Mach Learn (2008) 73: 133–153
DOI 10.1007/s10994-008-5064-8

Multilabel classification via calibrated label ranking

Johannes Fürnkranz · Eyke Hüllermeier · Eneldo Loza Mencía · Klaus Brinker

Received: 1 February 2007 / Revised: 21 April 2008 / Accepted: 12 June 2008 /
Published online: 6 August 2008

Abstract Label ranking studies the problem of learning a mapping from instances to rankings over a predefined set of labels. Hitherto existing approaches to label ranking implicitly operate on an underlying (utility) scale which is not calibrated in the sense that it lacks a natural zero point. We propose a suitable extension of label ranking that incorporates the calibrated scenario and substantially extends the expressive power of these approaches. In particular, our extension suggests a conceptually novel technique for extending the common learning by pairwise comparison approach to the multilabel scenario, a setting previously not being amenable to the pairwise decomposition technique. The key idea of the approach is to introduce an artificial calibration label that, in each example, separates the relevant from the irrelevant labels. We show that this technique can be viewed as a combination of pairwise preference learning and the conventional relevance classification technique, where a separate classifier is trained to predict whether a label is relevant or not. Empirical results in the area of text categorization, image classification and gene analysis underscore the merits of the calibrated model in comparison to state-of-the-art multilabel learning methods.

Keywords Multi-label classification · Preference learning · Ranking

Editor: Tom Fawcett.

J. Fürnkranz (✉) · E. Loza Mencía
TU Darmstadt, Darmstadt, Germany
e-mail: juffi@ke.informatik.tu-darmstadt.de

E. Loza Mencía
e-mail: eneldo@ke.informatik.tu-darmstadt.de

E. Hüllermeier · K. Brinker
Philipps-Universität Marburg, Marburg, Germany

E. Hüllermeier
e-mail: eyke@informatik.uni-marburg.de

K. Brinker
e-mail: brinker@informatik.uni-marburg.de

1 Introduction

Machine learning problems with structured input spaces have recently received increasing attention, as a variety of important application fields require methods to encode and exploit data which is organized in a complex way; classification of biomolecular structures may serve as an interesting example (e.g. Weskamp et al. 2007). Such input spaces cannot be mapped onto "flat" feature vectors of a fixed length without an inherent loss of essential information. The development of suitable kernels for structured data is an active research topic in the field of kernel-based learning (Gärtner 2003).

Structured spaces cannot only be found for input data but also arise naturally from *output data*, such as in predicting label sequences or natural language parsing trees (Tsochantaridis et al. 2004; Altun et al. 2006). In this paper, we will be concerned with structured outputs in the form of *label rankings*. More specifically, the problem we consider is to learn a mapping from instances to rankings (total orders) over a finite number of predefined class labels. Two general approaches to extending arbitrary (linear) binary classification algorithms to the label ranking setting have recently been proposed: *Ranking by pairwise comparison* (RPC) as a natural extension of pairwise classification (Fürnkranz and Hüllermeier 2003; Hüllermeier et al. 2008) and *constraint classification* (CC), which learns a linear utility function for each label (Har-Peled et al. 2002).

Label ranking extends conventional multiclass classification in the sense that it does not only predict a "top candidate" but instead gives an ordering of all class labels. Besides, it is well-known that a number of other learning tasks can be formalized within the setting of label ranking. Interestingly, however, this does not include *multilabel* classification. This is due to a substantial restriction of all hitherto existing label ranking methods, and actually of the concept of a ranking itself: Even though a ranking informs about the *relative* order of all alternatives, it does not provide any information about *absolute* preferences in the sense of distinguishing between "good" and "bad" alternatives. Roughly speaking, this is due to the fact that a ranking does not have a natural "zero-point". In document categorization, for example, a label ranking method is able to predict a ranking of all topics in decreasing order of relevance to a specific document. However, it is not possible to distinguish between the sets of (presumably) relevant and non-relevant topics.

