Published in: *Pattern Recognition*,
34, (2), 2001, 299–314.

Decision Templates for Multiple Classifier Fusion: An Experimental Comparison

Ludmila I. Kuncheva
School of Mathematics, University of Wales, Bangor
Bangor, Gwynedd LL57 1UT, United Kingdom, (*l.i.kuncheva@bangor.ac.uk*)

James C. Bezdek*
Department of Computer Science, University of West Florida
Pensacola, FL 32514, USA, (*jbezdek@argo.cs.uwf.edu*)

Robert P.W. Duin
Faculty of Applied Sciences, Delft University of Technology
P.O. Box 5046, 2600 GA Delft, The Netherlands, (*duin@ph.tn.tudelft.nl*)

Abstract

Multiple classifier fusion may generate more accurate classification than each of the constituent classifiers. Fusion is often based on fixed combination rules like the product and average. Only under strict probabilistic conditions can these rules be justified. We present here a simple rule for adapting the class combiner to the application. c decision templates (one per class) are estimated with the same training set that is used for the set of classifiers. These templates are then matched to the decision profile of new incoming objects by some similarity measure. We compare 11 versions of our model with 14 other techniques for classifier fusion on the Satimage and Phoneme datasets from the database ELENA. Our results show that decision templates based on *integral* type measures of similarity are superior to the other schemes on both data sets.

Keywords: Classifier fusion, Combination of multiple classifiers, Decision templates, Fuzzy similarity, Behavior-Knowledge-Space, Fuzzy integral, Dempster-Shafer, Class-conscious fusion, Class-indifferent fusion.

1 Introduction

Combining classifiers to achieve higher accuracy is an important research topic with different names in the literature:

- combination of multiple classifiers ([1, 2, 3, 4, 5]);
- classifier fusion ([6, 7, 8, 9, 10]);
- mixture of experts ([11, 12, 13, 14]);
- committees of neural networks ([15, 16]);
- consensus aggregation ([17, 18, 19]);
- voting pool of classifiers ([20]);
- dynamic classifier selection ([3]);

*Research supported by ONR grant N00014-96-1-0642

- composite classifier system ([21]);
- classifier ensembles ([16, 22]),
- divide-and-conquer classifiers [23];
- pandemonium system of reflective agents [24];
- change-glasses approach to classifier selection [25], etc.

The paradigms of these models differ on the: assumptions about classifier dependencies; type of classifier outputs; aggregation strategy (global or local); aggregation procedure (a function, a neural network, an algorithm), etc.

There are generally two types of combination: classifier selection and classifier fusion ([3]). The presumption in classifier *selection* is that each classifier is "an expert" in some local area of the feature space. When a feature vector $\mathbf{x} \in \Re^p$ is submitted for classification, the classifier responsible for the vicinity of $\mathbf{x}$ is given the highest credit when assigning the class label to $\mathbf{x}$. We can nominate exactly one classifier to make the decision, as in [26], or more than one "local expert", as in [27, 11]. Classifier *fusion* assumes that *all* classifiers are trained over the whole feature space, and are thereby considered as competitive rather than complementary [18, 4].

Multiple classifier outputs are usually made comparable by scaling them to the [0,1] interval. For some classifiers these values can be treated as classifier-conditional posterior probabilities for the classes [28]. In some cases, e.g., undertrained or overtrained neural networks, the probabilistic interpretation does not make sense. Furthermore, the probabilistic interpretation [5, 12] does not lead very far without some assumptions, which may appear unrealistic and restrictive, e.g., that the individual classifiers use mutually independent subsets of features, or commit independent misclassification errors. Under such assumptions, fusion often reduces to simple aggregation operators such as the product or average [5], or the so-called "Naive Bayes" rule explained in Section 4.

The more general interpretation of classifier outputs as the *support* for the classes is the basis of fuzzy aggregation methods, examples of which are simple connectives between fuzzy sets, the fuzzy integral [6, 29, 7, 30, 9, 31], and Dempster-Shafer fusion [32, 2, 4].

There is another way to look at the fusion problem: we can treat the classifier outputs simply as the input to a second-level classifier, and use classical pattern recognition techniques for the second level design [33]. The use of traditional feature-based classifiers in this approach is difficult because the class distributions in the intermediate feature space are not well-behaved (there will be many points in the regions close to 0 and 1, and very few in-between). So, simple classifiers like linear and quadratic discriminant functions that assume normal distributions will fail.

The method developed here is based on a set of c matrices called *decision templates* (DTs). DTs are a robust classifier fusion scheme that combines classifier outputs by comparing them to a characteristic template for each class. DT fusion uses *all* classifier outputs to calculate the final support for each class, which is in sharp contrast to most other fusion methods which use *only the support for that particular class* to make their decision.

Section 2 introduces the formalism of classifier fusion. In Section 3 we present DT schemes with 11 measures of similarity. Section 4 describes the algorithmic details of some simple aggregation schemes; *Naive-Bayes* (NB), *Behavior-Knowledge Space* (BKS), *Dempster-Shafer* (DS), and *Fuzzy integral* (FI). Section 5 contains our experiments with the 2 data sets (Satimage and Phoneme); Section 6, the discussion; and Section 7, a summary.

2 Classifier fusion

Let $\mathbf{x} \in \Re^n$ be a feature vector and $\{1, 2, \ldots, c\}$ be the label set of c classes. We call a *classifier* every mapping

$$D : \Re^n \to [0,1]^c - \{\mathbf{0}\},$$

where $\mathbf{0} = [0, 0, \ldots, 0]^T$ is the origin of $\Re^c$. We call the output of D a "class label" and denote it by $\mu_D(\mathbf{x}) = [\mu_D^1(\mathbf{x}), \ldots, \mu_D^c(\mathbf{x})]^T, \mu_D^i(\mathbf{x}) \in [0,1]$. The components $\{\mu_D^i(\mathbf{x})\}$ can be regarded as (estimates of)

the posterior probabilities for the classes, given $\mathbf{x}$, i.e. $\mu_D^i(\mathbf{x}) = P(i|\mathbf{x})$. Alternatively, $\mu_D^i(\mathbf{x})$ can be viewed as typicalness, belief, certainty, possibility, etc. Bezdek et al. [34] define three types of classifiers:

1. *Crisp* classifier: $\mu_D^i(\mathbf{x}) \in \{0,1\}, \sum_{i=1}^{c} \mu_D^i(\mathbf{x}) = 1, \forall \mathbf{x} \in \Re^n$;
2. *Fuzzy* classifier: $\mu_D^i(\mathbf{x}) \in [0,1], \sum_{i=1}^{c} \mu_D^i(\mathbf{x}) = 1, \forall \mathbf{x} \in \Re^n$; (Probabilistic interpretation of the outputs falls in this category)
3. *Possibilistic* classifier: $\mu_D^i(\mathbf{x}) \in [0,1], \sum_{i=1}^{c} \mu_D^i(\mathbf{x}) > 0, \forall \mathbf{x} \in \Re^n$;

The decision of D can be "hardened" so that a crisp class label in $\{1, 2, \ldots, c\}$ is assigned to $\mathbf{x}$. This is typically done by the *maximum membership rule*:

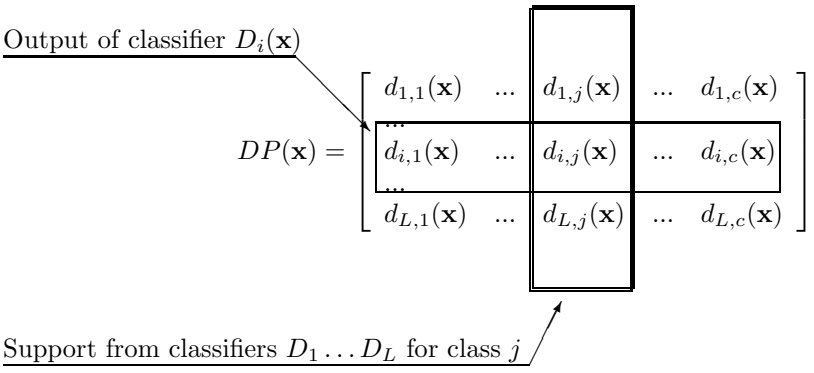
$$D(\mathbf{x}) = k \quad \Longleftrightarrow \quad \mu_D^k(\mathbf{x}) = \max_{i=1,\ldots,c} \left\{ \mu_D^i(\mathbf{x}) \right\}. \tag{1}$$

Let $\{D_1, \ldots, D_L\}$ be the set of L classifiers. We denote the output of the ith classifier as $D_i(\mathbf{x}) = [d_{i,1}(\mathbf{x}), \ldots, d_{i,c}(\mathbf{x})]^T$, 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 j. We construct $\hat{D}$, the fused output of the L first-level classifiers as

$$\hat{D}(\mathbf{x}) = \mathcal{F}(D_1(\mathbf{x}), \ldots, D_L(\mathbf{x})), \tag{2}$$

where $\mathcal{F}$ is called *aggregation rule*.

The classifier outputs can be organized in a **decision profile** (DP) as the matrix



(3)

Some methods calculate the support for class i ($\mu_{\hat{D}}(\mathbf{x})$) using only the ith column of $DP(\mathbf{x})$. Fusion methods that use the DP class-by-class will be called **class-conscious**. Examples of class-conscious fusion operators are discussed in Section 4: average, minimum, maximum, product, fuzzy integral, etc.

The choice of an aggregation operator $\mathcal{F}$ depends on the interpretation of $d_{i,j}(\mathbf{x}), i = 1, \ldots, L, j = 1, \ldots, c$. We can regard $d_{i,j}(\mathbf{x})$ as an estimate of the posterior probability $P(j|\mathbf{x})$ produced by classifier D_i (denoted $\hat{P}_i(j|\mathbf{x}), i = 1, \ldots, L,\ j = 1, \ldots, c$). Optimal (in the Bayesian sense) combination of these estimates is not straightforward. Kittler et al. [5] show two (different!) ways of combining such estimates starting from the same independence assumption. For many classifiers, the estimates $\{\hat{P}_i(j|\mathbf{x})\}$ can have both large bias and variance, which, together with the independence assumption, can invalidate the probabilistic approach.

