Classification by pairwise coupling of imprecise probabilities

Benjamin Quost, Sébastien Destercke

Classification by Pairwise Coupling of Imprecise Probabilities

Benjamin Quost[1] and Sébastien Destercke

UMR UTC-CNRS 7253 Heudiasyc
Université de Technologie de Compiègne
BP 20529 - F-60205 Compiègne cedex - France

Abstract

In this paper, we are interested in making decisions by combining classifiers providing uncertain outputs, in the form of sets of probability distributions. More precisely, each classifier provides lower and upper bounds on the conditional probabilities of the associated classes. The classifiers are combined by computing the set of unconditional probability distributions compatible with these bounds, by solving linear optimization problems. When the classifier outputs are inconsistent, we propose a correcting step that restores this consistency. The experiments show the interest of our approach for solving multi-class classification problems, particularly when information is scarce (i.e., a limited number of classifiers is available). In this case, modeling the lack of information associated with classifier outputs gives good results even when they are poorly regularized or overfit the data.

1. Introduction

Supervised classification aims at training classifiers to identify the class of future instances. Classically, a training set of n p-dimensional feature vectors $x_i \in \mathcal{X}$, $i = 1, \ldots, n$ associated with class labels $y_i \in \Omega = \{\omega_1, \ldots, \omega_K\}$ is available. Based on these data, a classifier can be trained to map the input space $\mathcal{X}$ to the label space Ω. Multi-class classification problems with a high number of classes and non-linear class boundaries usually require complex classifiers to be solved, which in turn calls for a larger amount of training data and computational power.

In a number of applications, it is not possible to use such complex models, due to limitations in terms of data, computational resources, or for the sake of interpretability. To overcome this issue, ensemble classification techniques may be used, either by training a set of simpler models who can jointly reproduce complex behaviors (such as boosting [1] or random forests [2]), or by decomposing the learning problem into simpler sub-problems, solve these sub-problems separately via specific classifiers, and then combine the results [3, 4, 5].

[1]Corresponding author. E-mail: benjamin.quost@hds.utc.fr. Tel.: +33.3.44.23.49.68. Fax: +33.3.44.23.44.77

Among those latter decomposition-and-combination strategies, the use of binary classifiers [6, 7] has received particular attention. In this case, each sub-problem consists in separating two (sets of) classes from each other. For example, the one-against-all decomposition scheme [8] consists in opposing each class to all the others; the pairwise strategy [4] (such as pictured in Figure 1 for a 4-class problem), in opposing each class to each other. Both approaches may be generalized within the theoretical framework of error-correcting output codes [3]. One can then use any kind of classifier to solve each of the binary problems (e.g., support vector machines, decision trees, naive Bayes, logistic regression [9, 10], etc). Note, however, that binary classifier combination is known to be beneficial with respect to direct multiclass approaches when considering a class of simple classifiers (e.g., combining linear classifiers makes it possible to compute a non-linear decision boundary). On the other hand, combining sophisticated classification algorithms (such as, e.g., kernel SVM, neural networks, or deep learning [11]) will not significantly increase classification accuracy compared to direct multi-class classification. This phenomenon was pointed out in [12], and further studied in [8] where the pairwise, one-against-all and direct multi-class schemes were shown to perform similarly when well-regularized classifiers are combined.

The key issue of how the classifiers should be combined is dependent on the nature of the classifier outputs [13] and on the decomposition strategy used. Popular strategies include voting [14, 15], averaging [16, 17], solving an optimization problem [18, 19] in case of probabilistic outputs, or adding an additional step for mapping the classifier outputs into a decision [20, 21, 22]. In this article, we focus on such binary classifier combination from a probabilistic point of view. Each sub-problem consists in training a probabilistic binary classifier $\mathcal{C}_i$ to separate two sets $A_i, B_i \subseteq \Omega$ from each other. Given a new instance x, each sub-classifier $\mathcal{C}_i$ then provides a conditional probability estimate

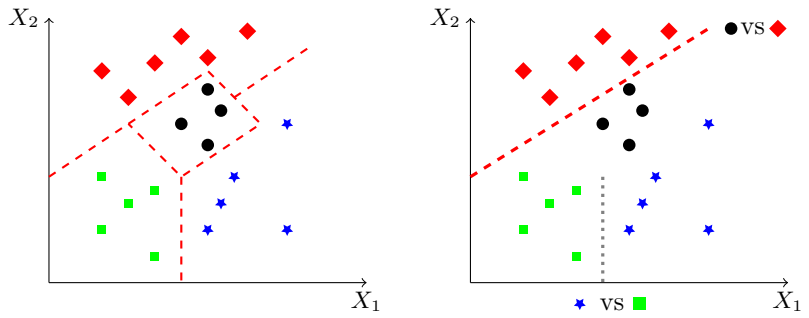
$$\widehat{P}(A_i|\{A_i, B_i\}, x) = 1 - \widehat{P}(B_i|\{A_i, B_i\}, x). \tag{1}$$

Rather than combining the classifiers by using a voting rule, a classical strategy consists in finding a joint posterior probability distribution $\hat{p}(\cdot|x) : \Omega \to [0, 1]$ as close as possible to the conditional binary probabilities given by Equation (1). Thus, the result of the combination is a probability distribution which provides a richer information with respect to the actual class of the instance to classify.

However, combining pairwise probabilistic classifiers usually faces two major difficulties:

- first, when classifying a new instance, some combined classifiers may provide irrelevant, noisy information. For instance, In Figure 1, a classifier trained to separate class "square" (■) from class "star" (★) will not provide any useful information concerning class "diamond" (◆). In the pairwise decomposition scheme, irrelevant classifiers outnumber the relevant ones as soon as $K \geq 5$. For this reason, some works have considered adding a correction or selection step in order to reduce the weight of irrelevant classifiers in the combination strategy [17, 19, 23].

- Also, since the classifiers are trained using partial information, their outputs are (almost) always inconsistent (i.e., there does not exist an unconditional probability distribution which can be conditioned so as to retrieve



Initial multi-class problem Binary separation

Figure 1: Multi-class problem and pairwise binary decomposition

the classifier outputs). Then, the combination usually requires to determine an unconditional probability distribution which minimizes some (subjective, arbitrarily chosen) distance to the conditional estimates given by Equation (1). The complexity of the resulting optimisation procedure generally depends on the chosen decomposition scheme and distance.

In this work, we consider imprecise classifiers, which provide lower and upper bounds on the conditional probabilities rather than a single estimation:

$$[\underline{P}(A_i|\{A_i, B_i\}, x), \overline{P}(A_i|\{A_i, B_i\}, x)]. \tag{2}$$

The imprecise probabilistic combination scheme investigated in this paper is summarized in Figure 2: the individual interval estimates $[\underline{P}(A_i), \overline{P}(A_i)]$ are combined into a joint set $\mathcal{P}$ of unconditional probabilities compatible with these intervals, from which predictions are then made.

Such a strategy offers an original and interesting solution to both the aforementioned issues. Indeed, imprecise probability theory offers decision strategies allowing for predicting multiple classes when the information is insufficient to safely make a unique, optimal prediction. Our approach, although versatile, is more specifically dedicated to classification problems with scarce information (in which case classifier combination may be preferred to direct multiclass classification). In such a case, a cautious behavior may be adopted, in particular regarding instances for which information is scarce or uncertain. Rather than choosing a single class, an imprecise decision can then be made, opening the way to subsequently involving a human expert in the process. Such alternative imprecise decision strategies will also be investigated in this paper.

Besides, form a technical point of view, an unconditional probability distribution whose conditionings are consistent with such interval-valued classifier outputs is more likely to exist than in presence of point estimates, as shown by Example 4 later on. As a matter of fact, the width of the intervals given by Equation (2) can be used to model the inaccuracy of a classifier, and thus the degree to which its outputs are to be discredited in the combination process, larger intervals having less influence on the final output. Should the classifier outputs still be inconsistent with each other, such an interpretation also opens the way to new correcting strategies, as we shall discuss in Section 3. In a nutshell, resolving conflict then consists in stretching some of the classifier outputs until consistency is met.

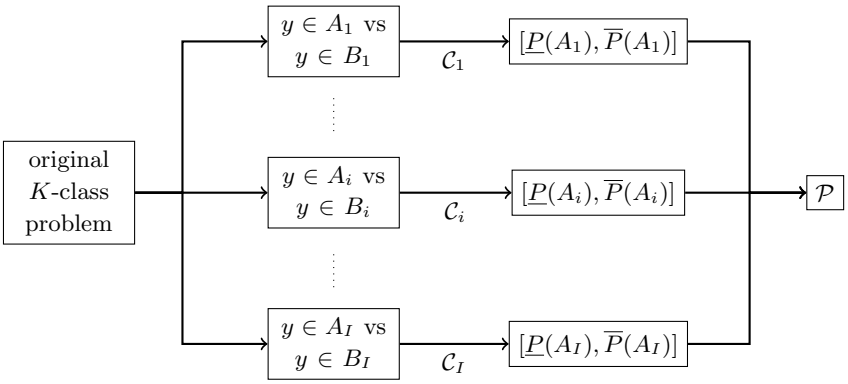


Figure 2: Proposed decomposition scheme

Some previous works based on various uncertainty theories already explored combining imprecise classifier outputs, using in particular belief functions and the associated theory of evidence. Many of these approaches [24, 25, 26, 16] consist in expressing the classifier outputs on the same domain and using an aggregation operator. In [19] however, the combination was carried out by retrieving an unconditional belief function whose conditionings were as close as possible to the classifier outputs according to the Euclidean distance. This was carried out by solving an optimization problem, similarly to the approach proposed by Hastie and Tibshirani [18]. Compared to such works, our proposal differs in two main different points: it is fully coherent with a robust probabilistic approach, as our final estimate is a set of probabilities; and it can produce cautious predictions in the form of sets of potential classes when the final information given by $\mathcal{P}$ is too imprecise. To our knowledge, this is the first proposal to address the binary decomposition problem with imprecise probabilities. In summary, we expect it to have the following advantages with respect to previous methodologies:

- taking into account the scarcity of the training data and the relevance of a classifier to differentiate two sets of classes A_i and B_i given x (both of these issues being closely related to each other);
- in presence of inconsistent classifier outputs, providing a cautious estimate by relaxing the constraints corresponding to these outputs, rather than violating them.

This paper is an extended version of [27, 28], with added examples, discussions and experiments (in particular, many different decomposition schemes are investigated). The necessary background concerning imprecise probabilities is summarized in Section 2, where we also mention some of the existing strategies to make imprecise decisions (i.e., corresponding to a set of possible classes). In Section 3, we present our approach for combining classifiers providing imprecise probabilistic outputs. In particular, we detail our approach to deal with inconsistent outputs, based on a specific correction of classifier bounds resulting in a set of potential unconditional probabilities. The section ends with a summary of the proposed approach and a discussion regarding its computational complexity. Section 4 is dedicated to comparing our approach to several state-of-the-art

combination techniques considering precise information. For this purpose, we consider the four main decomposition schemes in the experimentations (one-vs-all, one-vs-one, and error-correcting output coding with dense and sparse code matrices), while our previous papers only considered one-vs-one decomposition. Eventually, Section 5 presents our conclusions and lists some perspectives of future work.