In this paper, we present a simple yet elegant and effective technique to avoid the aforementioned disadvantage. The key idea is to add an additional label to the original label set which is interpreted as a "neutral element". This label calibrates a ranking by splitting it into a positive and a negative part. By extending conventional label ranking approaches, this novel framework provides a means to represent and learn bipartite partitions of alternatives and, thereby, combines multiclass classification and label ranking. Our technique is not tailored or limited to a specific label ranking method. Instead, it can be applied to all methods that make use of pairwise preferences between class labels as training information. In particular, it suggests a conceptually new technique for extending the common pairwise classification learning approach to the multilabel scenario, a setting previously not amenable to a pairwise decomposition technique.

The remainder of this paper is organized as follows: We start with a brief recapitulation of the problems of learning label rankings (Sect. 2) and multilabel classification and ranking (Sect. 3). In particular, we will discuss how ranking problems can be addressed with pairwise classifiers. In Sect. 4, we then introduce our technique that allows to generalize this approach to multilabel classification problems by introducing an artificial label whose position in the predicted ranking can serve as a calibration point. We will also show that this approach effectively combines binary relevance ranking with pairwise label ranking, and discuss its

computational complexity. Section 5 then describes the setup of our empirical evaluation of the calibrated approach to multilabel classification and ranking. The results on four real-world multilabel classification tasks (Sect. 6) show that it compares favorably to alternative approaches that are based on the binary relevance technique. Finally, we summarize related work in Sect. 7 and draw our conclusions in Sect. 8.

2 Label ranking

In label ranking, the problem is to learn a mapping from instances x from an instance space $\mathcal{X}$ to rankings $\succ_x$ (total strict orders) over a finite set of labels $\mathcal{L} = \{\lambda_1, \ldots, \lambda_c\}$, where $\lambda_i \succ_x \lambda_j$ means that, for instance x, label λ_i is preferred to λ_j. A ranking over $\mathcal{L}$ can be represented by a permutation as there exists a unique permutation τ such that $\lambda_i \succ_x \lambda_j$ iff $\tau(\lambda_i) < \tau(\lambda_j)$, where $\tau(\lambda_i)$ denotes the position of the label λ_i in the ranking.[1] We shall denote by $\tau^{-1}(i)$ the label λ having assigned position i. The target space of all permutations over c labels will be referred to as $\mathcal{S}_c$.

It has been pointed out in several publications (Har-Peled et al. 2002; Fürnkranz and Hüllermeier 2003; Dekel et al. 2004) that a variety of learning problems may be viewed as special cases of label ranking (perhaps supplemented by a straightforward projection of the output space $\mathcal{S}_c$) hence underscoring the importance of this setting. Among those are the following:

- *Multiclass classification:* A single class label λ_i is assigned to each example x. This implicitly defines the set of preferences $R_x = \{\lambda_i \succ_x \lambda_j \mid 1 \leq j \neq i \leq c\}$. The output space $\mathcal{S}_c$ is projected to the first component.
- *l-Multilabel classification:* Each training example x is associated with a subset $P_x \subseteq \mathcal{L}$ of possible labels. This implicitly defines the set of preferences $R_x = \{\lambda_i \succ_x \lambda_j \mid \lambda_i \in P_x, \lambda_j \in \mathcal{L} \setminus P_x\}$. The output space is projected to the first l components.

Ranking by pairwise comparison (RPC) and constraint classification (CC) both provide a general means to extend arbitrary (linear) binary classification algorithms to the label ranking scenario. Both approaches require (not necessarily complete) sets of pairwise preferences associated with the training instances to learn a ranking model which, as a post-processing step, may be projected from $\mathcal{S}_c$ to the specific output space $\mathcal{Y}$.

The key idea of RPC is to learn, for each pair of labels (λ_i, λ_j), a binary model $\mathcal{M}_{ij}(x)$ that predicts whether $\lambda_i \succ_x \lambda_j$ or $\lambda_j \succ_x \lambda_i$ for an input x. In order to rank the labels for a new instance, predictions for all pairwise label preferences are obtained and a ranking that is maximally consistent with these preferences is derived, typically by means of a voting scheme.[2] This technique describes a natural extension of pairwise classification, i.e., the idea to approach a multiclass classification problem by learning a separate model for each pair of classes.