Another approach is to use *all of* $DP(\mathbf{x})$ to calculate the support for each class. We call the range of the classifier outputs (decision profile matrices), $[0,1]^{(L \cdot c)} - \mathbf{0} \subset \Re^{(L \cdot c)}$, **intermediate feature space**. Each vector in this set is an "expanded" version of the DP matrix obtained by concatenating its L rows. The problem now is to design the second (fusion) stage of the classifier using the intermediate features, disregarding the matrix context. Fusion methods in this group will be called **class-indifferent**. Here we can use any classifier with the intermediate features as inputs and the class label $\hat{D}(\mathbf{x})$ as the output.

The difficulty comes from the specific structure of the intermediate feature space. If all L classifiers are perfect (produce the right crisp class label for every $\mathbf{x}$), then there will be no variance of the values of $DP(\mathbf{x})$ over the subset of the data set from class i. The covariance matrices for the classes (or the single covariance matrix for all classes) are singular. Classifiers such as linear and quadratic discriminant classifiers, which are based on the assumption of normally distributed classes, will fail when trying to estimate and invert

the covariance matrices. To get high overall accuracy, we try to use the most accurate individual classifiers. The higher the accuracy of all classifiers, the more likely it is that the covariance matrix of the intermediate features will be close to singular.

As an example, consider two classifiers D_1 and D_2 giving the values in Table 1 for 10 objects in $c = 2$ classes. from a certain data set, labeled in class 1. The values were generated at random and independently, so that the expected support for class 1 from D_1 is in [0.9, 1.0], and from classifier D_2, in [0.8, 1.0]. These values form the data set in the intermediate space. The mean is

$$\mathbf{m} = (0.9475, 0.0525, 0.8830, 0.1170)^T,$$

and the covariance matrix is

$$S = \begin{bmatrix} 0.0007 & -0.0007 & -0.0004 & 0.0004 \\ -0.0007 & 0.0007 & 0.0004 & -0.0004 \\ -0.0004 & 0.0004 & 0.0019 & -0.0019 \\ 0.0004 & -0.0004 & -0.0019 & 0.0019 \end{bmatrix}.$$

Table 1: Intermediate feature values for class i from classifiers D_1 and D_2

.

D_1		D_2	
Class 1	Class 2	Class 1	Class 2
0.9807	0.0193	0.9007	0.0993
0.9318	0.0682	0.8200	0.1800
0.9697	0.0303	0.8357	0.1643
0.9458	0.0542	0.8710	0.1290
0.9849	0.0151	0.8364	0.1636
0.9302	0.0698	0.8680	0.1320
0.9622	0.0378	0.9316	0.0684
0.9140	0.0860	0.9421	0.0579
0.9146	0.0854	0.9318	0.0682
0.9406	0.0594	0.8932	0.1068

To calculate a linear or quadratic discriminant function we have to invert the covariance matrix S. In Matlab, a warning is displayed that S is close to singular or badly scaled and that the results may be inaccurate. The following matrix results:

$$S^{-1} = \begin{bmatrix} 4.6117 & 4.6117 & 0.0000 & 0.0000 \\ 4.6117 & 4.6117 & -0.0000 & 0 \\ 0.0000 & 0 & 4.6117 & 4.6117 \\ 0.0000 & 0.0000 & 4.6117 & 4.6117 \end{bmatrix} \quad \underline{\mathbf{1.0e+18}}.$$

In our experiments we tried linear, quadratic, and logistic classifiers, and Fisher's discriminant as the fusion classifier $\hat{D}$.

Notice the difference between the class-conscious and class-indifferent groups of methods. The former use the context of the DP but disregard part of the information, using *only one column per class*. Class-indifferent methods use the whole DP but disregard the context (which might be useful). In this paper we propose a middle ground framework that makes use of both approaches.

In our approach it is assumed that we know the desirable DP for each class in advance. Consider an example with $L = 3$ and $c = 4$. Presumably, the most desirable decision profile for class 3 is the "crisp" decision profile shown in Table 2. Then the aggregation rule $\mathcal{F}$ in (2) can be used to measure the correspondence of the current $DP(\mathbf{x})$ to the "model" for class i, $i = 1, \ldots, c$.

Some popular fusion methods, like majority vote and naive Bayes, require crisp labels. To use them we first need to harden the decisions of $D_1, \ldots, D_L$. Some fusion schemes, like simple fuzzy aggregation connectives, do not require any additional training, i.e., once the individual classifiers are ready, the fusion can

Table 2: Most desirable (presumably) decision profile for class 3

class →	1	2	3	4
$D_1(\mathbf{x})$	0	0	1	0
$D_2(\mathbf{x})$	0	0	1	0
$D_3(\mathbf{x})$	0	0	1	0

Table 3: Classifier fusion techniques

First level output↓	Training at fusion level	
	No	Yes
Crisp	**C1:** Majority [35]	**C2:** Behavior-Knowledge Space [36] "Naive" Bayes [4]
Soft	**CC1:** Min, Max, OWA [37], Average, Product, [38, 5]	**CC2:** Probabilistic product [39, 40] Fuzzy integral [6, 7, 9], Trained linear combinations [41, 42, 43], **CI2** LDC, QDC, Fisher Logistic classifier Neural networks [44, 13], Dempster-Shafer [32, 2, 4], **Decision Templates**

be performed right away. Others, like the fuzzy integral and the probabilistic product, train a small number of parameters. Table 3 gives our grouping of classifier fusion methods divided by the absence/presence of parameters to train at the fusion level, type of classifier outputs, and the way DP is used. Acronym **CC** denotes class-conscious fusion methods, **CI**, class-indifferent methods, and **C**, fusion methods that require crisp class labels from the individual classifiers.

3 Decision templates (DT)

Let $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$, $\mathbf{z}_j \in \Re^n$, be the crisply labeled training data set.

Definition. *The* **decision template** $DT_i(Z)$ *of class* i *is the* $L \times c$ *matrix* $DT_i(Z) = [dt_i(k,s)(Z)]$ *whose* (k,s)*th element is computed by*

$$dt_i(k,s)(Z) = \frac{\sum_{j=1}^{N} Ind(\mathbf{z}_j, i)\; d_{k,s}(\mathbf{z}_j)}{\sum_{j=1}^{N} Ind(\mathbf{z}_j, i)}, \quad k = 1, \ldots, L, \quad s = 1, \ldots, c, \tag{4}$$

where $Ind(\mathbf{z}_j, i)$ *is an indicator function with value 1 if* $\mathbf{z}_j$ *has crisp label* i*, and 0, otherwise [45].* To simplify the notation $DT_i(Z)$ will be denoted by DT_i.

The decision template DT_i for class i is the average of the decision profiles of the elements of the training set Z labeled in class i. When $\mathbf{x} \in \Re^n$ is submitted for classification, the DT scheme matches $DP(\mathbf{x})$ to $DT_i, i = 1, \ldots, c$, and produces the soft class labels

$$\mu_D^i(\mathbf{x}) = \mathcal{S}(DT_i, DP(\mathbf{x})), \quad i = 1, \ldots, c, \tag{5}$$

where $\mathcal{S}$ is interpreted as a *similarity* measure. The higher the similarity between the decision profile of the current $\mathbf{x}$ ($DP(\mathbf{x})$) and the decision template for class i (DT_i), the higher the support for that class ($\mu^i_{\hat{D}}(\mathbf{x})$). Notice that we use the word "similarity" in a broad sense, meaning "degree of match" or "likeliness", etc. Among the measures of similarity that we consider are 4 (proper) measures of similarity, 5 inclusion indices, and one consistency measure. However, there is no reason to prefer these. Since the general idea is to compare the matrix $DP(\mathbf{x})$ to c template matrices ($DT_1, \ldots, DT_c$), any measure that does this might be appropriate. Figure 1 illustrates how the DT scheme operates. The decision templates are calculated in advance using Z in equation (4).

