)	75.7	76.1	86.9	81.8	97.4	79.5
Decision templates (S3)	76.2	78.4	87.0	81.8	97.5	79.6
Decision templates (E)	76.7	77.4	86.6	82.2	97.7	79.2

TABLE IX
TESTING RANKS FOR ENSEMBLES OF $L = 3$ CLASSIFIERS

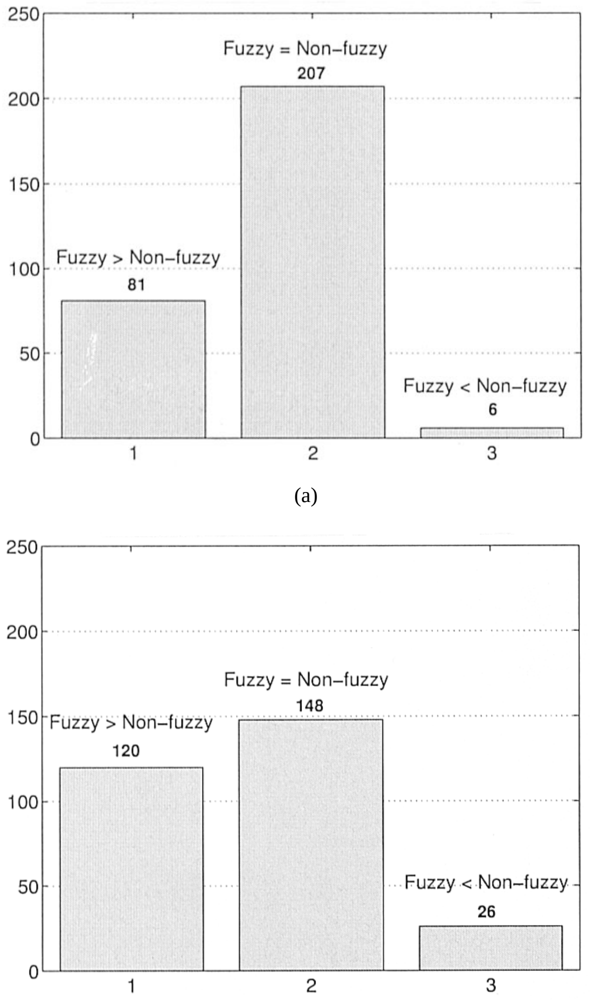
Method	Pima	Phoneme	Cone-torus	Cleveland	Wisconsin	Satimage	Total
Weighted majority	4.0	13.0	4.5	2.0	4.5	3.0	31.0
Majority	8.5	5.0	4.5	2.0	4.5	4.0	28.5
Naive Bayes	1.0	1.0	3.0	2.0	4.5	8.0	19.5
Average	13.5	11.0	9.0	12.5	11.5	7.0	64.5
Minimum	11.0	4.0	1.0	11.0	1.0	1.0	29.0
Maximum	7.0	2.0	7.0	5.0	2.0	6.0	29.0
Product	13.5	3.0	2.0	14.0	9.0	2.0	43.5
Fuzzy integral	10.0	10.0	6.0	4.0	4.5	5.0	39.5
Decision templates (I1)	2.5	8.5	12.5	8.0	11.5	9.5	52.5
Decision templates (I2)	2.5	8.5	12.5	8.0	11.5	9.5	52.5
Decision templates (S1)	5.5	6.5	14.0	8.0	7.5	14.0	55.5
Decision templates (S2)	5.5	6.5	10.0	8.0	7.5	12.0	49.5
Decision templates (S3)	8.5	14.0	11.0	8.0	11.5	13.0	66.0
Decision templates (E)	12.0	12.0	8.0	12.5	14.0	11.0	69.5

TABLE X
RESULTS FOR ENSEMBLES OF $L = 3$ CLASSIFIERS FROM A PAIRWISE STATISTICAL COMPARISON OF COMBINATION METHODS. THE ENTRIES MEAN “[BETTER, SAME, WORSE]” OUT OF 6 COMPARISONS

	WMAJ	MAJ	NB	AVR	MIN	MAX	PRO	FI	DT(I1)	DT(I2)	DT(S1)	DT(S2)	DT(S3)	DT(E)
WMAJ	-	141	141	051	330	141	330	141	141	141	141	141	051	051
MAJ	141	-	141	051	240	051	240	051	051	051	051	051	042	051
NB	141	141	-	132	222	141	222	141	033	033	033	033	033	033
AVR	150	150	231	-	240	150	240	060	051	051	051	051	042	051
MIN	033	042	222	042	-	042	042	042	042	042	042	042	033	033
MAX	141	150	141	051	240	-	240	060	051	051	051	051	042	042
PRO	033	042	222	042	240	042	-	042	042	042	042	042	033	033
FI	141	150	141	060	240	060	240	-	051	051	051	051	042	051
DT(I1)	141	150	330	150	240	150	240	150	-	060	051	051	042	051
DT(I2)	141	150	330	150	240	150	240	150	060	-	051	051	042	051
DT(S1)	141	150	330	150	240	150	240	150	150	150	-	060	051	060
DT(S2)	141	150	330	150	240	150	240	150	150	150	060	-	051	060
DT(S3)	150	240	330	240	330	240	330	240	240	240	150	150	-	060
DT(E)	150	150	330	150	330	240	330	150	150	150	060	060	060	-

in Fig. 8(a) is a bar graph for the early stopping of AdaBoost ($L = 3$ classifiers), and in Fig. 8(b), the bar graph for $L = 15$ classifiers. In both experiments, fuzzy combiners were found better. Part of the large number of insignificant differences for $L = 3$ (207) was “redistributed” for $L = 15$, again favoring fuzzy combiners.