A natural alternative option for modeling preference rankings is to represent each individual label by means of an associated (real-valued) utility function. More precisely, a utility function $f_i : \mathcal{X} \to \mathbb{R}$ is learned for each of the labels λ_i, $i = 1, \ldots, c$, where $f_i(x)$ is the utility assigned to λ_i by the instance x. To obtain a ranking for x, the labels are ordered according to these utility scores, i.e., $\lambda_i \succeq_x \lambda_j \Leftrightarrow f_i(x) \geq f_j(x)$. The challenge for the

[1] Note that we slightly depart from standard notation as permutations τ are defined on labels λ_i rather than their indices i to simplify the technical exposition.

[2] To account for equal vote statistics, we consider random tie breaking in our implementation.

learner is to find functions $f_1, f_2, \ldots, f_c$ that are as much as possible in agreement with all constraints. A corresponding method for learning *linear* utility functions $f_i(\cdot)$, $i = 1, \ldots, c$, from training data has been proposed in the framework of constraint classification (Har-Peled et al. 2002). Constraint classification encodes each constraint induced by comparative preference information on labels as two binary training examples. The set of utility functions can be learned by solving the overall inflated binary classification problem by means of any binary learning algorithm.

Unfortunately, certain categories of learning problems do not admit an embedding as a special case of the label ranking setting and, hence, are not amenable to pairwise ranking and constraint classification. In particular, conventional multilabel classification and ranking are of substantial practical relevance and have received considerable attention in text categorization (Schapire and Singer 2000) and genome analysis research (Elisseeff and Weston 2002), among others. The underlying reason for this severe restriction is that previous approaches to label ranking implicitly operate on non-calibrated scales. More precisely, these approaches learn preference models which assign importance scores, i.e., aggregated preference votes or utility values, to each possible label. However, the training process is performed in a manner such that absolute importance scores do not provide a reasonable basis for defining a threshold-based bipartite partition of labels. As a consequence, it is not possible to determine the set of relevant labels in multilabel classification for example.

3 Multilabel classification and ranking

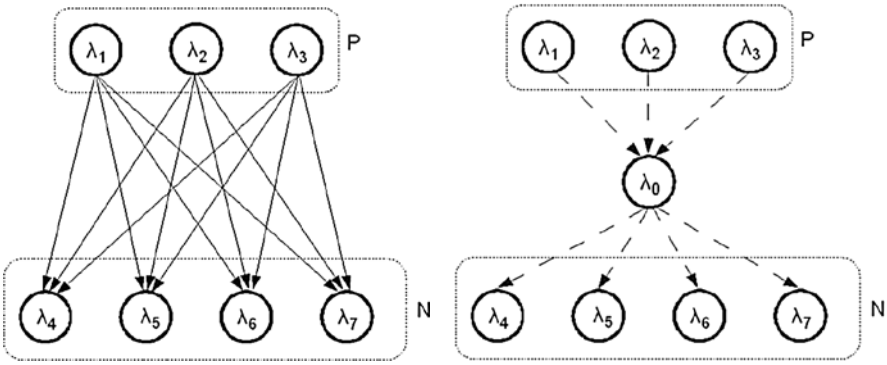
Multilabel classification refers to the task of learning a function that maps instances $x \in \mathcal{X}$ to label subsets $P_x \subset \mathcal{L}$, where $\mathcal{L} = \{\lambda_1, \ldots, \lambda_c\}$ is a finite set of predefined labels, typically with a small to moderate number of alternatives. Thus, in contrast to multiclass learning, alternatives are not assumed to be mutually exclusive such that multiple labels may be associated with a single instance. The set of labels P_x are called *relevant* for the given instance, the set $N_x = \mathcal{L} \setminus P_x$ are the *irrelevant* labels.