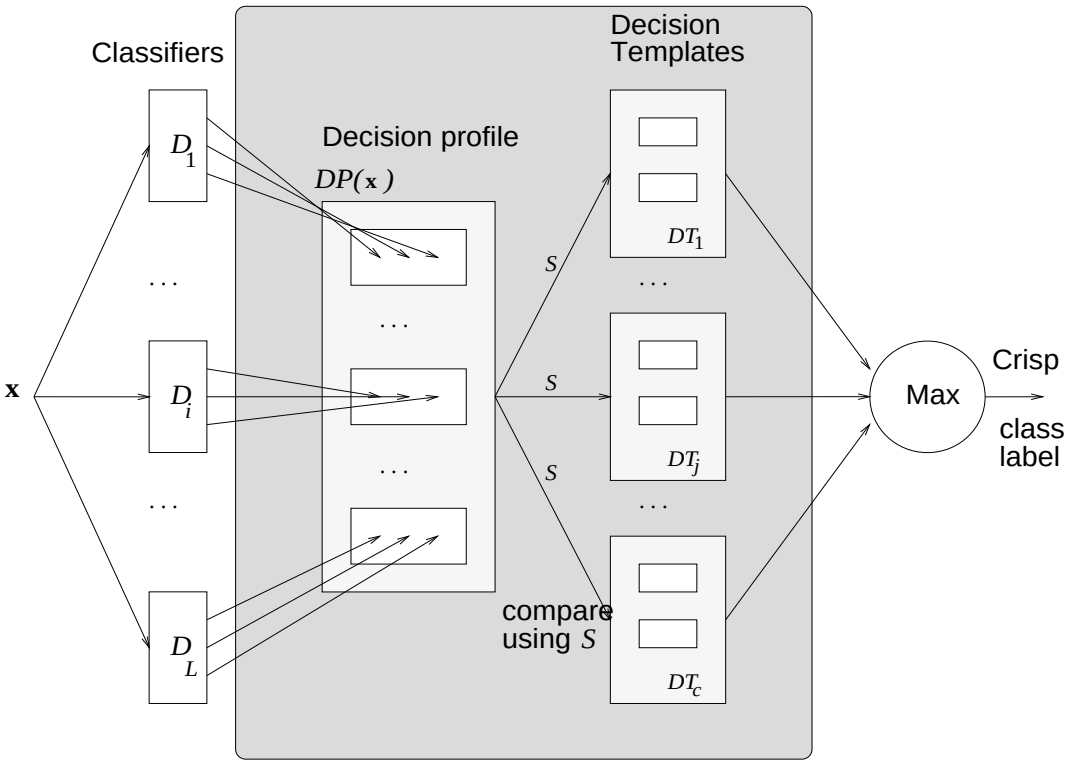


Figure 1: Architecture of the decision templates classifier fusion scheme

Regarding the arguments of $\mathcal{S}$ as fuzzy sets on some universal set with $L \cdot c$ elements, various fuzzy measures of similarity can be used. Let A and B be fuzzy sets on $U = \{u_1, \ldots, u_n\}$. In this study we used the following four proper *measures of similarity* [46]:

$$S_1(A, B) \equiv \frac{\| A \cap B \|}{\| A \cup B \|}, \tag{6}$$

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

$$\| \zeta \| = \frac{1}{n} \sum_{i=1}^{n} \mu_\zeta(u_i). \tag{7}$$

$$S_2(A, B) \equiv 1 - \| A \nabla B \|, \tag{8}$$

where $A \nabla B$ is the symmetric difference defined by the Hamming distance

$$\mu_{A \nabla B}(u) = |\mu_A(u) - \mu_B(u)|. \tag{9}$$

$$S_3(A, B) \equiv 1 - \| A \Delta B \|, \tag{10}$$

where

$$\mu_{A\Delta B}(u) = \max\{\mu_{A\cap\bar{B}}(u), \mu_{\bar{A}\cap B}(u)\}. \tag{11}$$

$$S_4(A,B) \equiv 1 - \sup_{u\in U} \{\mu_{A\nabla B}(u)\}. \tag{12}$$

We also used the following 5 *indices of inclusion* of A in B [46]

$$I_1(A,B) \equiv \frac{\parallel A\cap B \parallel}{\parallel A \parallel}. \tag{13}$$

$$I_2(A,B) \equiv 1 - \parallel A| - |B \parallel. \tag{14}$$

where $|-|$ is the bounded difference

$$\mu_{A|-|B}(u) = \max\{0, \mu_A(u) - \mu_B(u)\}. \tag{15}$$

$$I_3(A,B) \equiv \parallel \bar{A}\cup B \parallel, \tag{16}$$

$$I_4(A,B) \equiv \inf_{u\in U} \left\{\mu_{\overline{A|-|B}}(u)\right\}, \tag{17}$$

$$I_5(A,B) \equiv \inf_{u\in U} \{\mu_{\bar{A}\cup B}(u)\}. \tag{18}$$

The *consistency* index was

$$C(A,B) \equiv \sup_{u\in U} \{\mu_{A\cap B}(u)\}; \tag{19}$$

For intersection and union we use the minimum and maximum, respectively, and for complement, $\mu_{\bar{A}}(u) = 1-\mu_A(u)$. The 11-th decision template is based on the normalized Euclidean distance between matrices DP and DT_i,

$$N(DP, DT_i) = \mu^i_{\hat{D}}(\mathbf{x}) = 1 - \frac{1}{L\ c} \sum_{k=1}^{L}\sum_{s=1}^{c} (dt_i(k,s) - d_{k,s}(\mathbf{x}))^2. \tag{20}$$

While we use only the Euclidean norm in this study, there is no reason to stop at this choice. Any norm could be used in (20), e.g., the Minkowski norms (1 and sup norms), or the Mahalanobis norm.

It is important to notice the difference between *integral* and *point-wise* measures. Integral measures are based on cardinality $(S_1, S_2, S_3, I_1, I_2, I_3, N)$, while pointwise measures use a single degree of membership to determine their value (S_4, I_4, I_5, C). Therefore, point-wise measures tend to be more sensitive to outliers and prone to errors than integral measures.

4 Fusion techniques used for comparison

4.1 Techniques for crisp individual labels

C1: Majority vote. The class labels of the classifiers are crisp, or are hardened by (1), and the crisp label that is most represented in the set of L labels is assigned to $\mathbf{x}$. Ties are broken randomly. This fusion does not require any parameters to be trained, and is therefore classified in **C1** (Table 3).

C2: "Naive"-Bayes combination (NB) [4]. This scheme assumes that the classifiers are mutually independent (this is the reason we use the name "naive"); Xu et al. [4] and others call it *Bayes* combination. For each classifier D_j, a $c\times c$ confusion matrix CM^j is calculated by applying D_j to the training data set. The (k,s)th entry of this matrix, $cm^j_{k,s}$ is the number of elements of the data set whose true class label was k, and were assigned by D_j to class s. By $cm^j_{\cdot,s}$ we denote the total number of elements labeled by D_j into class s (this is calculated as the sum of the sth column of CM^j). Using these values, a $c\times c$ label matrix

LM^j is computed, whose (k,s)th entry $lm^j_{k,s}$ is an estimate of the probability that the true label is k given that D_j assigns crisp class label s.

$$lm^j_{k,s} = \hat{P}\left(k|D_j(\mathbf{x}) = s\right) = \frac{cm^j_{k,s}}{cm^j_{\cdot,s}},$$

For every $\mathbf{x} \in \Re^n$, D_j yields a crisp label vector $D_j(\mathbf{x})$ pointing at one of the classes, say s in $\in \{1,\ldots,c\}$. Associated with s is a *soft label vector* $[\hat{P}\left(1|D_j(\mathbf{x}) = s\right),\ldots,\hat{P}\left(c|D_j(\mathbf{x}) = s\right)]^T$, which is the sth column of the label matrix LM^j. Let $s_1,\ldots,s_L$ be the crisp class labels assigned to $\mathbf{x}$ by classifiers $D_1,\ldots,D_L$, respectively. Then, by the independence assumption, the estimate of the probability that the true class label is i, (which is the ith component of the final label vector) is calculated by

$$\mu^i_{\hat{D}}(\mathbf{x}) = \prod_{j=1}^{L} \hat{P}\left(i|D_j(\mathbf{x}) = s_j\right) = \prod_{j=1}^{L} lm^j_{i,s_j}, \quad i = 1,\ldots,c.$$

C2: Behavior-Knowledge Space (BKS) [36]. Let again $s_1,\ldots,s_L$ be the crisp class labels assigned to $\mathbf{x}$ by classifiers $D_1,\ldots,D_L$, respectively. Every possible combination of class labels $D_1(\mathbf{x})$, $(s_1,\ldots,s_L) \in \{1,\ldots,c\}^L$ is an index to a cell in a look-up table (BKS table). The table is filled in using the data set Z: $\mathbf{z}_j$ goes to the cell indexed by $D_1(\mathbf{z}_j),\ldots,D_L(\mathbf{z}_j)$. Thus, each entry in the look-up table contains one of the following: a single class label (the one that is most often encountered amongst the elements of Z in this cell); no label (no element of Z had the respective combination of class labels); or a set of tied class labels (if more than one class have the same highest number of elements in this cell).
Example. Let $c = 3,\ L = 2,\ N = 100$. A possible BKS look-up table is displayed in Table 4

Table 4: A possible BKS look-up table

s_1, s_2	1,1	1,2	1,3	2,1	2,2	2,3	3,1	3,2	3,3
Numbers from each class	10/3/3	3/0/6	5/4/5	0/0/0	1/16/6	4/4/4	7/2/4	0/2/5	0/0/6
Cell label	1	3	1,3	0	2	1,2,3	1	3	3

The decision for an $\mathbf{x} \in \Re^n$ is made according to the class label of the cell indexed by $D_1(\mathbf{x}),\ldots,D_L(\mathbf{x})$. Ties are broken randomly. If an empty cell is hit, the class label is chosen at random from $\{1,\ldots,c\}$. The operation of BKS is illustrated in Figure 2.

Both Naive Bayes and BKS have sets of parameters that are estimated using the trained classifiers and the training data: for NB these are the L label matrices, and for BKS, the look-up table. This places them in group **C2** (Table 3).