The results in the plots should be taken cautiously because of the nature of the methods grouped together. We chose for the nonfuzzy group the basic and most popular combiners. However, we could have included in this group other successful trainable combiners from the literature. One such candidate would be the so called behavior knowledge space method [22] which was



(a)

(b)

Fig. 8. Overall picture of the statistical comparisons.

found to be better than DTs in an identity verification study [23]. BKS is a multinomial classifier on the class label outputs which requires a look-up table of size L^c which makes it inappropriate for even moderate size ensembles. Apart from the danger of severe overtraining, the computational time would be substantial.

VI. Conclusion

We studied the potential of fuzzy combination methods for ensembles of classifiers designed by AdaBoost. The results involving sums of ranks and statistical comparisons showed that in general, fuzzy methods fared better than nonfuzzy methods. However, the study also highlighted the difficulties in choosing a particular method for a given problem.

Decision templates were found to be the best from the group of the fuzzy combiners, particularly the variants with Euclidean distance (DT(E)) and the similarity measure S_3 (DT(S3)). The capability of DTs to achieve higher accuracy can be attributed to the fact that they are not an idempotent combiner. Fuzzy integral also showed consistently good performance, working successfully on the data sets where DTs were inferior to the nonfuzzy methods. DT models were more prone to overtraining than the other trainable combiners studied here, i.e., fizzy integral and weighted majority vote. Thus, for ensembles of three classifiers, DT's performance was superior to the other two, whereas for 15 classifiers the performances were similar.

From the nonfuzzy group, the weighted majority vote was the best combiner. This is not a surprise as this combiner is the standard one used with AdaBoost. Average combination was also among the best, especially for small number of classifiers.

Thus, the claim here is not that the fuzzy combiners are better. A well known postulate in pattern recognition says that there is no "best" classifier or "best" combination method. What this study suggests is to keep fuzzy combiners high on the list of options.

References

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

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

[3] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants," *Machine Learning*, vol. 36, pp. 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 Anal., Theory, Meth., Applicat.*, vol. 30, no. 3, pp. 1323–1334, 1997.

[5] I. Bloch, "Information combination operators for data fusion: A comparative review with classification," *IEEE Trans. Syst., Man, Cybern. A*, vol. 26, pp. 52–67, Feb. 1996.

[6] B. Bouchon-Meunier, M. Rifqi, and S. Bothorel, "Toward general measures of comparison of objects," *Fuzzy Sets Syst.*, vol. 84, no. 2, pp. 143–153, 1996.

[7] L. Breiman, "Combining predictors," in *Combining Artificial Neural Nets*, A.J.C. Sharkey, Ed. New York: Springer-Verlag, 1999, pp. 31–50.

[8] S.-B. Cho and J. H. Kim, "Combining multiple neural networks by fuzzy integral and robust classification," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, pp. 380–384, Feb. 1995.

[9] S. B. Cho and J. H. Kim, "Multiple network fusion using fuzzy logic," *IEEE Trans. Neural Networks*, vol. 6, pp. 497–501, Mar. 1995.

[10] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 7, no. 10, pp. 1895–1924, 1998.

[11] ——, "Ensemble methods in machine learning," in *Multiple Classifier Systems*, J. Kittler and F. Roli, Eds. New York: Springer-Verlag, 2000, vol. 1857, Lecture Notes in Computer Science, pp. 1–15.

[12] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*. New York: Academic, 1980.

[13] ——, "A review of fuzzy set aggregation connectives," *Inform. Sci.*, vol. 36, pp. 85–121, 1985.

[14] ——, "The three semantics of fuzzy sets," *Fuzzy Sets Syst.*, vol. 90, pp. 141–150, 1997.

[15] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed, NY: Wiley , 2001.

[16] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[17] G. Fumera and F. Roli, "Performance analysis and comparison of linear combiners for classifier fusion," presented at the 16th Int. Conf. Pattern Recognition, 2002.

[18] P. D. Gader, M. A. Mohamed, and J. M. Keller, "Fusion of handwritten word classifiers," *Pattern Recogn. Lett.*, vol. 17, pp. 577–584, 1996.

[19] M. Grabisch, "On equivalence classes of fuzzy connectives — the case of fuzzy integrals," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 96–109, Feb. 1995.

[20] M. Grabisch and M. Sugeno, "Multi-attribute classification using fuzzy integral," *Proc. IEEE Int. Conf. Fuzzy Systems*, pp. 47–54, 1992.

[21] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 993–1001, Oct. 1990.