A common approach to multilabel classification is *binary relevance learning* (BR). BR trains a separate binary relevance model $\mathcal{M}_i$ for each possible label λ_i, using all examples x with $\lambda_i \in P_x$ as positive examples and all those with $\lambda_i \in N_x$ as negative examples. For classifying a new instance, all binary predictions are obtained and then the set of labels corresponding to positive relevance classification is associated with the instance. This scenario is, for example, commonly used for evaluating algorithms on the REUTERS text classification benchmark (Lewis 1997).

In the following, we will study the task of *multilabel ranking*, which is understood as learning a model that associates with a query input x both a ranking of the complete label set $\{\lambda_1, \ldots, \lambda_c\}$ and a bipartite partition of this set into relevant and irrelevant labels. Thus, multilabel ranking can be considered as a generalization of both multilabel classification and ranking.

In conventional label ranking, a training example typically consists of an instance $x \in \mathcal{X}$, represented with a fixed set of features, and a set of pairwise preferences over labels $R_x \subset \mathcal{L}^2$, where $(\lambda, \lambda') \in R_x$ is interpreted as $\lambda \succ_x \lambda'$. In multilabel classification, the training information consists of a set P_x of relevant labels and, implicitly, a set $N_x = \mathcal{L} \setminus P_x$ of irrelevant labels. Note that this information can be automatically transformed into a set of preferences $R_x = \{(\lambda, \lambda') \mid \lambda \in P_x \wedge \lambda' \in N_x\}$ (cf. Fig. 1(a)).

While it is straightforward to represent the training information for multilabel classification as a preference learning problem, the algorithms that operate in this framework merely



(a) the set of preferences representing a multilabel classification problem

(b) introducing a calibration label λ_0 that separates P and N

(c) the set of preferences representing a calibrated label ranking problem

Fig. 1 Calibrated label ranking

produce a ranking of the available options. In order to convert the learned ranking into a multilabel prediction, the learner has to be able to autonomously determine a point at which the learned ranking is split into sets of relevant and irrelevant labels. Previous applications of ranking techniques to multilabel learning, such as (Crammer and Singer 2003), have mostly ignored this problem and restricted their focus on producing rankings, but not on determining this correct *zero point* for splitting the ranking. A notable exception is the approach taken by Elisseeff and Weston (2002), who frame the problem as a meta-learning problem, where the task is to learn a predictor for the correct threshold in the ranking.

Multilabel ranking can, for example, be realized if the binary classifiers of the binary relevance approach provide real-valued confidence scores or *a posteriori* probability estimates for classification outcomes. For example, Schapire and Singer (2000) included an *ad hoc* extension to multilabel ranking in their experimental setup by ordering labels according to decreasing confidence scores.

4 Calibrated label ranking

In this section, we will introduce *calibrated ranking by pairwise comparison (CRPC)*, a conceptually new technique for extending the common pairwise learning approach to the multilabel scenario, a setting previously not being amenable to a pairwise decomposition approach. Within our framework, RPC can solve both multilabel classification and ranking problems in a consistent and generally applicable manner. We will first describe the technique in detail (Sect. 4.1) and then discuss its relation to binary relevance learning (Sect. 4.2) and its computational complexity (Sect. 4.3).

4.1 Calibrated ranking by pairwise comparisons

We start with a formal definition of the hypothesis space underlying the *calibrated label ranking framework*:

Definition 4.1 (Calibrated label ranking model) Denote by $\mathcal{X}$ a nonempty input space and by $\mathcal{S}_c^0$ the space of permutations over the set $\{\lambda_0, \lambda_1, \ldots, \lambda_c\}$, that is, the original set of labels plus an additional *calibration label* λ_0. Then, a model $h : \mathcal{X} \to \mathcal{S}_c^0$ is referred to as a *calibrated label ranking model*.

The key idea is to interpret the calibration label λ_0 as a split point between relevant and irrelevant labels: all relevant labels are preferred to λ_0, which in turn is preferred to all irrelevant labels. Thus, a *calibrated ranking*

$$\lambda_{i_1} \succ \cdots \succ \lambda_{i_j} \succ \lambda_0 \succ \lambda_{i_{j+1}} \succ \cdots \succ \lambda_{i_c} \tag{1}$$

induces both a ranking among the labels,

$$\lambda_{i_1} \succ \cdots \succ \lambda_{i_j} \succ \lambda_{i_{j+1}} \succ \cdots \succ \lambda_{i_c}, \tag{2}$$

and a bipartite partition into

$$P = \{\lambda_{i_1}, \ldots, \lambda_{i_j}\} \quad \text{and} \quad N = \{\lambda_{i_{j+1}}, \ldots, \lambda_{i_c}\} \tag{3}$$

in a straightforward way.