4.2 Class-conscious fusion techniques for soft labels

CC1: Minimum, Maximum, Average and Product. We used the L-place operators minimum, maximum, average and product as the aggregation rule ($\mathcal{F}$):

$$\mu^i_{\hat{D}}(\mathbf{x}) = \mathcal{F}\ \left(d_{1,i}(\mathbf{x}),\ldots,d_{L,i}(\mathbf{x})\right), \quad i = 1,\ldots,c.$$

Figure 3 shows the operation of simple aggregation rules. The following example helps to clarify these 4 fusion methods. Let $c = 3$ and $L = 5$. Assume that for a certain $\mathbf{x}$,

$$DP(\mathbf{x}) = \begin{bmatrix} 0.1 & 0.5 & 0.4 \\ 0.0 & 0.0 & 1.0 \\ 0.4 & 0.3 & 0.4 \\ 0.2 & 0.7 & 0.1 \\ 0.1 & 0.8 & 0.2 \end{bmatrix}.$$

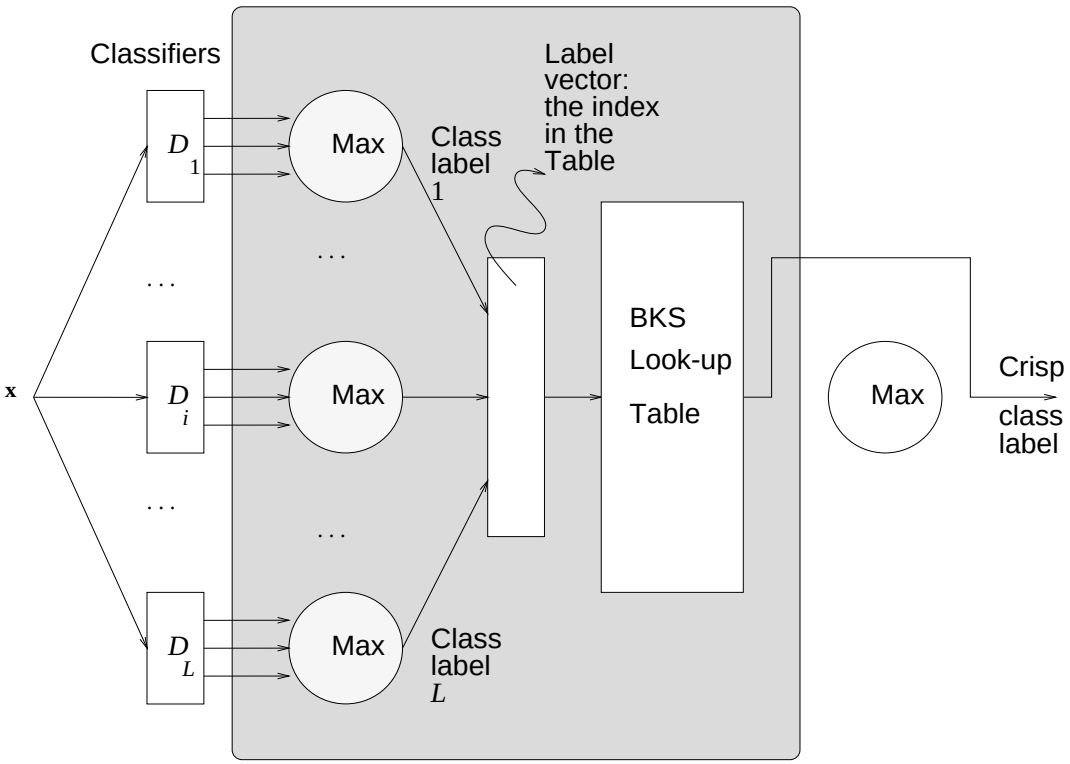


Figure 2: Operation of BKS method for classifier fusion

Classifiers D_1 and D_4 are fuzzy, D_2 is crisp, and D_3 and D_5 are possibilistic. Applying each of the operators columnwise, we obtain as the final soft class label $\mu_{\hat{D}}(\mathbf{x})$

$$\begin{array}{lcl} \text{Minimum} & = & (\ 0.0,\ 0.0,\ 0.1\ )^T; \\ \text{Maximum} & = & (\ 0.4,\ 0.8,\ 1.0\ )^T; \\ \text{Average} & = & (\ 0.16,\ 0.46,\ 0.42\ )^T; \\ \text{Product} & = & (\ 0.0,\ 0.0,\ 0.0032\ )^T; \end{array}$$

If hardened, minimum, maximum, and product will label $\mathbf{x}$ in class 3, whereas the average will put $\mathbf{x}$ in class 2.

CC2: Probabilistic product [39, 40] is an aggregation formula (derived in [47]) which gives the Bayes decision if the classifiers use mutually independent subsets of features and yield the true posterior probability, $d_{i,j}(\mathbf{x}) = P(i|\mathbf{x}_j)$, on their respective feature subspaces,

$$\mu_D^j(\mathbf{x}) = \frac{\prod_{i=1}^{L} d_{i,j}(\mathbf{x})}{P(j)^{L-1}}, \quad j = 1, \ldots, c.$$

For the prior probabilities $P(j)$ we used the sample based estimates from the training set Z

$$\hat{P}(j) = \frac{N_j}{N},\ j = 1, \ldots, c,$$

where N_j is the number of elements in Z from class j and N is the total training sample size. Even when the classifier outputs are not the true values but are estimates of the posterior probabilities, the probabilistic product works well as an aggregation connective.

CC2: Fuzzy integral (FI) [7, 9, 31, 6]. For an input $\mathbf{x}$ we calculate c vectors of length L. Each such vector corresponds to a class, and contains L values of a *fuzzy measure*. Then the ith column of the decision

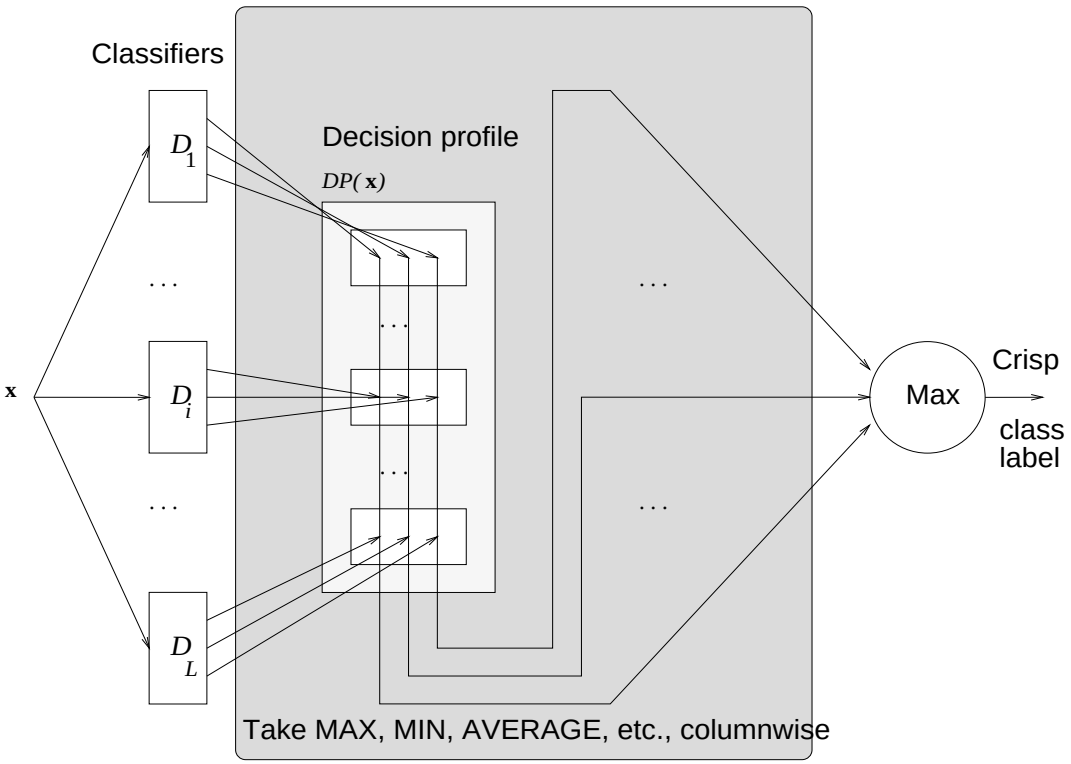


Figure 3: Operation of the simple aggregation rules

profile, with the L values of support for class i, is sorted and *fused* with the fuzzy measure for that class to get $\mu^i_{\hat{D}}(\mathbf{x})$. Thus, fuzzy integration is interpreted as searching for the maximal grade of agreement between the objective evidence (provided by the sorted classifier outputs for class i) and the expectation (the L fuzzy measure values) [6]. The problem is how to find the fuzzy measure vector. Most authors that use the fuzzy integral suggest computing a λ-fuzzy measure g_λ. To find the support for class k, $\mu^k_{\hat{D}}(\mathbf{x})$, the following procedure is applied

1. Fix the L *fuzzy densities* $g^1, \ldots, g^L$. These densities can be interpreted as importance of the classifiers. Wang et al. [31] report that good results have been obtained with $g^i = \hat{P}_i/2$, i.e., half of the estimated probability of correct classification of classifier D_i; Cho and Kim [6] use $g^i = \hat{P}_i, i = 1, \ldots, L$.

2. Calculate $\lambda > -1$ as the only real root greater than -1 of the equation

$$\lambda + 1 = \prod_{i=1}^{L}(1 + \lambda g^i). \tag{21}$$

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. Sort the densities correspondingly, i.e., $g^{i_1}, \ldots, g^{i_L}$

5. Set $g(1) = g^{i_1}$

6. For $t = 2$ to L calculate recursively

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

7. Calculate the final degree of support for class k as

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

Notice that the fuzzy measure vector might be different for each class, and is also specific for the current $\mathbf{x}$. The fuzzy measure vector for two classes will be the same only if the ordering of the classifier support for the classes is the same. For example, consider the DP and classifier accuracies shown in Table 5.

Table 5: Decision profile for $\mathbf{x}$

$DP(\mathbf{x}) =$

class →	1	2	3	4
$D_1(\mathbf{x})$	0.2	0.1	0.4	0.3
$D_2(\mathbf{x})$	0.7	0.0	0.1	0.2
$D_3(\mathbf{x})$	0.1	0.1	0.6	0.2

$$g^1 = \hat{P}_1 = 0.63,$$
$$g^2 = \hat{P}_2 = 0.70,$$
$$g^3 = \hat{P}_3 = 0.66,$$

Solving equation (21) we get $\lambda = -0.94977$. Applying the above procedure separately to each column of the table, we get the results shown in Table 6.