2. Imprecise probabilities

Many authors have argued that probability theory alone is not able to represent faithfully all kinds of uncertainty. In particular, the validity of the classical probabilistic framework can be questioned when only a small quantity of data is available, when the information at hand is imprecise, or for the purpose of modelling source reliability. Imprecise probabilistic approaches [29, 30], which are close in spirit to robust Bayesian ones [31], were introduced as a generalization of probabilities for solving the aforementioned issues. In practice, such approaches consist in considering a convex subset $\mathcal{P}$ of probability distributions as a model of uncertainty, rather than a single one p.

2.1. Basic definitions

Let $\Omega = \{\omega_1, \ldots, \omega_K\}$ be a finite set of outcomes (e.g., the possible classes for a test instance to be classified). A convex probability set $\mathcal{P}$ can be defined by specifying a set of constraints on the probabilities, in the form of expected lower and upper bounds. Such constraints make it possible to express partial knowledge of the outcomes (e.g., in the class example, to consider the probability distributions such that one class is more probable than another).

Let $\mathcal{L}(\Omega)$ denote the set of all real-valued bounded functions over Ω, and $f : \Omega \to \mathbb{R}$ an element of $\mathcal{L}(\Omega)$. A lower expectation bound $\underline{E}(f)$ on f defines a linear constraint on possible probability distributions of the form

$$\underline{E}(f) \leq \sum_{\omega \in \Omega} p(\omega) \cdot f(\omega) = E(f). \tag{3}$$

Similarly, an upper expectation bound $\overline{E}(f)$ of $E(f)$ writes

$$\overline{E}(f) \geq \sum_{\omega \in \Omega} p(\omega) \cdot f(\omega).$$

Note that we can work only with lower expectation bounds, since an upper bound $\overline{E}(f)$ on f can always be turned into a lower one:

$$-\overline{E}(f) \leq \sum_{\omega \in \Omega} p(\omega) \cdot \text{-}f(\omega),$$

meaning by duality that $-\overline{E}(f) = \underline{E}(-f)$.

Lower expectation bounds $\underline{E} : \mathcal{K} \to \mathbb{R}$ defined on a finite set of functions $\mathcal{K} \subseteq \mathcal{L}(\Omega)$ then induce a convex subset of probabilities

$$\mathcal{P}(\underline{E}) = \{p \in \mathbb{P}_\Omega | \underline{E}(f) \leq E(f) \text{ for all } f \in \mathcal{K}\}, \tag{4}$$

where $\mathbb{P}_\Omega$ is the set of all probability distributions over Ω. The set $\mathcal{P}(\underline{E})$ may be empty when the constraints are inconsistent, in which case $\mathcal{P}(\underline{E}) = \emptyset$.

Conversely, lower and upper expectations of a new function $g \in \mathcal{L}(\Omega)$ can be computed from a given (non-empty) set $\mathcal{P}$. Such bounds then correspond to

$$\overline{E}(g) = \sup_{p \in \mathcal{P}} E(g) \quad \text{and} \quad \underline{E}(g) = \inf_{p \in \mathcal{P}} E(g). \tag{5}$$

Note that since the expectation $E(g)$ is a linear function of the probabilities $p(\omega)$, solving Equation (5) comes down to solve a linear program when $\mathcal{P}(\underline{E})$ is defined by linear constraints (as in Equation (4) and throughout this paper), hence can be solved in polynomial time.

Lower and upper probabilities of an event $A \subseteq \Omega$ correspond to expectation bounds over the indicator function $\mathbb{1}_A$ (with $\mathbb{1}_A(\omega) = 1$ if $\omega \in A$, and 0 otherwise). When no confusion is possible, we will denote them $\underline{P}(A)$ and $\overline{P}(A)$. Formally, they are defined as

$$\underline{P}(A) = \inf_{P \in \mathcal{P}} P(A) \quad \text{and} \quad \overline{P}(A) = \sup_{P \in \mathcal{P}} P(A). \tag{6}$$

Example 1. *Consider a space $\Omega = \{\omega_1, \omega_2, \omega_3\}$ composed of three elements corresponding to the possible classes of instances. Confronted with a new instance, the following assessments are provided about the class by an expert or some (imprecise) classifier:*

- *the class ω_3 is at least twice more probable than ω_2: $2p(\omega_2) \leq p(\omega_3)$;*
- *the probability of class ω_1 is not higher than 0.4: $p(\omega_1) \leq 0.4$.*

The first statement can be transformed into $0 \leq p(\omega_3) - 2p(\omega_2)$: this amounts to provide the lower expectation $\underline{E}(f_1) = 0$, where the function f_1 is defined by $f_1(\omega_1) = 0, f_1(\omega_2) = -2, f_1(\omega_3) = 1$. The second statement says that $\overline{E}(\mathbb{1}_{\{\omega_1\}}) = 0.4 \Leftrightarrow \underline{E}(-\mathbb{1}_{\{\omega_1\}}) = -0.4$, which is equivalent to assess that $\underline{E}(f_2) = 0.6$, with $f_2 = \mathbb{1}_{\{\omega_2, \omega_3\}}$ (here, $\mathbb{1}_A$ stands for the indicator function of the set of classes $A \subseteq \Omega$).

The set $\mathcal{P}(\underline{E})$ induced by the constraints $\underline{E}$ defined on $\mathcal{K} = \{f_1, f_2\}$ is displayed in Figure 3. In the figure, each point of the triangle represents a probability distribution in barycentric coordinates, and the probability mass of an element is proportional to the distance between the corresponding point and the edge opposite to this element. See [32] for more details about the simplex representation. In the figure, probability distributions are given as vectors.

Given $\mathcal{P}(\underline{E})$, computing the lower expectation $\underline{E}(g)$ of a new function g, for instance $g(\omega_1) = -1, g(\omega_2) = 2, g(\omega_3) = 3$ comes down to solve

$$\min_{p \in \mathcal{P}(\underline{E})} -1p(\omega_1) + 2p(\omega_2) + 3p(\omega_3).$$

The minimum is obtained for the distribution $p(\omega_1) = 0.4, p(\omega_2) = 0.2, p(\omega_3) = 0.4$ (which correspond to an extreme point of $\mathcal{P}(\underline{E})$), and $\underline{E}(g) = 0.4 \cdot -1 + 0.2 \cdot 2 + 0.4 \cdot 3 = 1.2$. Note that g was here defined arbitrarily for the sake of the example; in general, it corresponds to utility values defining a decision criterion (for instance, in classification, the decision regarding the class of an instance, or a comparison between two possible classes).

In the sequel, we will repeatedly use a specific type of probability set $\mathcal{P}_v$, usually called *vacuous set*, which corresponds to the whole probability simplex, and thus models complete ignorance about the distribution of interest :

$$\mathcal{P}_v = \{p \in \mathbb{P}_\Omega\}. \tag{7}$$

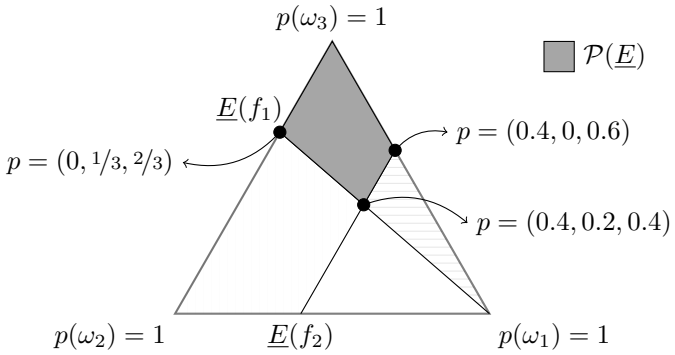


Figure 3: Set $\mathcal{P}(\underline{E})$ of Example 1

Note that given a function g, the lower and upper expectations of $\mathcal{P}_v$ are given by

$$\underline{E}_v(g) = \min_{\omega \in \Omega} g(\omega) \quad \text{and} \quad \overline{E}_v(g) = \max_{\omega \in \Omega} g(\omega). \tag{8}$$

2.2. Resolving inconsistency by discounting

As mentioned in Section 2.1, a set of lower bounds $\underline{E} : \mathcal{K} \to \mathbb{R}$ may induce an empty set $\mathcal{P}(\underline{E}) = \emptyset$, in which case the constraints defined by $\underline{E}$ are inconsistent. One possible way to solve this issue consists in weakening some of the constraints by decreasing the corresponding lower bounds, so that the resulting credal set is non-empty. For a given function f, this can be done by considering a so-called discounting operation, which consists in combining linearly the piece of knowledge expressed by $\underline{E}(f)$ with the vacuous set representing total ignorance. In practice, $\underline{E}(f)$ is transformed into

$$\underline{E}^{\epsilon}(f) = (1 - \epsilon)\underline{E}(f) + \epsilon \underline{E}_v(f), \tag{9}$$

with $\epsilon \in [0, 1]$, and where $\underline{E}_v(f)$ is given by Equation (8). In Equation (9), $1 - \epsilon$ can be interpreted as the degree of reliability of the initial piece of information; this latter is retrieved in case of total reliability ($\epsilon = 0$), while total ignorance is obtained in the opposite case ($\epsilon = 1$).

Example 2. *Let us continue Example 1, assuming the same information on f_1 and f_2 but having an additional expert (or classifier) assessing that the probability of class ω_3 is not higher than 0.2: $p(\omega_3) \leq 0.2$. Again, this is equivalent to provide the lower bound $\underline{E}(f_3) = 0.8$ for $f_3 = \mathbb{1}_{\{\omega_1, \omega_2\}}$.*

However, if we consider $\mathcal{K} = \{f_1, f_2, f_3\}$ with the provided corresponding lower bounds, then $\mathcal{P}(\underline{E}) = \emptyset$, as illustrated in Figure 4: the intersection of the three regions thus defined is empty. Discounting f_3 with a factor $\epsilon = 0.75$ makes it possible to solve this issue: in this case, the constraint becomes

$$\underline{E}^{0.75}(f_3) = 0.25\,\underline{E}(f_3) + 0.75\,\underline{E}_v(f_3) = 0.25 \times 0.8 + 0.75 \times 0 = 0.2.$$

As a result, the discounted constraint is no longer conflicting with the two others, and the credal set is now non-empty.

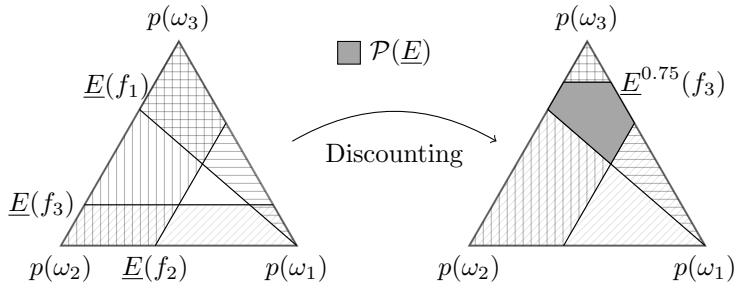


Figure 4: Initial and discounted constraints of Example 2

2.3. Decision rules

When the knowledge of the actual class of an instance is described by a single probability p over Ω, a common decision rule consists in predicting the most probable class :

$$\hat{y} = \arg\max_{\omega \in \Omega} p(\omega), \tag{10}$$

which corresponds to minimizing the error in a 0/1 cost setting (mistakes cost 1, correct decisions 0). It can also be viewed as choosing the maximal element according to the preference order $\succ$ induced by the probability distribution over Ω, i.e., to state that ω_i is preferred to ω_j ($\omega_i \succ \omega_j$) if

$$p(\omega_i) > p(\omega_j) \Leftrightarrow p(\omega_i) - p(\omega_j) > 0. \tag{11}$$

When the information at hand is described by a set $\mathcal{P}$, imprecise probability theory offers many ways to extend this classical decision rule [33]. In a nutshell, two strategies may be considered: we may either choose a single class, or a set of possible (optimal) classes. We will describe here the maximin rule which is of the former type, and the maximality rule of the latter.