As sketched in the previous section, the training information for a multilabel ranking problem consists of a set of preferences R_x, and subsets of labels $P_x, N_x \subset \mathcal{L}$ with $P_x \cap N_x = \emptyset$, which distinguish, respectively, *positive* labels that should be ranked above the zero-point element λ_0 and *negative* labels to be ranked below.[3] The bipartite partitions associated with the training instances, in conjunction with the calibration label λ_0, are used to induce additional constraints: the calibrated classifier h should predict $\lambda \succ_x \lambda_0$ for all $\lambda \in P_x$ and vice-versa $\lambda_0 \succ_x \lambda'$ for all $\lambda' \in N_x$ (cf. Fig. 1(b)). Moreover, as a consequence of transitivity, it should predict $\lambda \succ_x \lambda'$ for all $\lambda \in P_x$ and $\lambda' \in N_x$ (cf. Fig. 1(c)). Combining the

[3] In general, we do not need to assume complete training data, neither for the sets of preferences (R_x might even be empty) nor for the partitions (which do not necessarily have to cover all the labels, i.e., $P_x \cup N_x \neq \mathcal{L}$). Besides, in a *noisy learning scenario*, it may happen that $(\lambda', \lambda) \in R_x$ even though $\lambda \in P_x$ and $\lambda' \in N_x$. In this paper, we will not further consider these cases, and assume a strict multilabel scenario.

new partition-induced preference constraints with the original set of pairwise preferences for the training data, i.e.,

$$R'_x \stackrel{\text{def}}{=} R_x \cup \{(\lambda, \lambda_0) \mid \lambda \in P_x\} \cup \{(\lambda_0, \lambda') \mid \lambda' \in N_x\}, \tag{4}$$

the calibrated ranking model becomes amenable to existing label ranking methods: A model can be learned by solving a conventional ranking problem in the augmented calibrated hypothesis space, that is, a ranking problem with $c + 1$ alternatives. The training data for this problem is given by the modified sets of constraints R'_x on the original labels $\lambda_1, \ldots, \lambda_c$ *and* the calibration label λ_0.

This way, it becomes possible to incorporate and exploit partition-related preference information and to generalize to settings where predicting a zero-point is required. As the above derivation has shown, the unified approach is rather general in the sense of being applicable to all label ranking techniques that accept pairwise preferences between class labels as training information. In particular, this includes RPC and constraint classification (Brinker and Hüllermeier 2005).

4.2 Relation to binary relevance learning

Conventional pairwise label ranking learns a binary preference model $\mathcal{M}_{ij}$ for all combinations of labels λ_i and λ_j with $1 \leq i < j \leq c$,[4] where instances x with $(\lambda_i, \lambda_j) \in R_x$ are associated with positive and those with $(\lambda_j, \lambda_i) \in R_x$ with negative class labels (cf. Fig. 1 (b)). In the calibrated scenario, the partition-related constraints with respect to λ_0 as defined in (4) are required to learn an additional set of binary preference models $\mathcal{M}_{0j}$ with $1 \leq j \leq c$. It is important to notice, that these additional models are identical to the common binary-relevance models $\mathcal{M}_j$.