[22] Y. S. Huang and C. Y. Suen, "A method of combining multiple experts for the recognition of unconstrained handwritten numerals," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 90–93, Jan. 1995.

[23] J. Kittler, M. Ballette, J. Czyz, F. Roli, and L. Vandendorpe, "Decision level fusion of intramodal personal identity verification experts," in *Proc. 3rd Int. Workshop Multiple Classifier Systems*, vol. 2364, Lecture Notes in Computer Science, F. Roli and J. Kittler, Eds., Cagliari, Italy, 2002, pp. 314–324.

[24] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 226–239, Mar. 1998.

[25] *Proc. 1st Int. Workshop Multiple Classifier Systems*, vol. 1857, Lecture Notes In Computer Science, J. Kittler and F. Roli, Eds., Cagliari, Italy, 2000.

[26] *Proc. 2nd Int. Workshop Multiple Classifier Systems*, vol. 2096, Lecture Notes In Computer Science, J. Kittler and F. Roli, Eds., Cambridge, U.K., 2001.

[27] L. I. Kuncheva, "Combining classifiers: Soft computing solutions," in *Pattern Recognition, From Classical to Modern Approaches*, S.K. Pal and A. Pal, Eds. Singapore: World Scientific, 2001, ch. 15, pp. 427–452.

[28] ——, "A theoretical study on expert fusion strategies," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, pp. 281–286, Feb. 2002.

[29] L. I. Kuncheva, J. C. Bezdek, and R. P. W. Duin, "Decision templates for multiple classifier fusion: An experimental comparison," *Pattern Recogn.*, vol. 34, no. 2, pp. 299–314, 2001.

[30] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles," *Machine Learn.*, vol. 51, pp. 181–207, 2003.

[31] L. Lam and A. Krzyzak, "A theoretical analysis of the application of majority voting to pattern recognition," in *Proc. 12th Int. Conf. Pattern Recognition*, Jerusalem, Israel, 1994, pp. 418–420.

[32] L. Lam and C. Y. Suen, "Application of majority voting to pattern recognition: An analysis of its behavior and performance," *IEEE Trans. Syst., Man, Cybern.*, vol. 27, pp. 553–568, Oct. 1997.

[33] W. Pierce, "Improving reliability of digital systems by redundancy and adaptation," Ph.D. dissertation, Dept. Elect. Eng., Stanford Univ., Stanford, CA, 1961.

[34] *Proc. 3rd Int. Workshop Multiple Classifier Systems*, vol. 2364, Lecture Notes In Computer Science, F. Roli and J. Kittler, Eds., Cagliari, Italy, 2002.

[35] D. Ruta and B. Gabrys, "A theoretical analysis of the limits of majority voting errors for multiple classifier systems," Dept. Comput. Inform. Syst., Univ. Paisley, Paisley, U.K., Tech. Rep. 11, ISSN 1461-6122, 2000.

[36] R. E. Schapire, "Theoretical views of boosting," in *Proc. 4th Eur. Conf. Computational Learning Theory*, 1999, pp. 1–10.

[37] ——, "The boosting approach to machine learning. An overview," presented at the MSRI Workshop Nonlinear Estimation and Classification, 2002.

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

[39] K. Tumer and J. Ghosh, "Analysis of decision boundaries in linearly combined neural classifiers," *Pattern Recogn.*, vol. 29, no. 2, pp. 341–348, 1996.

[40] ——, "Error correlation and error reduction in ensemble classifiers," *Connection Sci.*, vol. 8, no. 3/4, pp. 385–404, 1996.

[41] ——, "Linear and order statistics combiners for pattern classification," in *Combining Artificial Neural Nets*, A.J.C. Sharkey, Ed, U.K.: Springer-Verlag, 1999, pp. 127–161.

[42] A. Verikas, A. Lipnickas, K. Malmqvist, M. Bacauskiene, and A. Gelzinis, "Soft combination of neural classifiers: A comparative study," *Pattern Recogn. Lett.*, vol. 20, pp. 429–444, 1999.

[43] 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 Trans. Syst., Man, Cybern.*, vol. 28B, pp. 583–591, Aug. 1998.

[44] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–260, 1992.

[45] K. Woods, W. P. Kegelmeyer, and K. Bowyer, "Combination of multiple classifiers using local accuracy estimates," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 405–410, Apr. 1997.

[46] R. R. Yager, "On a general class of fuzzy connectives," *Fuzzy Sets Syst.*, vol. 4, pp. 235–242, 1980.

[47] ——, "Ordered weighted averaging operators in multicriteria decision making," *IEEE Trans. Syst., Man, Cybern.*, vol. 18, pp. 183–193, Feb. 1988.

Ludmila I. Kuncheva received the M.Sc. degree from the Technical University, Sofia, Bulgaria, in 1982, and the Ph.D. degree from the Bulgarian Academy of Sciences in 1987.

Until 1997, she worked at the Central Laboratory of Biomedical Engineering, Bulgarian Academy of Sciences, as a Senior Research Associate. She is currently a Senior Lecturer at the School of Informatics, University of Wales, Bangor, U.K. Her interests include pattern recognition, classifier combination, diversity measures, fuzzy classifiers, and prototype classifiers.