The maximin decision rule amounts to make the decision $\widehat{y}$ such that

$$\widehat{y} \stackrel{\text{def}}{=} \arg\max_{\omega \in \Omega} \underline{p}(\omega). \tag{12}$$

Notice the similarity with Equation (10), with p being replaced by its lower bound. Using this rule thus requires to compute K lower probabilities by solving K linear systems. Ties can be broken arbitrarily (typically, by picking a class at random), as in the precise probabilistic case.

The maximality rule follows a pairwise comparison approach: a class is considered as possible if it is not dominated by any other one. Under the maximality rule, a class ω_i is said to dominate ω_j, written $\omega_i \succ_M \omega_j$, if $p(\omega_i) > p(\omega_j)$ for all $p \in \mathcal{P}$: that is, if Equation (11) holds for any distribution p. Note that we may have incomparabilities (simultaneously $\omega_i \not\succ_M \omega_j$ and $\omega_j \not\succ_M \omega_i$). In practice, whether $\omega_i \succ_M \omega_j$ stands is determined by solving the optimization problem

$$\inf_{p \in \mathcal{P}} p(\omega_i) - p(\omega_j)$$

and checking if its solution is strictly positive. The final prediction

$$\widehat{Y} = \{\omega \in \Omega |\ \nexists \omega' \text{ s.t. } \omega' \succ_M \omega\} \tag{13}$$

comes down to consider all maximal (non-dominated) elements of $\succ_M$. Consequently, the set $\widehat{Y}$ can include multiple classes and thus be imprecise. Finding $\widehat{Y}$ requires to solve at most $K^2 - K$ linear programs (but usually less, as non-dominated classes can be discarded from computations as soon as they are identified). Note that the set of solutions $\widehat{Y}$ thus obtained necessarily contains the solution $\widehat{y}$ obtained via maximin [33].

Example 3. *In Example 1, we have*

$$\underline{p}(\omega_1) = 0, \quad \underline{p}(\omega_2) = 0, \quad \underline{p}(\omega_3) = 0.4,$$

hence predicting $\hat{y} = \omega_3$ using the maximin strategy. If we use the maximality rule, the only pair (ω_i, ω_j) for which $p_i - p_j$ is strictly positive is (ω_3, ω_2), since we have

$$\inf_{p \in \mathcal{P}} p(\omega_3) - p(\omega_2) = 0.2,$$

reached by the distribution $p = (0.4, 0.2, 0.4)$. This means that only ω_2 is dominated (by ω_3) according to $\succ_M$, hence $\widehat{Y} = \{\omega_1, \omega_3\}$.

3. Combining binary probability intervals

As mentioned in Section 1, using probabilistic decomposition-combination strategies consists in replacing the direct estimation of the joint posterior probability $p(y|x)$ of every class y given an instance x by binary, easier-to-get estimations. The main problem is then to recover or to infer information about the joint probability $p(y|x)$, especially when the binary estimates are inconsistent with each other.

In this section, we formalize this issue when probability estimates are interval-valued, and we propose original correction mechanisms for solving inconsistencies.

3.1. Induced joint probability set

Let A (respectively, B) be the set of positive (resp., negative) classes considered in a binary sub-problem of the decomposition strategy (see Figure 2). Note that these sets depend on the chosen strategy; the main ones will be reviewed in Section 4. When evaluating a test instance x, a classical probabilistic classifier then outputs an estimate of the conditional probability $P(A|A \cup B, x)$ (with $P(B|A \cup B) = 1 - P(A|A \cup B)$ by duality[2]).

The quality of such an estimate highly depends on the training data — in particular their amount and the level of noise. In practice, this point estimate can be replaced by an interval-valued estimate determining a set of possible conditional probabilities, by providing a pair of values bounding $P(A|A \cup B)$. This interval can reflect, for instance, the estimation uncertainty resulting from the lack of information, and can be obtained for example via confidence or credibility intervals over the estimated conditional probability. Let us denote by α_j, β_j the bounds provided by the jth classifier ($j = 1, \ldots, J$):

$$\alpha_j \leq P(A_j|A_j \cup B_j) \leq \beta_j, \tag{14}$$

[2]From now on, we will drop the x in the conditional statements, since the combination always concerns a unique instance which input features remain the same.

and by complementation

$$1 - \beta_j \leq P(B_j|A_j \cup B_j) \leq 1 - \alpha_j. \tag{15}$$

Our approach consists in combining the classifier outputs by computing the set $\mathcal{P}$ of probability distributions over Ω which are compatible with the corresponding conditional assessments. For this purpose, we first turn them into linear constraints over the unconditional probabilities to be determined. More precisely, assuming that $P(A_j \cup B_j) > 0$, we transform Equations (14) and (15) into

$$\alpha_j \leq \frac{P(A_j)}{P(A_j \cup B_j)} \leq \beta_j \text{ and } 1 - \beta_j \leq \frac{P(B_j)}{P(A_j \cup B_j)} \leq 1 - \alpha_j.$$

These two equations can then be transformed into two linear constraints over the unconditional probabilities:

$$\frac{\alpha_j}{1-\alpha_j} P(B_j) \leq P(A_j) \text{ and } P(A_i) \leq \frac{\beta_j}{1-\beta_j} P(B_j),$$

or equivalently

$$0 \leq (1-\alpha_j) \sum_{\omega_i \in A_j} p_i - \alpha_j \sum_{\omega_i \in B_j} p_i, \tag{16}$$

$$0 \leq \beta_j \sum_{\omega_i \in B_j} p_i - (1-\beta_j) \sum_{\omega_i \in A_j} p_i, \tag{17}$$

where $p_i \stackrel{\text{def}}{=} p(\omega_i)$. Such constraints define the set $\mathcal{P}$ of probability distributions that are compatible with the classifier outputs, as illustrated in Figure 2. Then, the probability bounds on this set may be retrieved by solving a linear optimization problem under Constraints (16) and (17), for $j = 1, \ldots, J$.

Note that the number of constraints grows linearly with the number J of classifiers, while the number of variables is equal to the number K of classes. As the amount of classifiers usually remains reasonable, the linear optimization problem can be efficiently solved using modern optimization techniques.

Example 4. *Let us assume that $N = 3$ classifiers provided the following outputs:*

$$\begin{aligned} P(\{\omega_1\}|\{\omega_1, \omega_2\}) &\in [0.1, 0.3], \\ P(\{\omega_1\}|\{\omega_1, \omega_3\}) &\in [0.2, 0.4], \\ P(\{\omega_2\}|\{\omega_2, \omega_3\}) &\in [0.6, 0.8]. \end{aligned}$$

According to Equations (16)-(17)*, these constraints on conditional probabilities may be transformed into the following constraints over the unconditional probabilities p_1, p_2, and p_3:*

$$1/9\, p_2 \leq p_1 \leq 3/7\, p_2,\ 1/4\, p_3 \leq p_1 \leq 2/3\, p_3,\ 3/2\, p_3 \leq p_2 \leq 4\, p_3.$$

The induced set of probability distributions is not empty, since for instance $p_1 = 0.1$, $p_2 = 0.6$ and $p_3 = 0.3$ is a feasible solution. Getting the minimal/maximal probabilities for each class then comes down to solve six optimization problems (i.e., minimizing and maximizing each of the unconditional

probabilities p_i, under the constraints mentioned above), which eventually yields the following intervals:

$$p_1 \in [0.071, 0.207] \quad p_2 \in [0.477, 0.735] \quad p_3 \in [0.176, 0.364].$$

Hence, we can safely classify the instance into ω_2, since every selection of p_1, p_2, p_3 into these intervals would give a higher value to p_2: indeed, $p_2 - p_1$ and $p_2 - p_3$ are always positive, and $\omega_2 \succ_M \omega_1$, $\omega_2 \succ_M \omega_3$. □

Note, however, that the classifier outputs may not be consistent with each other (see Example 2), for instance when the training sets are disjoint. As mentioned in the Introduction, some classifiers may then provide erroneous probability bounds, in particular when they were not trained to recognize the actual class of the test instance to be classified. It is then necessary to correct their outputs, for instance by relaxing the constraints they induce, in order to restore consistency.

3.2. Correcting inconsistent outputs

Since interval-valued classifier outputs are considered, a natural way to turn inconsistent outputs into consistent ones consists in widening these intervals so that their intersection becomes non-empty. In practice, this can be done through the discounting operation described in Section 2.2. Two approaches may be considered:

1. the discounting is applied once the conditional (non-linear) constraints are transformed into (linear) unconditional ones, that is, on Equations (16)-(17) (unconditional discounting);
2. the conditional constraints (14) themselves are discounted (conditional discounting).

As will be shown by the following discussion, the first one is preferable from several points of view.

Unconditional discounting

Consider the unconditional constraints provided by the jth classifier. In practice, Equations (16)-(17) provide lower expectation bounds equal to zero for two different functions: $\underline{E}_{f_{j,1}} = 0$ and $\underline{E}_{f_{j,2}} = 0$, with

$$f_{j,1}(x) = \begin{cases} 1-\alpha_j & \text{if } x \in A_j, \\ -\alpha_j & \text{if } x \in B_j, \\ 0 & \text{else;} \end{cases} \quad \text{and} \quad f_{j,2}(x) = \begin{cases} \beta_j - 1 & \text{if } x \in A_j, \\ \beta_j & \text{if } x \in B_j, \\ 0 & \text{else.} \end{cases}$$

Let us apply the discounting procedure given by Equation (9). If we denote by ϵ_j the discounting factor of the classifier, we obtain the following discounted equations:

$$\epsilon_j(-\alpha_j) \leq (1-\alpha_j)P(A_j) - \alpha_j P(B_j), \tag{18}$$

$$\epsilon_j(\beta_j - 1) \leq (\beta_j - 1)P(A_j) + \beta_j P(B_j). \tag{19}$$

A first remark is that when $\epsilon_j = 1$, the constraints provided by the jth classifier become trivial, which is equivalent to state that $P(A_j|A_j \cup B_j) \in [0, 1]$. This means that there always exists a set of coefficients $\{\epsilon_j\}_{j=1,\ldots,N}$ that makes the

problem feasible. A second remark is that Equations (18)-(19) are linear in variables p_i and ϵ_j, thus still allowing us to use efficient linear programming techniques.

The issue is to estimate the discounting rates ϵ_j, $j = 1, \ldots, J$. We propose to minimally relax the constraints so that the joint probability set $\mathcal{P}$ of Section 3.1 is non-empty: this amounts to make the classifier combination feasible, while preserving the classifier ouputs as much as possible. In practice, we minimize the sum of discounting coefficients:

$$\min \sum_{j=1}^{J} \epsilon_j, \tag{20}$$

under the constraints

$$\sum_{k=1}^{K} p_k = 1, \quad 0 \leq p_k \leq 1 \text{ for } k = 1, \ldots, K, \quad 0 \leq \epsilon_j \leq 1 \text{ for } j = 1, \ldots, J, \tag{21}$$

and Constraints (18)–(19). The objective function is null if and only if initial constraints are consistent. Note that this strategy is similar to the ones proposed to find minimal sets of infeasible constraints in linear programs [34].