Table 6: Applying the FI procedure to the columns of $DP(\mathbf{x})$ in Table 5

column 1	g
0.7	0.70
0.2	0.91
0.1	1.00

column 2	g
0.1	0.66
0.1	0.90
0.0	1.00

column 3	g
0.6	0.66
0.4	0.90
0.1	1.00

column 4	g
0.3	0.63
0.2	0.90
0.2	1.00

$$\mu^1_{\hat{D}}(\mathbf{x}) = 0.7 \qquad \mu^2_{\hat{D}}(\mathbf{x}) = 0.1 \qquad \mu^3_{\hat{D}}(\mathbf{x}) = 0.6 \qquad \mu^4_{\hat{D}}(\mathbf{x}) = 0.3$$

Thus, the class label for $\mathbf{x}$ is 1. The FI scheme is illustrated in Figure 4.

CI2: Dempster-Shafer combination (DS) [2]. This technique is the one closest to the DT. The classifier outputs $\{D_i(\mathbf{x})\}$ are possibilistic. Instead of calculating the similarity between the decision template DT_i and the decision profile $DP(\mathbf{x})$, the DS algorithm goes further. The following steps are performed:

1. Let DT^i_j denote the ith row of the decision template for class j. We calculate the "proximity" Φ between DT^i_j and $D_i(\mathbf{x})$ for every class $j = 1, \ldots, c$ and for every classifier $i = 1, \ldots, L$. As recommended in [2], this proximity is calculated as

$$\Phi_{j,i}(\mathbf{x}) = \frac{\left(1 + ||DT^i_j - D_i(\mathbf{x})||^2\right)^{-1}}{\sum_{k=1}^{c} \left(1 + ||DT^i_k - D_i(\mathbf{x})||^2\right)^{-1}}, \tag{22}$$

   where $\| * \|$ is any matrix norm.

2. Using (22), we calculate for every class, $j = 1, \ldots, c$; and for every classifier, $i = 1, \ldots, L$, the following *belief* degrees

$$b_j\left(D_i(\mathbf{x})\right) = \frac{\Phi_{j,i}(\mathbf{x}) \prod_{k \neq j} (1 - \Phi_{k,i}(\mathbf{x}))}{1 - \Phi_{j,i}(\mathbf{x}) \left[1 - \prod_{k \neq j} (1 - \Phi_{k,i}(\mathbf{x}))\right]}$$

3. The final DS label vector with membership degrees has the components

$$\mu^j_{\hat{D}}(\mathbf{x}) = K \prod_{i=1}^{L} b_j\left(D_i(\mathbf{x})\right), \quad j = 1, \ldots, c,$$

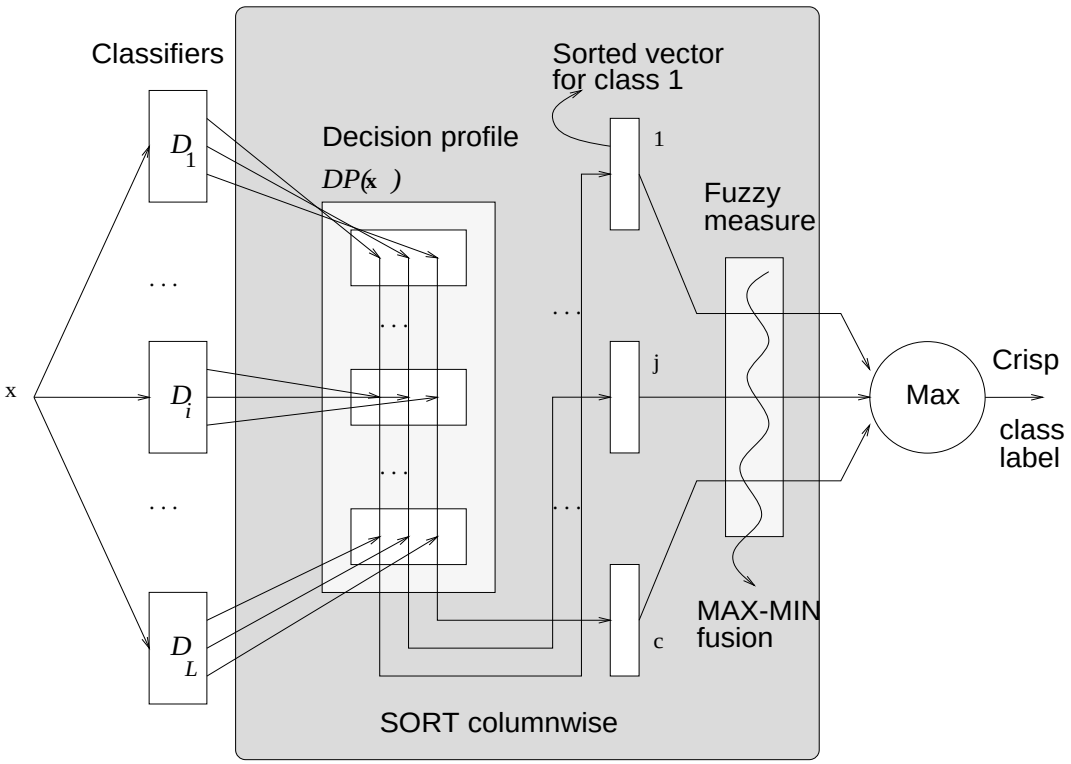


Figure 4: Operation of fuzzy integral for classifier fusion

where K is a normalizing constant.

In the **CI2** category we also use some well known classifiers: linear and quadratic discriminant classifiers (LDC and QDC assuming normal densities [33]), the logistic classifier (LOG) [48], and Fisher's discriminant (FSH) [33]. LDC and Fisher are identical for well-sampled two class problems. For undersampled datasets LDC (and even more severely QDC) suffer from unstable covariance matrix estimates. Our Fisher implementation uses the pseudo-inverse approach [49] in those situations.

5 Experiments

We used two data sets from the ELENA database (anonymous ftp at ftp.dice.ucl.ac.be, directory pub/neural-nets/ELENA/databases). Results with the same data using classifier *selection* methods can be found in [3].

The **Satimage** data was generated from Landsat Multi-Spectral Scanner image data. It consists of 6435 pixels with 36 attributes (4 spectral bands $\times$ 9 pixels in a 3×3 neighborhood). The pixels are crisply classified in 6 classes, and are presented in random order in the database. The classes are: red soil (23.82 %), cotton crop (10.92 %), grey soil (21.10 %), damp grey soil (9.73 %), soil with vegetation stubble (10.99 %), and very damp grey soil (23.43 %). What makes this database attractive is: large sample size; numerical, equally ranged features; no missing values; and compact classes of approximately equal size, shape and prior probabilities. Figure 5 is a scatterplot of the 6 Satimage classes on features # 17 and # 18. In our experiments we used only features # 17 to # 20, as recommended by the database designers.

The **Phoneme** data consists of 5404 5-dimensional vectors characterizing two classes of phonemes: nasals (70.65 %) and orals (29.35 %). The scatterplot of features 3 and 4 of 800 randomly selected data points is shown in Figure 6. The two classes are highly overlapping with complex classification boundaries, suggesting that parametric classifiers will not achieve good results.

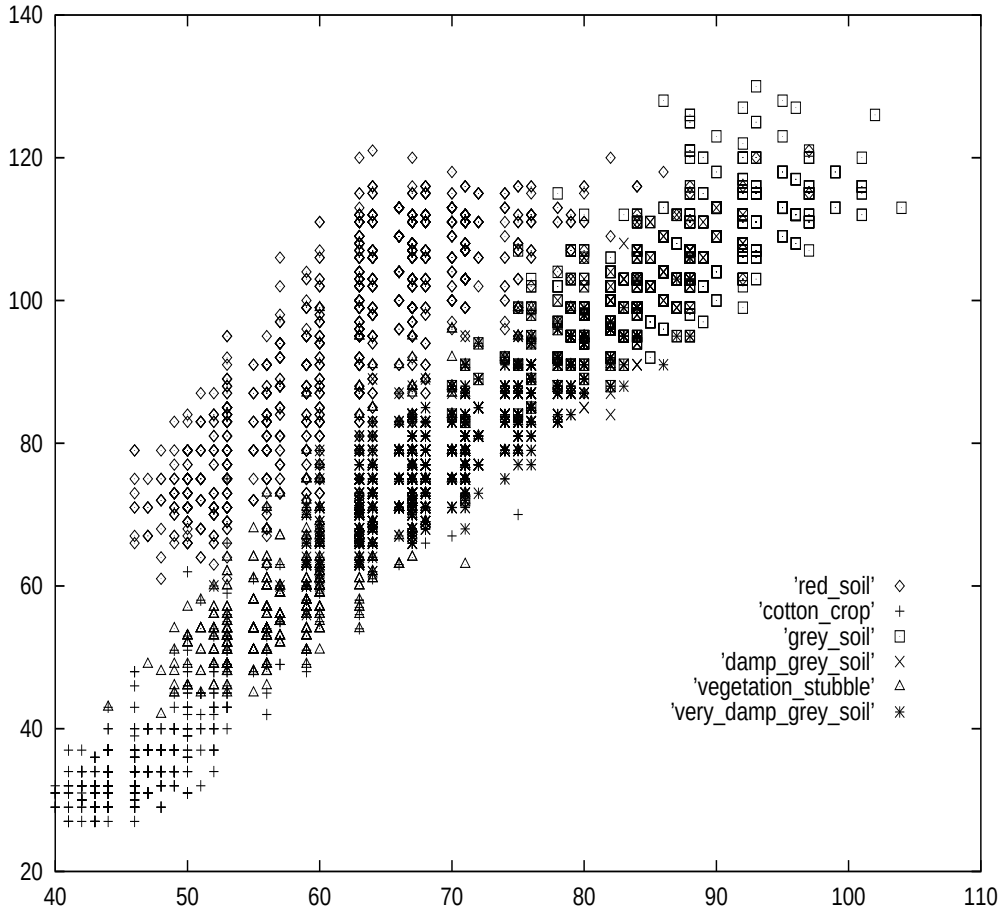


Figure 5: Satimage data on features # 17 and # 18

Using the Matlab code for the Quadratic Discriminant Classifier from the package PRTOOLS [50] we designed 6 classifiers with the Satimage data and 10 classifiers with the Phoneme data using all combinations of 2 features in each case. For example, we trained six QDCs for the 4-D Satimage data, using feature pairs (17,18), (17,19), ... (19,20). We split each 2D data set into training and testing sets, and averaged the results from 10 random data shuffles. Four training set sizes were used with both Satimage and Phoneme: 100, 200, 1000, and 2000. For all experiments, the test set was the remaining part of the data set after taking out the training set.