Conditional discounting

The alternative consists in applying the discounting strategy to Equation (14); for the jth classifier, the discounted interval is then

$$(1 - \epsilon_j)\alpha_j \leq P(A_j | A_j \cup B_j) \leq \epsilon_j + (1 - \epsilon_j)\beta_j, \quad j = 1, \ldots, J. \tag{22}$$

This approach, initially proposed in [27], has the advantage of being consistent with some previous proposals made in other frameworks such as the theory of evidence [35]. However, this strategy runs into the following issue: the problem

$$\min \sum_{j=1}^{J} \epsilon_j$$

under the constraints

$$\sum_{k=1}^{K} p_k = 1, \quad 0 \leq p_k \leq 1 \text{ for all } k = 1, \ldots, K, \quad 0 \leq \epsilon_j \leq 1 \text{ for all } j = 1, \ldots, J,$$

and constraints (22) instead of (18)–(19), is quadratic with indefinite form, since all the coefficients of the square terms p_i^2 and ϵ_j^2 are zero. Its resolution is consequently much more computationally expensive. This issue remains even with other functions of the discounting coefficients ϵ_j.

To overcome this problem, all discounting factors may be assumed to be equal [27]: $\epsilon_1 = \cdots = \epsilon_J$, in which case the optimal value can be easily determined by using a dichotomic search. As a result, all classifiers are treated or discounted in the same way, ignoring the fact that only a minority may be inconsistent with the others, or that some classifiers are more reliable than others. Therefore, if only one classifier strongly disagrees with all the others (these latter being consistent with each other), all of them will nevertheless be significantly discounted. This is less likely to happen when using a specific discounting rate for each classifier, as Example 5 illustrates.

Example 5. *We consider a set of four classes* $\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4\}$*. Six classifiers are trained to separate each class* ω_i *from each other* ω_j *(*$j > i$*); they provide the following results:*

$$P(\{\omega_i\}|\{\omega_i, \omega_j\}) \in [0.6, 1] \quad \textit{for all} \quad 1 \leq i < j \leq 4,$$

for all pairs $1 \leq i < j \leq 4$*, except for* $P(\{\omega_1\}|\{\omega_1, \omega_4\}) \in [0, 0.4]$*. Thus, all classifier outputs are consistent with* $p_1 > p_2 > p_3 > p_4$*, except* $P(\{\omega_1\}|\{\omega_1, \omega_4\})$ *from which one would conclude* $p_4 > p_1$*. Now, if we were to discount all of them in the same way, we would obtain as minimal discounting rates* $\epsilon_{ij} = 1/6$ *for each classifier, with* $p_1 = p_2 = p_3 = p_4 = 1/4$ *being the only feasible solution. Thus, in this case, all the information provided by the classifiers is lost, and we are unable to choose between one of the four classes.*

Now, assume that each classifier is discounted using a specific rate; then, taking $\epsilon_{14} = 1/3$ *restores consistency (e.g.,* $p_1 = 0.5$*,* $p_2 = 0.31$*,* $p_3 = 0.2$*,* $p_4 = 0.09$ *is a solution) while still preserving the ordering* $p_1 > p_2 > p_3 > p_4$*, thus still allowing us to choose* ω_1*.*

3.3. Summary

We summarize here the different steps of our proposed method, and we provide an insight on their computational complexity. Assume that we have trained J binary classifiers, where the jth classifier must predict whether the class of an instance x belongs to subset A_j or B_j; our strategy involves:

1. collecting the outputs of the classifiers, i.e. the J intervals $[\alpha_j, \beta_j] = [\underline{P}(A_j|A_j, B_j), \overline{P}(A_j|A_j, B_j)]$;
2. solving the linear optimisation problem defined by Equation (20) under Constraints (18)–(19) and (21), and storing the discounting factors ϵ_j. This linear program, which has a polynomial complexity, has J variables and $2J + 1$ constraints, meaning that its size grows linearly in the number of classifiers.
3. Computing the final credal set $\mathcal{P}$ from which predictions will be made, which is defined by constraints (18)-(19):
   - if a precise classification is desired, the maximin rule is used: for each class ($k = 1, \ldots, K$), solve
     $$\min_p p_k$$
     under Constraints (18)–(19) (using the ϵ_j previously computed) and $\sum_{k=1}^{K} p_k = 1$, $p_k \in [0, 1]$ for all $k = 1, \ldots, K$. The predicted class is then the one that obtained the highest value. These linear program have the same constraints, and therefore the same complexity as the previous, and must be solved K times, a linear number of times in the number of classes.
   - If an imprecise classification is desired, the maximality rule consists in determining the set of undominated classes: for each pair of classes ($k = 1, \ldots, K$, $\ell = 1, \ldots, K$, and $\ell \neq k$), solve
     $$\min_p p_k - p_\ell$$
     under Constraints (18)–(19) (again, using the coefficients ϵ_j previously computed) and $\sum_{k=1}^{K} p_k = 1$, $p_k \in [0, 1]$ for all $k = 1, \ldots, K$.

Class ω_ℓ is then discarded iff the solution is strictly positive. Again, the associated linear programs have the same constraints, and must be solved at worst $K(K-1)/2$ times to get the prediction, thus a quadratic number of times in the number of classes.

Thus, $K(K-1)/2$ linear programs must at worst be solved, with a number of constraints which is linear in the number of classifiers. Using efficient resolution methods in polynomial complexity, our approach remains tractable as long as the number of classes remains reasonable (say, a few hundreds at most). The exact complexity, as well as the possibility to apply specific techniques, highly depends on the chosen binary decomposition scheme, and given our previous remark is only relevant for large-scale problems. This issue therefore arising only for specific applications, we will not deal further with it.

4. Experiments

This section presents experiments showing in which situations our approach is likely to present some advantages. These experiments are performed for four decomposition schemes, and compared to classical probabilistic strategies for combining classifiers. Before giving details about these experiments, we think useful to remind that the primary goal of imprecise probabilistic methods is not to systematically perform better than their precise counterparts, but to provide safeguards against wrong predictions in case of scarce information: thus, we can expect imprecise probabilistic approaches to be beneficial

- either when precise probabilistic estimates are only based on a limited amount of data, for example when classifiers are overfitted,
- or when decomposition schemes include fewer classifiers, leading to potential biases in precise probabilistic methods that may be avoided when considering sets of probabilities.

Although we may also expect imprecise methods to perform well when inconsistencies arise — a case that usually happens when the decomposition scheme involves a high number of classifiers, the fact that we use a minimal correction strategy of the classifier outputs (see Section 3.2) may limit the effects of using imprecise approaches in this case. In Section 5, we mention correction strategies where the resulting imprecision is proportional to the amount of conflict; note that these strategies involve extensive computations, and thus appear to be impractical in a learning setting.

4.1. Decomposition and combination strategies

Many decomposition and recombination strategies have been proposed in the literature. We cover four of them in the next sections.

4.1.1. One-versus-all

The most natural way to decompose a multi-class problem into binary ones consists in creating K binary subproblems so as to separate each class from all the others. This scheme, known as the one-versus-all (OVA) decomposition, has been explored in a number of articles. When the classifiers output decisions, the combination can be carried out according to majority voting, to the maximal

score [36, 37], or by transforming the outputs into membership functions that are then pooled using the min operator [38]. Probabilistic or credal outputs can be combined for instance by averaging [16].

Since the OVA decomposition scheme involves only K classifiers, we can expect imprecise methods to be particularly beneficial.

4.1.2. One-versus-one

Another popular approach consists in decomposing the original problem by opposing each class to each other, thus creating $(K-1)K/2$ subproblems. This one-versus-one (OVO) decomposition has received much attention in the literature. In the case of crisp outputs, the combination strategies include voting [14, 12, 39, 15, 40] and aggregating membership functions derived from the outputs [41].

As explained above, probabilistic outputs are interpreted as estimates of the conditional probabilities $P(\{\omega_i\}|\{\omega_i,\omega_j\})$. Then, a combination strategy consists in selecting some of the constraints derived from the classifier outputs to obtain as many equations as unknowns [42]. In [43], closed forms of the unconditional probability estimates are proposed. However, the most popular approaches [18, 44] involve solving an optimization problem to compute these unconditional probabilities from the classifier outputs. The work presented in [17] advocates training additional classifiers so as to estimate the probability $P(\{\omega_i,\omega_j\})$ for each pair of classes; it turns out that after this additional calibration step, solving the optimisation problem amounts to average the unconditioned pairwise probability estimates.

Since the OVO decomposition involves a large number $(K-1)K/2$ of classifiers as the number of class K increases, we expect our method to be less interesting in this setting.

4.1.3. Error-correcting output coding

Both the OVA and OVO decomposition schemes have been shown to be particular cases of a more general strategy known as error-correcting output codes (ECOC) decomposition [3]. An ECOC decomposition is described by a code matrix, composed of elements $c_{ij} \in \{-1, 0, +1\}$, in which each row (by convention) is associated with a class and each column with a classifier. A value $c_{ij} = +1$ (respectively, $c_{ij} = -1$) indicates that ω_i appears in the set of positive (resp., negative) classes in the training set of the j^{th} classifier. A null value indicates that the class does not appear in the training set of the classifier. Code matrices composed exclusively elements $c_{ij} \in \{-1, 1\}$ are referred to as *dense* matrices, in contrast with matrices containing zeros, called *sparse*.

Some results [3] relate the classification accuracy to the decomposition matrix. Two distant line vectors (according to some dissimilarity measure) mean that the corresponding classes are often separated from each other by the set of classifiers. Similarly, when a column vector (and its complement) is distinct from another one, this means that the two corresponding classifiers have different training sets: they are thus likely to be less correlated, potentially improving the classification accuracy by encouraging diversity and avoiding the redundancy of erroneous estimates. A "good" code matrix should then count enough columns for the classes to be well separated, but not too much (since for a fixed number of classes, adding new columns eventually results in creating identical classifiers).

The OVA scheme is retrieved for a square code matrix where the diagonal elements are $+1$ and the others -1; the OVO decomposition, for a $K \times (K-1)K/2$ matrix where each column counts two non-zero elements, one being positive and the other one being negative (see Figure 5).

$$\begin{pmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{pmatrix} \qquad \begin{pmatrix} +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{pmatrix}$$

Figure 5: Code matrices for the OVA (left) and OVO (right) decomposition schemes, for a $K = 4$ class problem.

Classifiers providing crisp outputs can be combined by classifying instances into the closest class according to the Hamming distance [3]. Various strategies have been proposed to combine probabilistic classifiers: Passerini et al. [45] derived a closed form estimate for the posterior probabilities of the classes under a mild conditional independence assumption. In [46], an iterative algorithm based on the Kullback-Leibler divergence, similar to that proposed in [18], is described. Eventually, [47] formalized the problem and proposed an algorithm based on the same divergence, which was shown to boil down to the algorithm [18] in the OVO case.

4.1.4. Directed Acyclic Graphs (DAGs)

A variant of the aforementioned strategies consists in recursively eliminating one class from the set of (remaining) classes [48]: a classifier separates class ω_1 from the other ones ($\Omega \setminus \{\omega_1\}$), another one separates class ω_2 from $\Omega \setminus \{\omega_1, \omega_2\}$, etc. Alternatives consider hierarchical sets of classifiers, each of them reducing the set of possible classes [49, 50].

The instances are then evaluated in a sequential manner, rather than independently by the various classifiers. The choice of the classes to be separated from the others is known to have a significant influence on the results, and highly depends on the application and on the data considered. For this reason, and since our aim is to demonstrate that our approach is versatile and adapts well to a wide range of situations, such approaches will not be considered here.

4.2. Experimental setup

4.2.1. Datasets

We compared our approach to the existing strategies on various datasets, briefly described in Table 1. Most of them were obtained from the UCI Machine Learning Repository, except for the faces dataset, which was obtained from the CMU-Pittsburgh image database. It consists in 216 images of size 60×70, corresponding to six facial expressions (joy, surprise, sadness, disgust, anger, fear). Each was obtained by asking a person to mimic a specified expression (note that according to [51], we used aligned and cropped versions of the images). The images were then presented to five subjects, who assessed their beliefs for each of the six expressions; the label for each image was then obtained by voting.

The data were randomly divided into training and test sets, respectively counting roughly $2/3$ and $1/3$ of the instances. We retained $2/3$ of the whole sample

dataset name	#classes K	#input features	#samples
ecoli	8	7	336
faces	6	4200	216
glass	6	9	214
iris	3	4	150
pageblocks	5	10	5473
satimage	6	36	6435
segment	7	19	2310
USPS	10	256	9298
vowel	11	10	990
waveform	3	21	5000
wine	3	13	178
yeast	10	8	1484

Table 1: UCI dataset characteristics

to learn the models, the remaining 1/3 being used for computing error rates. In order to provide confidence intervals over the classification error, 25 datasets were thus randomly generated from the original data. The various combination strategies were evaluated on each of these datasets so as to determine comparable average error rates.

4.2.2. Base classification algorithm

CART decision trees [52] were used as a base classification technique to obtain conditional probability estimates $P(A_j|A_j, B_j)$. This algorithm presents the advantage to provide region-dependent class information and probability estimates. In order to distinguish between high and low density regions, the upper and lower probability bounds $\alpha_j = \underline{P}(A_j|A_j, B_j)$ and $\beta_j = P(A_j|A_j, B_j)]$ were defined as the Bayesian credibility interval bounds obtained using a beta prior with parameters $a_j = b_j = 3.5$ (see [53, p. 261]). This essentially means that a given leaf must contain seven instances for the likelihood to have the same weight as the prior. An imprecise classifier output thus takes into account the level of information in the neighborhood of a test instance, since the interval width depends on the amount of training samples in the region of the input space where the test instance is located by the tree.

We remark here that other classifiers such as logistic regression [54] or calibrated SVMs [55] would be less appropriate, since the width of a confidence interval is not region-dependent (it depends on the size of the whole training set). Also for this reason, we expect our strategy to give good results even when unpruned decision trees are used, since leaves associated with very few training instances will be characterized by large credibility intervals and thus cautious classifier outputs. As a consequence, we conducted experiments using both unpruned and pruned trees (then using cross-validation as pruning strategy).

Note that our aim is here to compare different decomposition strategies, not to determine the best base classifier for a given problem. For this latter purpose, recent approaches to decision tree induction [56, 57] may be used; alternatively, dimensionality reduction techniques [58, 59] can be used so as to improve the set of considered features and increase the classification performances. Similarly,

base classifier selection techniques [60] could be used to improve the quality of the results.

4.2.3. Decomposition and combination schemes

We considered four different decomposition strategies: OVA, OVO, and ECOC using dense and sparse code matrices. In the ECOC dense case, all the classes are considered for training each classifier (i.e., the code matrix does not contain zero elements), which is not the case in the sparse case. Since designing the optimal code matrix is NP-hard [61], we used a rule-of-thumb strategy: positive and negative classes were randomly chosen with the same probability — 1/2 in dense matrices, and 1/4 in sparse ones (in which case zero elements have probability 1/2). In the dense case, $10\log_2(K)$ classifiers were generated, a number which was raised up to $15\log_2(K)$ in the sparse case.

We compared our approach using maximin to the Bradley-Terry combination scheme [47] (or [18] in the OVO case), the estimation strategy proposed by Passerini *et al.* [45], and the combination procedure proposed by Zadrozny [46]. The accuracy of the Hamming strategy was also given as a baseline. Furthermore, in the OVO case, we also provided the results obtained via the two combination strategies described by Wu *et al.* [44], which are specific to this decomposition scheme. Note that as we work in a probabilistic setting, we only compared our approach to other proposals aiming at combining binary conditional probabilities provided by binary classifiers.

4.2.4. Accuracy measures for imprecise predictions

When combining imprecise classifiers using our approach, we considered both the maximin rule, resulting in precise predictions, and the maximality rule presented in Section 2 which computes the set of possible classes induced by the classifier outputs. In this latter case, set accuracy and discounted accuracy were computed to evaluate the classification accuracy. We recall here that the former consists in considering the set $\widehat{Y}$ of predicted classes as correct whenever it contains the actual class, and can thus be interpreted as an optimistic indicator of the classification accuracy which clearly favors producing sets of classes as predictions:

$$\text{set acc.} = \mathbb{1}_{y\in\widehat{Y}}.$$

The latter rule, however, rewards decisions with low cardinality:

$$\text{disc. acc.} = \frac{1}{|\widehat{Y}|}\mathbb{1}_{y\in\widehat{Y}};$$

it is statistically equivalent to choosing a class at random among the set $\widehat{Y}$ of predicted classes. In fact, it has been shown by Zaffalon *et al.* [62] to penalize set-valued predictions, as it does not reward caution, as is shown by Example 6. It should be noted that when all predictions are precise, both set and discounted accuracy reduce to usual accuracy.

Example 6. *Let us consider three classes* $\{\omega_1, \omega_2, \omega_3\}$*, and a test sample with actual class* $y = \omega_2$*. Then, if our prediction is* $\widehat{Y} = \{\omega_1, \omega_2\}$*, we have*

$$\text{set acc.} = 1$$

since ω_2 *is indeed included in* $\widehat{Y}$*, and*

$$\text{disc. acc.} = 1/2$$

since $|\widehat{Y}| = 2$ *(two potential classes were predicted).*

4.3. Results

The results obtained using unpruned decision trees for the OVA, dense ECOC, sparse ECOC and OVO decomposition schemes are given in Tables 2 to 5; those obtained with pruned decision trees in the OVA, dense ECOC, sparse ECOC and OVO cases are given in Tables 6 to 9. In each case, the best result was underlined. Furthermore, the result obtained via maximin was printed in bold if it outperformed the results obtained using any precise probabilistic strategy (and conversely, the result obtained with a precise approach was printed in bold if it outperformed maximin).

Note that the results obtained using set accuracy and discounted accuracy were left out of the comparison, since they do not provide a single decision. They however give an insight into the number and quality of imprecise predictions: a set-accuracy close to the maximin accuracy suggests that there were only a few imprecise predictions; conversely, should it be critically lower, would it reflect an important number of decision indeterminacy. Whether or not these imprecise predictions allowed to correct mistakes (by including the true class) can be evaluated by comparing the discounted and maximin accuracy: a significantly higher value means that imprecision was damaging, while a lower or comparable value indicates that they were meaningful.

Overall, we can bring the following conclusions out of the results obtained:

- our method for combining imprecise probabilistic classifiers performs better than the other approaches in the OVA case, when using unpruned decision trees, and to a lesser extent in the dense ECOC case (still with unpruned decision trees). Since this corresponds to the case where we have the fewer classifiers (which are furthermore likely to be overfitted in the precise case), this corresponds to our earlier expectations.

- When using pruned decision trees, our method compares favourably to the other ones in the OVA case (again, most differences being non significant), but generally performs worse than the best method in the dense, sparse and OVO cases; it is, however, seldom the worst one.

- The low differences between set accuracy and discounted accuracy in the OVO, sparse and dense ECOC case for pruned trees confirm that in these cases, there are very few imprecise predictions, due to the fact that we have a high number of regularized classifiers and a minimal correction strategy.

- Quite remarkably, combining unpruned trees via our method almost always performs better than combining pruned trees (the exceptions being for the Iris dataset, in the ECOC dense and sparse cases, where the results are almost identical). In addition, it almost always outperforms (or at least equals) a precise approach with pruned trees. This can be explained as follows. The overfitting of unpruned trees, which generally

degrades generalization performances, is here counterbalanced by the fact that when a given leaf is associated with very few data, the associated interval will be large. This is clearly an advantage of our technique, since pruning a decision tree may deteriorate the classification accuracy in the case of small datasets (which is exactly the case where our approach is expected to perform especially well).

In general, we can remark that a high level of imprecision (i.e., a significant difference between set accuracy and maximin) generally coincides with a good performance of our strategy. To confirm this insight, Table 10 provides some statistics about the instances for which our method provided imprecise predictions in the OVA case with unpruned trees (Table 2). In addition to the performance measures of the other tables, we also added the average number of imprecise predictions and the corresponding average cardinality of $\mathcal{P}$. A first remark is that if we look at the results of the classical precise methods, the error rates are critically higher than for the whole training set (e.g., for the Pageblocks dataset, it increases from 5% to 53.5% for the Bradley-Terry combination strategy, the difference being somehow less critical for the Yeast and Glass datasets). This confirms that those instances for which we produce imprecise predictions are difficult to classify with classical approaches.

We can also see that for these imprecise instances, set accuracy is always remarkably lower than any error rate. Thus, being cautious by making imprecise decisions is worth the effort, since it significantly reduces the risk of making classification errors. Eventually, the results show that the maximin performs generally much worse than (the ideal measure of) set accuracy, although it keeps performing better than classical approaches. This suggests that the classification accuracy obtained via maximin may be further improved, for instance by considering strategies involving human judgement whenever several classes are in competition for a given instance. This seems reasonable since the average number of retained classes is generally low (i.e., between 2 and 3).

5. Conclusions

In this article, we presented an approach to solve multiclass classification problems by combining imprecise binary classifiers. Each classifier is trained to separate two sets of classes of the original training set. It is assumed to provide bounds on the conditional probabilities that an instance belongs to the sets of considered classes. The classifiers are combined by computing the set of probability distributions which are consistent with their outputs. The bounds are first expressed as constraints on the unconditional probabilities of the classes. Then, the maximality rule can be used to determine the set of plausible (undominated) classes. Alternativey, should a single decision be made, the maximin rule returns the class with highest lower probability. Inconsistencies are resolved by discounting the classifier outputs so as to find at least a probability distribution which is consistent with the classifier outputs.

We conducted experiments for various decomposition strategies: one-versus-all, one-versus-one, and error-correcting output codes with dense and sparse code matrices. The results obtained show that our method performs well when using unpruned decision trees, and in particular when few information is available (i.e., few classifiers are combined). This confirms that using an imprecise

approach is indeed beneficial when information is scarce. In this case, lowering the influence of inaccurate assessments according to this lack of information makes it possible to increase robustness. This ability to take into account classifier uncertainty makes our approach well-suited to classification problems when information is scarce.