We did not consider here the reject option – all ties were broken randomly. Therefore in our version of majority voting we assigned a class label, even if the number of votes for that class might be less than half (this concerns the Satimage data where 6 classes are possible and the votes may spread so that none of the classes gets more than half of all votes).

Tables 7 and 8 show the classification accuracy for the two data sets, respectively. We display only the % correct on the test sets, which have not been seen during training of either the individual classifiers or the second level fusion models.The fusion schemes are arranged as follows:

Group C1

1. MAJ. Majority voting.

Group C2

1. NB. Naive Bayes.
2. BKS. Behavior Knowledge Space method.

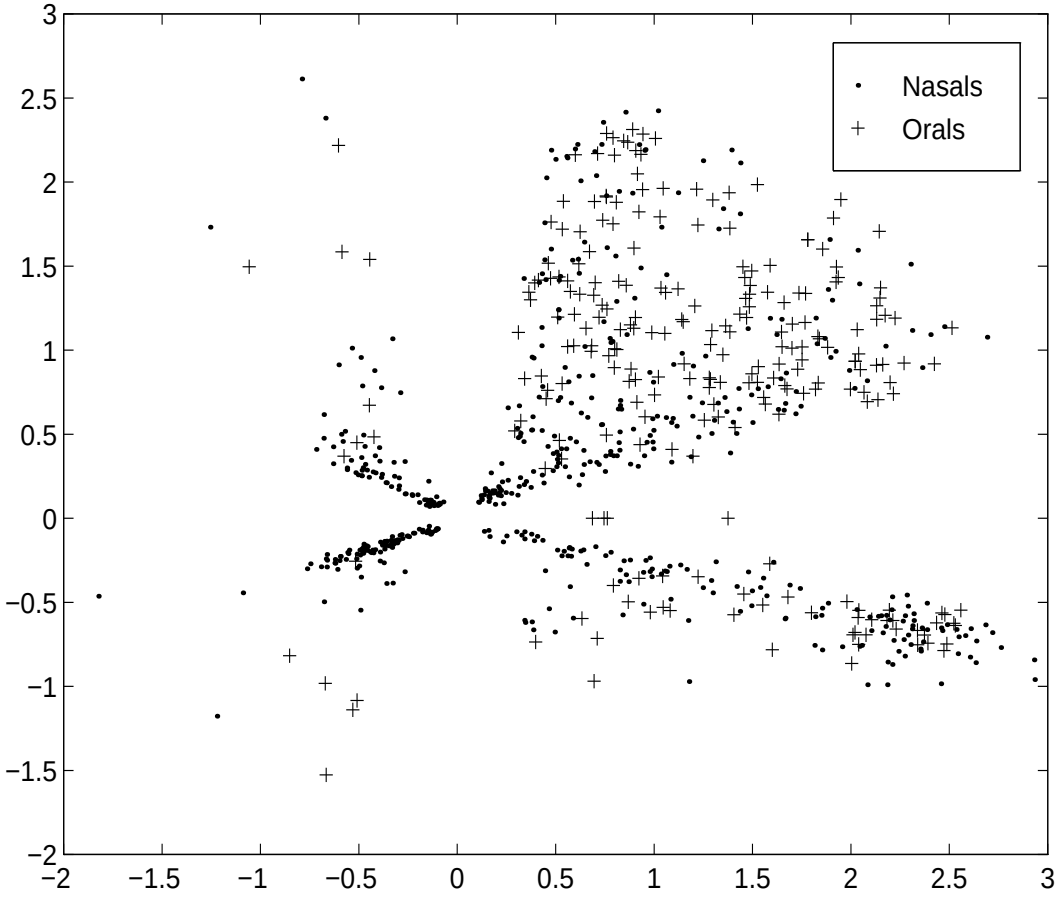


Figure 6: Phoneme data on features # 3 and # 4

Group CC1

1. MAX. Maximum aggregation rule.
2. MIN. Minimum aggregation rule.
3. AVR. Average aggregation rule.
4. PRO. Product aggregation rule.

Group CC2

1. PPR. Probabilistic product.
2. FI. Fuzzy integral.

Group CI2

1. DS. Dempster-Shafer.
2. LDC. Linear discriminant classifier on the intermediate-output space.
3. QDC. Quadratic discriminant classifier.
4. LOG. Logistic classifier.
5. FSH. Fisher linear classifier.

Group CI2 (DTs) Each DT scheme is denoted by "DT:ss", where "ss" stands for the respective similarity measure, e.g., DT:I5. By DT:NM we denote the DT scheme based on the Euclidean distance. "NM" stands for the "nearest mean".

Also given in Tables 7 and 8 are the accuracy of the *single best* (SB) classifier and the *"oracle"* (OR). The "oracle" works as follows: assign the *correct* class label to $\mathbf{x}$ iff at least one individual classifier produces the correct class label of $\mathbf{x}$ (when its decision is hardened).

For classifiers 2,3,4 and 5 in Group CI2, we used the Matlab code in PRTools.

Not surprisingly, the fusion schemes have approximately the same performance. To find out which were consistently better (even a little better) than the others, we sorted the testing accuracies and calculated their ranks. To the right of classification accuracy in each column is the *rank* of the fusion scheme, based on that column. For an individual test, the ranks range from 1 (poorest) to 26 (best). The last column in each table is the total rank (the sum of the four) for the respective data set.

Table 9 shows the 25 schemes (without FSH, which cannot be run for 6 classes, and was applied only to the Phoneme data) sorted by their ranks for Satimage, Phoneme, and both data sets. Since there were two data sets and four tests (averaged) each, the maximum possible score is 26(8)=208, which is achieved by the oracle. If the same classifier had ranked lowest at all 8 tests, the minimum cumulative value of 8(1)=8 would be attained, but this did not happen.

Table 7: Average test accuracy (in %) and ranks for the SATIMAGE data

Training size →	100		200		1000		2000		Total rank
SB	77.50	7	79.73	6	80.67	7	80.62	6	26
C1									
MAJ	80.89	9	81.27	9	82.13	10	82.23	10	38
C2									
NB	77.18	6	80.28	8	83.45	13	83.70	13	40
BKS	68.15	5	72.26	5	79.81	6	81.59	9	25
CC1									
MAX	81.44	11	82.19	13	82.91	12	82.84	12	48
MIN	82.51	15	83.53	23	84.42	23	84.57	25	86
AVR	82.63	17	82.85	14	83.81	14	83.88	15	60
PRO	83.02	21	83.55	24	84.44	24	84.48	22	91
CC2									
PPR	82.53	16	83.42	16	84.14	16	84.18	16	64
FI	81.75	13	81.95	12	82.64	11	82.77	11	47
CI2									
DS	83.01	20	83.43	17	84.42	22	84.50	23	82
LDC	41.97	3	41.64	4	32.33	2	42.45	2	11
QDC	23.83	1	22.55	1	28.69	1	47.24	3	6
LOG	57.23	4	39.58	3	39.89	4	55.14	4	15
DT:NM	82.98	19	83.48	22	84.50	25	84.57	24	90
DT:I1	83.11	23.5	83.45	19.5	84.32	18.5	84.38	19.5	81
DT:I2	83.11	23.5	83.45	19.5	84.32	18.5	84.38	19.5	81
DT:I3	81.62	12	81.60	10	81.54	8	81.26	7	37
DT:I4	82.36	14	83.57	25	84.32	21	84.37	17	77
DT:I5	81.21	10	81.91	11	82.00	9	81.39	8	38
DT:S1	83.11	23.5	83.45	19.5	84.32	18.5	84.38	19.5	81
DT:S2	83.11	23.5	83.45	19.5	84.32	18.5	84.38	19.5	81
DT:S3	82.93	18	83.28	15	83.88	15	83.87	14	62
DT:S4	34.81	2	32.15	2	34.37	3	31.53	1	8
DT:C	79.50	8	79.85	7	79.36	5	79.29	5	25
OR	90.15	26	90.29	26	90.47	26	90.51	26	104

Table 8: Average test accuracy (in %) and ranks for the Phoneme data

Training size →	100		200		1000		2000		Total rank
SB	74.29	11	75.45	13	75.52	9	75.17	9	42
C1									
MAJ	75.51	21	75.96	19	76.38	16	76.08	15	71
C2									
NB	74.13	9	73.58	6	74.09	8	73.61	8	31
BKS	72.71	5	75.12	10	77.48	24	77.96	24	63
CC1									
MAX	75.42	19.5	75.40	11.5	75.80	10.5	75.47	10.5	52
MIN	75.42	19.5	75.40	11.5	75.80	10.5	75.47	10.5	52
AVR	75.82	25	75.81	17	76.37	15	75.91	14	71
PRO	75.77	24	75.81	16	76.34	14	75.88	13	67
CC2									
PPR	73.67	7	73.62	7	71.81	5	72.11	5	24
FI	75.65	23	75.60	14	76.10	12	75.76	12	61
CI2									
DS	75.08	18	75.85	18	76.79	18	77.12	18	72
LDC	37.63	1	50.05	1	37.38	1	54.02	2	5
QDC	54.07	2	51.00	2	52.74	2	47.30	1	7
LOG	74.97	12	77.13	25	77.91	25	77.97	25	87
FSH	75.08	-	76.53	-	77.09	-	76.75	-	-
DT:NM	75.02	17	75.78	15	76.73	17	77.12	19	68
DT:I1	75.00	14.5	76.35	21.5	77.07	20.5	77.45	21.5	78
DT:I2	75.00	14.5	76.35	21.5	77.07	20.5	77.45	21.5	78
DT:I3	73.86	8	73.71	8	73.85	7	73.30	7	30
DT:I4	74.16	10	73.86	9	76.17	13	76.94	17	49
DT:I5	69.06	4	66.99	3	65.21	3	65.30	3	13
DT:S1	75.00	14.5	76.35	21.5	77.07	20.5	77.45	21.5	78
DT:S2	75.00	14.5	76.35	21.5	77.07	20.5	77.45	21.5	78
DT:S3	75.64	22	76.50	24	77.12	23	76.65	16	85
DT:S4	65.03	3	68.31	4	68.45	4	69.04	4	15
DT:C	73.36	6	73.07	5	73.22	6	72.69	6	23
OR	97.44	26	97.25	26	97.74	26	97.62	26	104