Future works may either be focused on expanding this first proposal or solving some of its identified issues. For instance, the credal set obtained by our method after a correction step (see Section 3.2) will always be fairly precise. An alternative would be to design methods for which the level of imprecision of a corrected credal set reflects the inconsistency between the constraints. Quaeghebeur [63] proposed an elegant approach for this purpose, which however requires to solve multi-objective optimization problems. This technique would hence only be tractable for datasets with a small number of classes. Approximate solutions would therefore be needed for the case of large datasets. Another issue would be to investigate decision rules other than the maximin (e.g., maximum entropy), since this latter performs sometimes poorly compared to the ideal case of set accuracy. The impact of using other binary classification techniques handling noisy data [64] should also be assessed, since imprecise probabilistic approaches may then be useful.

Finally, we might consider adding human feedback to the system when providing imprecise classification. We may also extend this current work to other settings where pairwise decomposition strategies are common: such as in multilabel classification, preference learning or ranking [65]. From a theoretical perspective, we may investigate related areas involving combining binary information, such as multiple hypothesis testing [66].

Acknowledgements

This work was carried out in the framework of the Labex MS2T, which was funded by the French Government, through the program "Investments for the future" by the National Agency for Research (reference ANR-11-IDEX-0004-02).

References

[1] Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

[2] Gérard Biau, Luc Devroye, and Gábor Lugosi. Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9(Sep):2015–2033, 2008.

[3] Thomas G. Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[4] Eyke Hüllermeier and Stijn Vanderlooy. Combining predictions in pairwise classification: An optimal adaptive voting strategy and its relation to weighted voting. *Pattern Recognition*, 43(1):128–142, 2010.

[5] Lior Rokach. Decomposition methodology for classification tasks: a meta decomposer framework. *Pattern Analysis and Applications*, 9(2-3):257–271, 2006.

[6] Ping Zhang, Tien D. Bui, and Ching Y. Suen. A novel cascade ensemble classifier system with a high recognition performance on handwritten digits. *Pattern Recognition*, 40:3415–3429, 2007.

[7] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44:1761–1776, 2011.

[8] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.

[9] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[10] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[11] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[12] Johannes Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2:721–747, 2002.

[13] Lei Xu, Adam Krzyzak, and Ching Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Trans. on Systems, Man and Cybernetics*, 22(3):418–435, 1992.

[14] Jerome H. Friedman. Another approach to polychotomous classification. Technical report, Department of Statistics and Stanford Linear Accelerator Center, Stanford University, 1996.

[15] F. Cutzu. Polychotomous classification with pairwise classifiers: a new voting principle. In *Multiple Classifier Systems*, pages 115–124, 2003.

[16] P. Vannoorenberghe and T. Denœux. Handling uncertain labels in multiclass problems using belief decision trees. In *Proceedings of the 9th international conference on Information Processing with the Management of Uncertainty in knowledge-based systems (IPMU'02)*, volume 3, pages 1919–1926, 2002.

[17] Miguel Moreira and Eddy Mayoraz. Improved pairwise coupling classification with correcting classifiers. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*, volume LNAI-1398 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1998.

[18] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. *The Annals of Statistics*, 26(2):451–471, 1998.

[19] B. Quost, T. Denœux, and M.-H. Masson. Pairwise classifier combination using belief functions. *Pattern Recognition Letters*, 28(5):644–653, April 2007.

[20] David H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[21] Bertrand Clarke. Comparing bayes model averaging and stacking when model approximation error cannot be ignored. *Journal of Machine Learning Research*, 4:683–712, 2003.

[22] Venkataraman Santhanam, Vlad I. Morariu, David Harwood, and Larry S. David. A non-parametric approach to extending generic binary classifiers for multi-classification. *Pattern Recognition*, 58:149–158, 2016.

[23] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. Dynamic classifier selection for one-vs-one strategy: Avoiding non-competent classifiers. *Pattern Recognition*, 46:3412–3424, 2013.

[24] Fabrice Janez and Alain Appriou. Théorie de l'évidence et cadres de discernement non exhaustifs. *Traitement du Signal*, 13(3):237–250, 1996.

[25] Fabrice Janez and Alain Appriou. Theory of evidence and non-exhaustive frames of discernment: plausibilities correction methods. *International Journal of Approximate Reasoning*, 18(1-2):1–19, 1998.

[26] Alain Appriou. Approche générique de la gestion de l'incertain dans les processus de fusion multisenseur. *Traitement du Signal*, 22(4):307–319, 2005.

[27] Sébastien Destercke and Benjamin Quost. Combining binary classifiers with imprecise probabilities. In Y. Tang, V.-N. Huynh, and J. Lawry, editors, *Integrated Uncertainty in Knowledge Modelling and Decision Making*, volume 7027 of *Lecture Notes in Computer Science*, pages 219–230. Springer, 2011.

[28] Sébastien Destercke and Benjamin Quost. Correcting binary imprecise classifiers: Local vs global approach. In E. Hüllermeier, S. Link, T. Fober, and B. Seeger, editors, *6th International Conference on Scalable Uncertainty Management*, volume 7520 of *Lecture Notes in Computer Science*, pages 299–310. Springer, 2012.

[29] P. Walley. *Statistical reasoning with imprecise Probabilities*. Chapman and Hall, New York, 1991.

[30] Thomas Augustin, Frank PA Coolen, Gert de Cooman, and Matthias CM Troffaes. *Introduction to imprecise probabilities*. John Wiley & Sons, 2014.

[31] J. O. Berger. An overview of robust Bayesian analysis. *Test*, 3:5–124, 1994. With discussion.

[32] E. Quaeghebeur and G. De Cooman. Extreme lower probabilities. *Fuzzy Sets and Systems*, In press, 2008.

[33] M.C.M. Troffaes. Decision making under uncertainty using imprecise probabilities. *International Journal of Approximate Reasoning*, 45:17–29, 2007.

[34] John W Chinneck. Finding a useful subset of constraints for analysis in an infeasible linear program. *INFORMS Journal on Computing*, 9:164–174, 1997.

[35] David Mercier, Benjamin Quost, and Thierry Denoeux. Refined modeling of sensor reliability in the belief function framework using contextual discounting. *Information Fusion*, 9(2):246–258, 2008.

[36] R. Anand, K. Mehrotra, C. K. Mohan, and S. Ranka. Efficient classification for multiclass problems using modular neural networks. *IEEE Trans. on Neural Networks*, 6(1):117–124, 1995.

[37] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. Jackel, Y. Le Cun, U. A. Muller, E. Sackinger, P. Simard, and V. Vapnik. Comparison of classifier methods: a case study in handwritten digit recognition. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, volume 2, pages 77–82, 1994.

[38] T. Inoue and S. Abe. Fuzzy support vector machines for pattern classification. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1449–1454, 2001.

[39] U. Kressel. Pairwise classification and support vector machines. In *Advances in Kernel Methods: Support Vector Learning*. The MIT Press, 1999.

[40] Bo Liu, Zhifeng Hao, and Eric CC Tsang. Nesting one-against-one algorithm based on svms for pattern classification. *IEEE Transactions on Neural Networks*, 19(12):2044–2052, 2008.

[41] S. Abe and T. Inoue. Fuzzy support vector machines for multiclass problems. In *Proceedings of the European Symposium on Artificial Neural Networks*, pages 113–118, 2002.

[42] Philippe Refregier and François Vallet. Probabilistic approach for multiclass classification with neural networks. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 1003 – 1007, Amsterdam, 1991. North-Holland.

[43] David Price, Stefan Knerr, Léon Personnaz, Gérard Dreyfus, et al. Pairwise neural network classifiers with probabilistic outputs. In *Advances in Neural Information Processing Systems*, volume 7, pages 1109–1116, 1995.

[44] T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 2(5):975–1005, 2004.

[45] A. Passerini, M. Pontil, and P. Frasconi. New results on error correcting output codes of kernel machines. *IEEE Transactions on Neural Networks*, 15(1):45–54, 2004.

[46] B. Zadrozny. Reducing multiclass to binary by coupling probability estimates. In *Advances in Neural Information Processing Systems*, volume 14, 2002.

[47] T.-K. Huang, R. C. Weng, and C.-J. Lin. Generalized Bradley-Terry models and multi-class probability estimates. *Journal of Machine Learning Research*, 7:85–115, 2006.

[48] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multi-class classification. In *Advances in Neural Information Processing Systems*, volume 12, pages 547–553, 2000.

[49] B. Kijsirikul, N. Ussivakul, and S. Meknavin. Adaptive directed acyclic graphs for multiclass classification. In *Proceedings of the 7th Pacific Rim International Conference on Artificial Intelligence*, pages 158–168, 2002.

[50] F. Takahashi and S. Abe. Optimizing directed acyclic graph support vector machines. In *Proceedings of the Conference on Artificial Neural Networks in Pattern Recognition*, volume 1, pages 166–170, 2003.

[51] Séverine Dubuisson, Franck Davoine, and Marie-Hélène Masson. A solution for facial expres- sion representation and recognition. *Signal Processing: Image Communication*, 17(9):657–673, 2002.

[52] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[53] Christian P. Robert. *The Bayesian choice*. Springer, 2007.

[54] David R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 215–242, 1958.

[55] Antonio Bella, Cèsar Ferri, José Hernández-Orallo, and María José Ramírez-Quintana. Calibration of machine learning models. *Handbook of Research on Machine Learning Applications*, pages 128–146, 2009.

[56] Mathieu Serrurier and Henri Prade. Entropy evaluation based on confidence intervals of frequency estimates: Application to the learning of decision trees. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1576–1584, 2015.

[57] Carlos J Mantas and Joaquín Abellán. Credal-C4.5: Decision tree based on imprecise probabilities to classify noisy data. *Expert Systems with Applications*, 41(10):4625–4637, 2014.

[58] Jieping Ye and Qi Li. A two-stage linear discriminant analysis via qr-decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):929–941, 2005.

[59] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286, 2006.

[60] Anderson Rocha and Siome Klein Goldenstein. Multiclass from binary: Expanding one-versus-all, one-versus-one and ECOC-based approaches. *IEEE Transactions on Neural Networks and Learning Systems*, 25(2):289–302, 2014.

[61] K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. *Machine learning*, 47(2-3):201–233, 2002.

[62] Marco Zaffalon, Giorgio Corani, and Denis Mauá. Evaluating credal classifiers by utility-discounted predictive accuracy. *International Journal of Approximate Reasoning*, 53(8):1282–1301, 2012.

[63] Erik Quaeghebeur. Characterizing coherence, correcting incoherence. *Int. J. Approx. Reasoning*, 56:208–223, 2015.

[64] Aditya Menon, Brendan V Rooyen, Cheng S Ong, and Bob Williamson. Learning from corrupted binary labels via class-probability estimation. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 125–134, 2015.

[65] Nicolas Usunier, David Buffoni, and Patrick Gallinari. Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th annual international conference on machine learning*, pages 1057–1064. ACM, 2009.

[66] Domenico Ciuonzo, A De Maio, and P Salvo Rossi. A systematic framework for composite hypothesis testing of independent bernoulli trials. *IEEE Signal Processing Letters*, 22(9):1249–1253, 2015.