Table 9: Fusion schemes sorted by their ranks, for Satimage, Phoneme, and both data sets

Satimage		Phoneme		Total for both		Group
OR	104	OR	104	**OR**	208	-
PRO	91	LOG	87	**DT:S2**	159	**CI2**
DT:NM	90	DT:S3	85	**DT:S1**	159	**CI2**
MIN	86	DT:S1	78	**DT:I2**	159	**CI2**
DS	82	DT:I1	78	**DT:I1**	159	**CI2**
DT:S1	81	DT:I2	78	**DT:NM**	158	**CI2**
DT:I1	81	DT:S2	78	**PRO**	158	**CC1**
DT:S2	81	DS	72	**DS**	154	**CI2**
DT:I2	81	MAJ	71	**DT:S3**	147	**CI2**
DT:I4	77	AVR	71	**MIN**	138	**CC1**
PPR	64	DT:NM	68	**AVR**	131	**CC1**
DT:S3	62	PRO	67	**DT:I4**	126	**CI2**
AVR	60	BKS	63	**MAJ**	109	**C1**
MAX	48	FI	61	**FI**	108	**CC2**
FI	47	MIN	52	**LOG**	102	**CI2**
NB	40	MAX	52	**MAX**	100	**CC1**
DT:I5	38	DT:I4	49	**PPR**	88	**CC2**
MAJ	38	SB	42	**BKS**	88	**C2**
DT:I3	37	NB	31	**NB**	71	**C2**
SB	26	DT:I3	30	**SB**	68	-
BKS	25	PPR	24	**DT:I3**	67	**CI2**
DT:C	25	DT:C	23	**DT:I5**	51	**CI2**
LOG	15	DT:S4	15	**DT:C**	48	**CI2**
LDC	11	DT:I5	13	**DT:S4**	23	**CI2**
DT:S4	8	QDC	7	**LDC**	16	**CI2**
QDC	6	LDC	5	**QDC**	13	**CI2**

6 Discussion

Overall classification accuracy. The accuracy of the combinations in our experimental setting is not very high compared to studies on the same data sets reported elsewhere [3]. We believe this is because we did not confer special attention on designing the individual first-level classifiers. In this study we were interested in comparing the second-level fusion schemes, and hence, the type of first-level classifiers was immaterial. The 2-features quadratic discriminant classifier (QDC) that we adopted for all individual classifier designs, was not a bad choice for the Satimage data because the classes are distributed in roughly compact "clouds" (Figure 5). This explains the better accuracy with this data set than with Phoneme data.

Improvement over the Single-best classifier. The gap from 68 to 208 between the single best classifier and the"oracle" presumably shows the "potential" of the pool of classifiers. Interestingly, although for both data sets there is a big gap, many of the fusion schemes did not improve very much on the single-best classifier rate. This is probably due to dependencies between the classifiers. If we used a large number of features and built the classifiers on disjoint subsets the chance to obtain good improvement over the single best classifier would have been higher. The best improvement for the Satimage data (Table 7) was 5.61 % for a training set of size 100, 3.84 % for 200, 3.83 % for 1000, and 3.95 % for 2000. Improvement for the Satimage data was higher than improvement with the Phoneme data (Table 8): 1.53 % for a training set of size 100, 1.68 % for 200, 2.38 % for 1000, and 2.80 % for 2000, although the Phoneme data had a better "oracle".

Effects of sample size. Surprisingly, many of the fusion schemes fared better with smaller amounts of training data. With the Satimage data, DT schemes using integral measures were the best on the smallest training size (100), which can be explained by the fact that they estimate large enough, but not too large, numbers of parameters (like, e.g., BKS). For the Phoneme data, the 2-class problem with training size 100 appears to be best solved by the simplest aggregation rules (which were not so accurate for the 6 Satimage classes). This might be evidence of overtraining for those schemes that use second-level training.

Overtraining and number of parameters. BKS appeared to be most prone to overtraining because its look-up table needs large data sets to be properly calculated. In almost all the experiments the BKS method gave the best training accuracy but did badly on testing. From Tables 7 and 8 it can be seen that BKS had the highest rate of improvement on the testing accuracy from 100 to 2000 training size. The BKS overtraining problem is especially severe with a large number of classes and classifiers: the number of parameters (cells in the table) is L^c, which for the Satimage data is $6^6 = 46656$, and for the Phoneme data is $10^2 = 100$. Not all combinations will be encountered in practice, but a large number of them might be. To compare, DT, DS, and NB acquire $L * c^2$ parameters (216 for Satimage and 40 for Phoneme), FI, $L + 1$ (7 for Satimage and 3 for phoneme), PPR, L (2 for Satimage and 1 for phoneme), while the simple aggregation techniques need to learn none.

Statistical classifiers. LDC and QDC are not appropriate on the intermediate space because the covariance matrices needed for these designs are close to singular, as explained in the introduction. The other two schemes though, the logistic classifier (LOG) and Fisher's discriminant (FSH), do not share this drawback. These two classifiers both gave excellent results with the 2-class Phoneme data. LOG failed on the Satimage data because some of the classes were almost separable in the intermediate space (i.e., the classifiers had very distinct decision templates). It would be interesting to look at other conventional classifiers, for example decision trees or nearest neighbor and multiple prototype rules for the intermediate-output space.

Assumption-based schemes. The assumption-based classifier fusion schemes, Naive Bayes and the probabilistic product, did not reach the performance of the other schemes. They both failed to improve over the single best rate with the Phoneme data. In the overall ranking Naive Bayes was the poorer of these two.

Dempster-Shafer method [2]. This method rated comparatively high on both data sets. It had a little lower final rank than the best DT techniques, and can be put basically in the same group. The calculations that it involves however, are more complex than any of the DT schemes.

Simple aggregation rules. It is somewhat surprising to see how well the simple aggregation rules, with no second-level training, compete with the more sophisticated ones. This is probably the reason that simple aggregation continues to be popular [5, 40]. One problem with simple aggregation is, that although they have good overall performance, it is not clear which one is good for a particular data set. The Product

and Minimum, for example, gave excellent results with the Satimage data (see Table 9 where PRO rates as the best model), but were not as good for the Phoneme data. Interestingly, in our experiments they outperformed the Average, which is viewed as the favorite in this group [5].

Fuzzy integral. In our experiments the fuzzy integral using a λ-fuzzy measure rates in the middle. Gader et al. [7] report the results from a handwritten word recognition problem, where the fuzzy integral dramatically outperforms various neural networks. The authors attribute this to the efficient way in which the FI fusion model uses the additional information (called here class-consciousness). This shows again that there is no "perfect" classifier or fusion scheme that will surpass the others on all data sets.

Decision Templates. In our experiments, DT classifier fusion schemes based on *integral* measures tended to give good results with both data sets. The overall ranking (Table 9) puts 5 of them on the top, just one rank point above the PRO (!). The 4 schemes based on S_1, S_2, I_1, and I_2 were not distinguishable. It can be formally proven that when the individual classifier outputs sum up to the same (fixed) value (1, for the individual QDCs used here), fusion by these four DTs induces the same order on the set of class labels, and therefore, leads to the same decision.

The basic conclusion in our experiments is that the DT fusion model shows superior performance to the other techniques in these experiments. A practically established postulate in pattern recognition is that there is no "best" classifier that will outperform every other method on all data sets. Therefore, we try to design a scheme (for classification or classifier fusion) which yields *generally* better performance amongst similar schemes. Although in terms of increase in classification accuracy the improvement in our experimental study is minor, decision templates appear to be such a scheme. Classifier fusion using DTs does not rely on questionable assumptions (NB and PPR do), is less likely to overtrain than BKS, and rated high on *both* data sets unlike LOG, FSH, PRO, and MIN. The fusion scheme is simple and intuitive, and does not require heavy calculations.

References

[1] L. Lam and C.Y. Suen. Optimal combination of pattern classifiers. *Pattern Recognition Letters*, 16:945–954, 1995.

[2] G. Rogova. Combining the results of several neural network classifiers. *Neural Networks*, 7:777–781, 1994.

[3] K. Woods, W.P. Kegelmeyer, and K. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:405–410, 1997.

[4] L. Xu, A. Krzyzak, and C.Y. Suen. Methods of combining multiple classifiers and their application to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22:418–435, 1992.

[5] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[6] 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.

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

[8] M. Grabisch and F. Dispot. A comparison of some for fuzzy classification on real data. In *2nd International Conference on Fuzzy Logic and Neural Networks*, pages 659–662, Iizuka, Japan, 1992.

[9] J.M. Keller, P. Gader, H. Tahani, J.-H. Chiang, and M. Mohamed. Advances in fuzzy integration for pattern recognition. *Fuzzy Sets and Systems*, 65:273–283, 1994.