Table 2: Error rates (%), unpruned decision trees, OVA

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Hamming	Passerini	Zadrozny
Ecoli	8.1 ± 1.0	17.1 ± 0.9	**12.1 ± 1.0**	19.5 ± 1.6	15.5 ± 1.4	19.5 ± 1.6	17.8 ± 1.5
Faces	44.1 ± 2.5	47.4 ± 2.3	**47.0 ± 2.4**	65.6 ± 2.5	52.3 ± 2.5	65.6 ± 2.5	60.0 ± 2.6
Glass	23.3 ± 2.0	38.1 ± 2.0	**33.4 ± 2.1**	49.8 ± 2.2	40.1 ± 2.2	49.8 ± 2.2	48.2 ± 2.0
Iris	3.8 ± 1.3	6.7 ± 1.2	6.1 ± 1.3	7.1 ± 1.4	6.4 ± 1.2	7.1 ± 1.4	6.7 ± 1.3
Pageblocks	2.2 ± 0.2	3.8 ± 0.2	**3.4 ± 0.2**	5.0 ± 0.2	4.1 ± 0.2	5.0 ± 0.2	4.5 ± 0.2
Satimage	11.6 ± 0.3	14.6 ± 0.3	**14.0 ± 0.3**	23.3 ± 0.4	17.9 ± 0.4	23.3 ± 0.4	20.6 ± 0.3
Segment	4.2 ± 0.4	5.0 ± 0.4	**4.5 ± 0.4**	7.4 ± 0.4	5.6 ± 0.4	7.4 ± 0.4	6.7 ± 0.4
USPS	28.5 ± 0.3	30.3 ± 0.3	**29.8 ± 0.3**	37.7 ± 0.3	32.9 ± 0.3	37.7 ± 0.3	34.7 ± 0.3
Vowel	22.2 ± 1.2	31.8 ± 1.0	**26.8 ± 1.3**	39.2 ± 1.5	31.4 ± 1.4	39.2 ± 1.5	34.9 ± 1.4
Waveform	21.2 ± 0.4	24.4 ± 0.4	**24.3 ± 0.4**	39.3 ± 0.4	27.5 ± 0.4	39.3 ± 0.4	34.7 ± 0.4
Wine	7.0 ± 1.3	8.6 ± 1.3	**7.9 ± 1.3**	16.6 ± 1.8	10.0 ± 1.4	16.6 ± 1.8	14.0 ± 2.0
Yeast	20.1 ± 0.6	44.1 ± 0.8	**33.1 ± 1.0**	45.7 ± 1.1	38.8 ± 1.2	45.7 ± 1.1	41.2 ± 1.2

Table 3: Error rates (%), unpruned decision trees, dense code matrix

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Hamming	Passerini	Zadrozny
Ecoli	8.7 ± 1.2	10.3 ± 1.1	8.8 ± 1.1	9.9 ± 1.3	9.9 ± 1.2	19.7 ± 1.6	14.4 ± 1.4
Faces	28.9 ± 2.3	28.9 ± 2.3	28.9 ± 2.3	31.3 ± 2.6	31.4 ± 3.0	74.7 ± 1.6	43.6 ± 2.8
Glass	28.2 ± 2.2	29.7 ± 2.1	29.1 ± 2.1	29.0 ± 2.0	28.6 ± 2.3	46.0 ± 2.4	39.5 ± 2.4
Iris	3.8 ± 1.3	6.7 ± 1.2	6.1 ± 1.3	6.4 ± 1.3	6.3 ± 1.2	7.1 ± 1.4	6.7 ± 1.3
Pageblocks	2.5 ± 0.1	3.5 ± 0.2	3.3 ± 0.2	3.5 ± 0.1	3.6 ± 0.1	4.0 ± 0.1	4.1 ± 0.2
Satimage	10.0 ± 0.2	10.0 ± 0.2	10.0 ± 0.2	10.1 ± 0.3	10.3 ± 0.3	19.4 ± 0.3	15.7 ± 0.3
Segment	2.4 ± 0.3	2.4 ± 0.3	2.4 ± 0.3	2.3 ± 0.3	2.3 ± 0.3	6.4 ± 0.5	4.6 ± 0.4
USPS	21.5 ± 0.2	21.5 ± 0.2	21.5 ± 0.2	21.5 ± 0.2	21.6 ± 0.2	41.2 ± 0.3	22.8 ± 0.5
Vowel	12.6 ± 1.0	12.6 ± 1.0	12.6 ± 1.0	**9.8 ± 0.9**	10.2 ± 0.8	50.6 ± 1.0	23.6 ± 1.2
Waveform	21.2 ± 0.4	24.4 ± 0.4	**24.3 ± 0.4**	27.9 ± 0.4	27.6 ± 0.4	33.9 ± 0.4	33.9 ± 0.4
Wine	7.0 ± 1.3	8.6 ± 1.3	7.9 ± 1.3	10.7 ± 1.4	10.0 ± 1.6	12.3 ± 1.4	14.8 ± 1.5
Yeast	28.9 ± 0.9	32.0 ± 0.9	**29.5 ± 0.9**	32.5 ± 0.9	32.4 ± 0.9	49.8 ± 1.0	37.9 ± 0.9

Table 4: Error rates (%), unpruned decision trees, sparse code matrix

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Hamming	Passerini	Zadrozny
Ecoli	10.3 ± 1.3	10.3 ± 1.3	10.3 ± 1.3	9.7 ± 1.0	10.2 ± 1.1	14.9 ± 1.5	11.5 ± 1.5
Faces	30.1 ± 2.7	30.1 ± 2.7	30.1 ± 2.7	30.7 ± 2.7	31.6 ± 2.5	48.9 ± 2.5	35.4 ± 3.0
Glass	30.5 ± 2.1	31.6 ± 2.2	31.3 ± 2.2	29.5 ± 2.3	30.1 ± 1.7	35.0 ± 2.1	31.2 ± 2.2
Iris	3.3 ± 0.8	6.0 ± 0.9	5.6 ± 1.0	6.2 ± 1.1	6.2 ± 1.1	6.2 ± 1.1	6.6 ± 1.4
Pageblocks	3.6 ± 0.2	3.6 ± 0.2	3.6 ± 0.2	3.5 ± 0.1	3.5 ± 0.1	3.6 ± 0.1	3.6 ± 0.1
Satimage	10.6 ± 0.2	10.6 ± 0.2	10.6 ± 0.2	10.8 ± 0.2	12.1 ± 0.3	12.7 ± 0.2	11.6 ± 0.2
Segment	2.7 ± 0.3	2.7 ± 0.3	2.7 ± 0.3	2.8 ± 0.3	2.8 ± 0.2	3.4 ± 0.3	2.8 ± 0.3
USPS	21.7 ± 0.3	21.7 ± 0.3	21.7 ± 0.3	21.7 ± 0.2	23.0 ± 0.4	31.0 ± 0.3	22.4 ± 0.3
Vowel	15.6 ± 0.9	15.6 ± 0.9	15.6 ± 0.9	**12.3 ± 0.7**	13.2 ± 0.9	28.9 ± 1.0	15.2 ± 1.0
Waveform	19.6 ± 0.4	20.5 ± 0.4	20.5 ± 0.4	21.0 ± 0.4	21.0 ± 0.4	21.3 ± 0.4	30.7 ± 0.4
Wine	6.5 ± 1.2	6.9 ± 1.3	7.1 ± 1.4	6.2 ± 1.5	6.4 ± 1.5	6.4 ± 1.5	10.3 ± 1.7
Yeast	31.9 ± 0.8	32.2 ± 0.8	32.1 ± 0.8	31.4 ± 0.7	31.9 ± 0.6	39.1 ± 0.9	32.4 ± 0.7

Table 5: Error rates (%), unpruned decision trees, OVO

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Wu 1	Wu 2	Hamming	Passerini	Zadrozny
Ecoli	12.1 ± 1.4	12.2 ± 1.4	12.2 ± 1.4	12.3 ± 1.3	11.3 ± 1.4	12.1 ± 1.4	11.7 ± 1.4	11.5 ± 1.3	12.2 ± 1.4
Faces	33.7 ± 2.4	33.9 ± 2.4	33.8 ± 2.4	38.3 ± 2.7	36.5 ± 2.6	38.8 ± 2.4	36.7 ± 2.2	38.9 ± 2.4	38.6 ± 2.8
Glass	25.3 ± 2.0	33.1 ± 1.5	30.4 ± 1.8	29.6 ± 2.0	28.6 ± 2.1	28.1 ± 1.9	27.7 ± 1.7	28.3 ± 2.1	29.4 ± 2.0
Iris	2.8 ± 0.8	6.3 ± 0.9	6.0 ± 1.2	6.2 ± 1.4	6.2 ± 1.4	6.0 ± 1.2	6.2 ± 1.4	6.0 ± 1.2	6.2 ± 1.4
Pageblocks	3.8 ± 0.2	3.9 ± 0.2	3.9 ± 0.2	3.7 ± 0.1	3.7 ± 0.1	3.8 ± 0.1	3.7 ± 0.2	3.7 ± 0.1	3.7 ± 0.1
Satimage	12.9 ± 0.2	12.9 ± 0.2	**12.9 ± 0.2**	14.6 ± 0.3	14.1 ± 0.2	14.5 ± 0.3	14.0 ± 0.3	14.7 ± 0.2	14.6 ± 0.3
Segment	4.5 ± 0.3	4.5 ± 0.3	4.5 ± 0.3	4.3 ± 0.3	**3.8 ± 0.3**	4.1 ± 0.3	3.9 ± 0.3	4.3 ± 0.4	4.3 ± 0.4
USPS	24.4 ± 0.2	24.4 ± 0.2	24.4 ± 0.2	26.0 ± 0.2	24.8 ± 0.2	25.5 ± 0.2	24.9 ± 0.2	25.9 ± 0.2	25.9 ± 0.2
Vowel	27.6 ± 1.2	27.8 ± 1.2	27.8 ± 1.2	24.6 ± 1.0	**21.6 ± 1.1**	23.8 ± 1.2	21.9 ± 1.3	24.5 ± 1.2	24.8 ± 1.0
Waveform	15.8 ± 0.4	23.4 ± 0.3	**23.3 ± 0.3**	25.0 ± 0.3	24.5 ± 0.3	24.5 ± 0.3	24.5 ± 0.3	24.5 ± 0.3	24.4 ± 0.3
Wine	6.2 ± 1.5	9.6 ± 1.5	7.7 ± 1.5	8.3 ± 1.7	7.6 ± 1.5	7.9 ± 1.7	7.9 ± 1.7	8.1 ± 1.8	7.5 ± 1.6
Yeast	38.0 ± 0.9	38.3 ± 0.9	38.4 ± 0.9	34.0 ± 0.8	33.3 ± 0.8	34.2 ± 0.8	**33.1 ± 0.9**	34.0 ± 0.9	33.9 ± 0.7