[10] I. Bloch. Information combination operators for data fusion: a comparative review with classification. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 26:52–67, 1996.

[11] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 1991.

[12] R.A. Jacobs. Methods for combining experts' probability assessments. *Neural Computation*, 7:867–888, 1995.

[13] M.I. Jordan and L. Xu. Convergence results for the EM approach to mixtures of experts architectures. *Neural Networks*, 8:1409–1431, 1995.

[14] S.J. Nowlan and G.E. Hinton. Evaluation of adaptive mixtures of competing experts. In R.P. Lippmann, J.E. Moody, and D.S. Touretzky, editors, *Advances in Neural Information Processing Systems 3*, pages 774–780, 1991.

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

[16] H. Drucker, C. Cortes, L.D. Jackel, Y. LeCun, and V. Vapnik. Boosting and other ensemble methods. *Neural Computation*, 6:1289–1301, 1994.

[17] J.A. Benediktsson and P.H. Swain. Consensus theoretic classification methods. *IEEE Transactions on Systems, Man, and Cybernetics*, 22:688–704, 1992.

[18] K.-C. Ng and B. Abramson. Consensus diagnosis: A simulation study. *IEEE Transactions on Systems, Man, and Cybernetics*, 22:916–928, 1992.

[19] 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.

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

[21] B.V. Dasarathy and B.V. Sheela. A composite classifier system design: concepts and methodology. *Proceedings of IEEE*, 67:708–713, 1978.

[22] E. Filippi, M. Costa, and E. Pasero. Multy-layer perceptron ensembles for increased performance and fault-tolerance in pattern recognition tasks. In *IEEE International Conference on Neural Networks*, pages 2901–2906, Orlando, Florida, 1994.

[23] C.-C. Chiang and H.-C. Fu. A divide-and-conquer methodology for modular supervised neural network design. In *IEEE International Conference on Neural Networks*, pages 119–124, Orlando, Florida, 1994.

[24] F. Smieja. The pandemonium system of reflective agents. *IEEE Transactions on Neural Networks*, 7:97–106, 1996.

[25] L.I. Kuncheva. Change-glasses approach in pattern recognition. *Pattern Recognition Letters*, 14:619–623, 1993.

[26] L.A. Rastrigin and R.H. Erenstein. *Method of Collective Recognition*. Energoizdat, Moscow, 1981. (In Russian).

[27] E. Alpaydin and M. I. Jordan. Local linear perceptrons for classification. *IEEE Transactions on Neural Networks*, 7(3):788–792, 1996.

[28] R.P.W. Duin and D.M.J. Tax. Classifier-conditional posterior probabilities. In A. Amin, D. Dori, P. Pudil, and H. Freeman, editors, *Advances in Pattern Recognition*, volume 1451 of *Lecture Notes in Computer Science*, pages 611–619. Springer, Berlin, 1998.

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

[30] M. Grabisch and J.-M. Nicolas. Classification by fuzzy integral. *Fuzzy Sets and Systems*, 65:255–271, 1994.

[31] 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.

[32] Y. Lu. Knowledge integration in a multiple classifier system. *Applied Intelligence*, 6:75–86, 1996.

[33] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, NY, 1973.

[34] J.C. Bezdek, J.M Keller, R. Krishnapuram, and N.R. Pal. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Kluwer Academic Publishers, 1999.

[35] 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.

[36] Y.S. Huang and C.Y. Suen. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:90–93, 1995.

[37] L.I. Kuncheva. An application of OWA operators to the aggregation of multiple classification decisions. In R.R. Yager and J. Kacprzyk, editors, *The Ordered Weighted Averaging operators. Theory and Applications*, pages 330–343. Kluwer Academic Publishers, USA, 1997.

[38] M. van Breukelen, R.P.W Duin, D.M.J. Tax, and J.E. den Hartog. Combining classifiers for the recognition of handwritten digits. In *I-st IAPR TC1 Workshop on Statistical Techniques in Pattern Recognition*, pages 13–18, Prague, Czech Republic, 1997.

[39] L.I. Kuncheva. On combining multiple classifiers. In *Proc. 7th International Conference on Information Processing and Management of Uncertainty (IPMU'98)*, pages 1890–1891, Paris, France, 1998.

[40] D.M.J. Tax, R.P.W. Duin, and M. ban Breukelen. Comparison between product and mean classifier combination rules. In *Proc. Workshop on Statistical Pattern Recognition*, Prague, Czech, 1997.

[41] S. Hashem, B. Schmeiser, and Y. Yih. Optimal linear combinations of neural networks: an overview. In *IEEE International Conference on Neural Networks*, pages 1507–1512, Orlando, Florida, 1994.

[42] S. Hashem. Optimal linear combinations of neural networks. *Neural Networks*, 10(4):599–614, 1997.

[43] V. Tresp and M. Taniguchi. Combining estimators using non-constant weighting functions. In G. Tesauro, D.S. Touretzky, and T.K. Leen, editors, *Advances in Neural Information Processing Systems 7*, Cambridge, MA, 1995. MIT Press.

[44] Y.S. Huang and C.Y. Suen. A method of combining multiple classifiers - a neural network approach. In *12th International Conference on Pattern Recognition*, pages 473–475, Jerusalem, Israel, 1994.

[45] L.I. Kuncheva, R.K. Kounchev, and R.Z. Zlatev. Aggregation of multiple classification decisions by fuzzy templates. In *Third European Congress on Intelligent Technologies and Soft Computing EUFIT'95*, pages 1470–1474, Aachen, Germany, August 1995.

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

[47] R.F. Bordley. A multiplicative formula for aggregating probability assessments. *Management Science*, 28:1137–1148, 1982.

[48] J.A. Anderson. Logistic discrimination. In P.R. Krishnaiah and L.N. Kanal, editors, *Classification, Pattern Recognition and Reduction of Dimensionality*, volume 2 of *Handbook of Statistics*, pages 169–191. North Holland, Amsterdam, 1982.

[49] S. Raudys and R.P.W. Duin. On expected classification error of the fisher linear classifier with pseudo-inverse covariance matrix. *Pattern Recognition Letters*, 19(5-6):385–392, 1998.

[50] R.P.W. Duin. PRTOOLS(Version 2). A Matlab toolbox for pattern recognition. Pattern Recognition Group, Delft University of Technology, June 1997.

Summary

In this paper we described *decision templates* (DTs) for combining multiple classifier outputs using 11 *similarity* measures. DTs are based on the similarity between the matrix of classifier outputs for an input $\mathbf{x}$ (the Decision Profile $DP(\mathbf{x})$) and the c matrix templates found as the class means of the classifier outputs. For comparison we formed 3 groups of classifier fusion schemes: **C**, fusion methods which require crisp class labels from the individual classifiers, **CC** "class-conscious" fusion methods, and **CI**, "class-indifferent" methods. Depending on whether or not the fusion scheme needs to train and store parameters for its operation we further divided the two groups into schemes without training (C1, and CC1), and with training (C2, CC2, CI2). The following methods were considered:

- **C1**: Majority voting.
- **C2**: Naive Bayes, Behavior Knowledge Space method.
- **CC1**: Maximum, minimum, average, product aggregation rules.
- **CC2**: Probabilistic product, fuzzy integral.
- **CI2**: A Dempster-Shafer fusion version, linear discriminant classifier on the intermediate-output space, quadratic discriminant classifier, logistic classifier, and Fisher's linear classifier.

We also calculated the accuracy of the single best among the individual classifiers, and the "oracle". We carried out experiments on 10 permutations of two data sets Satimage (6 classes, 6435 4-D vectors) and Phoneme (2 classes, 5404 5-D vectors) from the ELENA database. Four training set sizes were used: 100, 200, 1000, and 2000. The basic conclusion from our experiments is that DTs based on integral measures of similarity (in the broad sense) are superior to the other techniques. The other techniques rely on faulty assumptions (NB, PPR), or have too many parameters and so are overtrained (BKS). Some of the fusion schemes were excellent for the one data set but not as good (or even applicable) for the other (LOG, FSH, PRO, MIN).

An option for improving the overall accuracy may be to *select* a subset from the pool of classifiers instead of using all of them. This will put all the combination rules in the re-training group. The method of selection can vary from exhaustive search (in the case of few classifiers), to procedures borrowed from feature selection or from editing methods for the k-nearest neighbor rule.

Ludmila I. Kuncheva received the MSc degree from the Technical University, Sofia, 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. Dr. Kuncheva is currently a lecturer at the School of Mathematics, University of Wales, Bangor, UK. Her interests include pattern recognition, neural networks, fuzzy classifiers, prototype classifiers and multiple classifier systems.

James C. Bezdek received the BSCE degree from the University of Nevada, Reno, in 1969 and the Ph.D. Degree in Applied Math from Cornell University, Ithaca, NY, in 1973. He is currently a professor in the Computer Science Department at the University of West Florida, Pensacola. His interests include optimization, pattern recognition, computer vision and image processing, computational neural networks and medical applications. Dr. Bezdek is the founding Editor of the *International Journal of Approximate Reasoning* and the *IEEE Transactions on Fuzzy Systems*.

Robert P.W. Duin studied applied physics at Delft University of Technology in the Netherlands. In 1978 he received the Ph.D. degree for a thesis on the accuracy of statistical pattern recognizers. In his research he included various aspects of the automatic interpretation of measurements, learning systems and classifiers. Between 1980 and 1990 he studied and developed hardware architectures and software configurations for interactive image analysis.

At present he is an associate professor of the Faculty of Applied Sciences of Delft University of Technology. His present research interest is in the design and evaluation of learning algorithms for pattern recognition applications. This includes in particular neural network classifiers, support vector classifiers and classifier combining strategies. Recently he started to study the possibilities of relational methods for pattern recognition.