Table 6: Error rates (%), pruned decision trees, OVA

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Hamming	Passerini	Zadrozny
Ecoli	14.2 ± 2.1	18.7 ± 1.9	16.6 ± 1.8	16.8 ± 1.3	19.5 ± 2.3	16.8 ± 1.3	21.3 ± 2.0
Faces	36.0 ± 7.4	58.7 ± 2.7	59.8 ± 3.1	61.3 ± 3.5	61.7 ± 2.7	60.6 ± 3.2	63.1 ± 2.1
Glass	28.5 ± 3.2	40.9 ± 3.0	39.4 ± 3.1	40.6 ± 3.0	51.1 ± 4.8	40.6 ± 3.0	46.3 ± 3.4
Iris	5.2 ± 1.4	6.7 ± 1.3	6.1 ± 1.3	7.0 ± 1.4	6.8 ± 1.3	7.0 ± 1.4	6.7 ± 1.2
Pageblocks	4.8 ± 0.6	4.9 ± 0.6	4.9 ± 0.6	4.9 ± 0.8	4.9 ± 0.6	4.9 ± 0.8	5.5 ± 0.7
Satimage	27.2 ± 2.9	27.2 ± 2.9	27.2 ± 2.9	**19.8 ± 0.5**	25.2 ± 2.4	**19.8 ± 0.5**	25.1 ± 1.7
Segment	5.8 ± 0.5	6.2 ± 0.5	6.0 ± 0.5	5.7 ± 0.4	6.2 ± 0.4	5.7 ± 0.4	7.2 ± 0.5
USPS	39.4 ± 1.2	39.5 ± 1.2	39.5 ± 1.2	**34.9 ± 0.9**	38.4 ± 1.0	**34.9 ± 0.9**	37.8 ± 0.9
Vowel	30.5 ± 1.9	36.4 ± 1.6	33.5 ± 1.9	35.2 ± 1.3	37.1 ± 1.7	35.2 ± 1.4	38.9 ± 1.7
Waveform	39.8 ± 2.5	40.6 ± 2.6	40.7 ± 2.6	**32.9 ± 2.7**	39.4 ± 3.3	**32.9 ± 2.7**	40.6 ± 3.3
Wine	11.0 ± 2.1	11.7 ± 2.0	11.3 ± 2.1	12.7 ± 2.1	11.7 ± 1.7	12.7 ± 2.1	14.8 ± 2.0
Yeast	43.0 ± 2.5	54.4 ± 2.9	53.3 ± 3.1	51.1 ± 4.0	62.8 ± 3.9	51.1 ± 4.0	51.3 ± 4.0

Table 7: Error rates (%), pruned decision trees, dense code matrix

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Hamming	Passerini	Zadrozny
Ecoli	11.6 ± 0.9	11.6 ± 0.9	11.6 ± 0.9	9.9 ± 1.2	10.1 ± 1.3	17.2 ± 1.3	13.5 ± 1.5
Faces	42.5 ± 3.2	42.5 ± 3.2	42.5 ± 3.2	**37.4 ± 2.4**	43.8 ± 3.7	43.4 ± 2.3	38.3 ± 2.6
Glass	33.8 ± 1.8	33.8 ± 1.8	33.8 ± 1.8	31.0 ± 2.3	32.1 ± 2.2	37.0 ± 2.8	33.0 ± 2.3
Iris	5.4 ± 1.6	6.7 ± 1.4	6.4 ± 1.4	5.9 ± 1.2	6.3 ± 1.2	6.3 ± 1.2	6.5 ± 1.2
Pageblocks	4.1 ± 0.4	4.1 ± 0.4	4.1 ± 0.4	4.0 ± 0.3	4.2 ± 0.4	3.9 ± 0.3	4.5 ± 0.3
Satimage	15.5 ± 0.2	15.5 ± 0.2	15.5 ± 0.2	15.4 ± 0.3	15.5 ± 0.3	14.9 ± 0.3	17.0 ± 0.3
Segment	3.2 ± 0.2	3.2 ± 0.2	3.2 ± 0.2	2.9 ± 0.2	2.9 ± 0.3	5.0 ± 0.4	5.2 ± 0.3
USPS	24.2 ± 0.3	24.2 ± 0.3	24.2 ± 0.3	24.0 ± 0.3	24.2 ± 0.3	24.8 ± 0.3	**22.4 ± 0.7**
Vowel	14.5 ± 0.9	14.5 ± 0.9	14.5 ± 0.9	**10.8 ± 0.7**	11.2 ± 0.7	47.9 ± 1.5	21.8 ± 1.0
Waveform	41.2 ± 2.9	41.3 ± 2.9	41.3 ± 2.9	**32.9 ± 3.3**	39.2 ± 3.4	**32.9 ± 3.3**	41.7 ± 4.0
Wine	10.4 ± 1.6	11.3 ± 1.6	11.1 ± 1.6	10.2 ± 1.8	10.8 ± 1.3	11.2 ± 2.0	15.5 ± 1.8
Yeast	40.3 ± 1.9	40.3 ± 1.9	40.3 ± 1.9	**32.7 ± 0.9**	35.8 ± 1.5	47.9 ± 1.0	36.0 ± 1.2

Table 8: Error rates (%), pruned decision trees, sparse code matrix

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Hamming	Passerini	Zadrozny
Ecoli	13.4 ± 1.9	13.4 ± 1.9	13.4 ± 1.9	10.6 ± 1.3	12.9 ± 1.6	11.3 ± 1.3	10.5 ± 1.1
Faces	31.8 ± 2.4	31.8 ± 2.4	31.8 ± 2.4	30.2 ± 2.5	30.9 ± 2.3	36.6 ± 2.4	34.8 ± 2.4
Glass	34.2 ± 2.3	34.2 ± 2.3	34.2 ± 2.3	**29.7 ± 1.9**	32.6 ± 3.4	30.9 ± 1.9	30.1 ± 1.7
Iris	5.0 ± 1.0	5.7 ± 1.0	5.5 ± 1.0	5.6 ± 1.0	5.6 ± 1.0	5.8 ± 1.1	6.2 ± 1.3
Pageblocks	3.8 ± 0.3	3.8 ± 0.3	3.8 ± 0.3	3.9 ± 0.3	4.5 ± 0.7	4.0 ± 0.6	4.0 ± 0.3
Satimage	15.0 ± 0.4	15.0 ± 0.4	15.0 ± 0.4	14.7 ± 0.4	15.0 ± 0.4	14.3 ± 0.3	15.4 ± 0.4
Segment	3.5 ± 0.3	3.5 ± 0.3	3.5 ± 0.3	**2.8 ± 0.3**	3.8 ± 0.4	3.1 ± 0.3	3.6 ± 0.4
USPS	24.0 ± 0.3	24.0 ± 0.3	24.0 ± 0.3	23.6 ± 0.3	25.4 ± 0.3	24.6 ± 0.3	23.6 ± 0.3
Vowel	17.5 ± 1.0	17.5 ± 1.0	17.5 ± 1.0	**13.3 ± 0.8**	14.0 ± 0.9	24.5 ± 1.2	15.4 ± 0.9
Waveform	28.6 ± 1.8	29.0 ± 1.9	28.9 ± 1.9	25.4 ± 1.2	29.1 ± 2.3	**25.1 ± 1.2**	31.2 ± 2.8
Wine	6.9 ± 1.4	7.6 ± 1.5	7.9 ± 1.9	6.8 ± 1.6	7.1 ± 1.6	7.1 ± 1.5	13.2 ± 3.6
Yeast	39.8 ± 1.7	39.8 ± 1.7	39.8 ± 1.7	**34.1 ± 0.8**	40.4 ± 1.6	41.7 ± 1.3	34.3 ± 0.8

Table 9: Error rates (%), pruned decision trees, OVO

dataset	set accuracy	discounted accuracy	maximin	Bradley-Terry	Wu 1	Wu 2	Hamming	Passerini	Zadrozny
Ecoli	13.6 ± 1.7	13.6 ± 1.7	13.6 ± 1.7	13.2 ± 1.6	13.5 ± 1.6	13.7 ± 1.6	13.6 ± 1.5	13.2 ± 1.6	13.2 ± 1.6
Faces	33.9 ± 2.4	34.3 ± 2.3	34.4 ± 2.4	36.7 ± 2.2	36.4 ± 2.6	38.1 ± 2.6	37.4 ± 2.6	38.3 ± 2.4	36.8 ± 2.3
Glass	29.3 ± 2.8	37.1 ± 3.1	35.7 ± 3.7	34.5 ± 3.4	34.9 ± 3.3	34.2 ± 3.7	35.5 ± 3.8	34.1 ± 3.5	34.5 ± 3.4
Iris	3.4 ± 0.9	6.1 ± 1.0	5.8 ± 1.2	6.1 ± 1.4	6.1 ± 1.4	5.8 ± 1.2	6.0 ± 1.3	5.8 ± 1.2	6.1 ± 1.4
Pageblocks	4.1 ± 0.3	4.1 ± 0.3	4.1 ± 0.3	3.9 ± 0.2	3.9 ± 0.2	3.9 ± 0.2	4.0 ± 0.3	3.9 ± 0.2	3.9 ± 0.2
Satimage	16.3 ± 0.5	16.3 ± 0.5	16.3 ± 0.5	15.7 ± 0.4	15.7 ± 0.4	15.7 ± 0.4	16.1 ± 0.5	15.8 ± 0.4	15.7 ± 0.4
Segment	4.8 ± 0.4	4.8 ± 0.4	4.8 ± 0.4	4.4 ± 0.4	4.4 ± 0.4	4.4 ± 0.4	4.5 ± 0.4	4.4 ± 0.4	4.4 ± 0.4
USPS	25.9 ± 0.3	25.9 ± 0.3	25.9 ± 0.3	25.7 ± 0.3	25.9 ± 0.3	26.0 ± 0.3	26.3 ± 0.3	26.3 ± 0.3	25.7 ± 0.3
Vowel	29.1 ± 1.2	29.2 ± 1.1	29.2 ± 1.1	23.6 ± 1.2	**23.4 ± 1.1**	24.1 ± 1.1	23.8 ± 1.1	23.7 ± 1.2	23.7 ± 1.3
Waveform	30.4 ± 1.2	30.5 ± 1.2	30.5 ± 1.2	28.8 ± 1.3	28.7 ± 1.3	28.6 ± 1.4	29.5 ± 1.5	28.7 ± 1.3	28.8 ± 1.3
Wine	7.0 ± 1.7	9.7 ± 1.5	8.0 ± 1.7	8.1 ± 1.6	7.7 ± 1.5	7.9 ± 1.5	8.1 ± 1.7	8.0 ± 1.6	7.7 ± 1.5
Yeast	45.8 ± 1.9	45.8 ± 2.0	45.8 ± 1.9	40.8 ± 1.8	40.6 ± 1.8	40.8 ± 2.2	42.7 ± 2.4	40.5 ± 1.8	40.8 ± 1.8

Table 10: Error rates (%), imprecisely classified instances, unpruned decision trees, OVA

dataset	nb. impr. instances	nb. retained classes	set accuracy	discounted accuracy	maximin	Bradley-Terry	Hamming	Passerini	Zadrozny
Ecoli	18.3	2.5	6.1	60.5	30.2	45.1	37.6	45.1	42.0
Faces	6.5	2.1	29.0	66.3	61.7	85.2	64.8	85.2	75.9
Glass	20.7	2.5	12.0	62.8	46.6	65.2	53.6	65.2	64.0
Iris	2.8	2.0	0.0	50.0	39.4	42.3	42.3	42.3	40.8
Pageblocks	57.0	2.2	3.4	54.6	40.4	53.5	44.9	53.5	50.2
Satimage	126.5	2.4	9.7	59.7	50.3	73.0	59.0	73.0	66.6
Segment	12.6	2.1	4.1	53.5	24.1	42.9	32.7	42.9	35.9
USPS	290.7	2.8	67.3	86.6	81.9	91.2	86.7	91.2	89.3
Vowel	62.1	2.8	14.3	65.2	38.5	56.2	47.9	56.2	52.3
Waveform	107.4	2.0	0.0	50.3	48.3	74.5	54.7	74.5	67.7
Wine	1.9	2.0	0.0	50.0	29.2	45.8	25.0	45.8	39.6
Yeast	212.8	3.0	10.3	66.3	40.6	53.9	47.9	53.9	50.6