Theorem 4.2 *The models $\mathcal{M}_{0j}$ that are learned by a pairwise approach to calibrated ranking, and the models $\mathcal{M}_j$ that are learned by conventional binary relevance ranking, are equivalent.*

Proof Each training example x, for which label λ_j is relevant, is, by definition, a positive example in the training set for model $\mathcal{M}_j$. The calibrated ranking approach adds the preference $\lambda_j \succ_x \lambda_0$, which means that x will be a negative example for $\mathcal{M}_{0j}$. Similarly, if λ_j is irrelevant, x is negative for $\mathcal{M}_j$ and positive for $\mathcal{M}_{0j}$. Assuming a symmetric learner, the learned models will be equivalent in the sense that $\mathcal{M}_j = -\mathcal{M}_{0j}$. □

Thus, calibrated RPC may be viewed as a method for combining RPC with conventional binary relevance ranking, in the sense that the binary models that are learned for RPC, and the binary models that are learned for BR, are pooled into a single ensemble. However, CRPC provides a new interpretation to the BR models, that not only allows for ranking the labels, but also to determine a suitable split into relevant and irrelevant categories.

By training a larger number of pairwise models, the calibrated extension of RPC achieves two potential advantages over simple relevance learning. Firstly, it provides additional information about the ranking of labels. Secondly, it may also improve the discrimination

[4]The case $i > j$ is typically not required as a consequence of the symmetry of binary classifiers with respect to positive and negative instances.

between relevant and irrelevant labels. In fact, it is legitimate to assume (and indeed supported by empirical evidence in Sect. 6) that the additional binary models can somehow "stabilize" the related classification. For example, while an error of model $\mathcal{M}_j = -\mathcal{M}_{0j}$ definitely causes a misclassification of label λ_j in simple relevance learning, this error might be compensated by the models $\mathcal{M}_{ij}$, $1 \leq i \neq j \leq c$, in the case of RPC. The price to pay is, of course, a higher computational complexity, which, as we will show in the next section, depends on the average number of relevant labels for an example.

4.3 Computational complexity

In this section, we will derive bounds for the number of training examples that are constructed for training the pairwise classifiers. The total computational complexity depends on the complexity of the base classifier used for processing these examples. In brief, superlinear classifiers like SVMs will profit from distributing the work-load on many small problems instead of fewer larger problems as for the BR approach.

Theorem 4.3 *CRPC is trained on $O(lc)$ examples, where $l = \sum_x |P_x|$ is the total number of all relevant labels in the training set, and c is the number of possible labels.*

Proof In previous work, it has been shown that training a pairwise classifier requires $O(cn)$ training examples (Fürnkranz 2002). To see this, note that each of the n original training examples will only appear in the training sets of $c-1$ models, therefore the total number of training examples that have to be processed is $(c-1)n$.

For multilabel classification, RPC will compare a training example's $|P_x|$ relevant labels to all $|N_x| = c - |P_x|$ labels that are not relevant for this example. In addition, CRPC will include every example in the c training sets of the models $\mathcal{M}_{0j}$, $j = 1, \ldots, c$.

Thus, each example occurs in

$$|P_x| \cdot |N_x| + c = |P_x| \cdot (c - |P_x|) + c \leq |P_x| \cdot c + c = c \cdot (|P_x| + 1)$$

training sets. The total number of training examples in all training sets is therefore

$$c \cdot \sum_x (|P_x| + 1)) = c \cdot (l + n) \leq c \cdot 2l$$

assuming $n \leq l$, i.e., that each example has at least one label. Thus, the total number of training examples is $O(lc)$. □

This result shows that the complexity of training CRPC depends critically on the total number of relevant labels in the training examples. For the case of $l = n$, i.e., for conventional pairwise classification, the derived bound reduces to the linear $O(nc)$ bound that was shown in (Fürnkranz 2002).[5] Multilabel classification increases this bound by a factor l/n to $O(cn \cdot l/n) = O(cl)$, i.e., the complexity of CRPC is within a factor of l/n of the $O(cn)$ examples needed for training a BR classifier.

Thus, the crucial factor that determines the complexity of the approach is l/n, the average number of labels per example. We would like to point out that in many practical applications,

[5] Note that the O-notation hides, in this case, a constant factor of two, which results from the introduction of the calibration label λ_0.

this factor is determined by the procedure that is used for labeling the training examples, and is independent of c. A typical example is the number of keywords that are assigned to a text, which typically is a low number and does not depend on the number of available keywords. For example, in the *Reuters-RCV1* text categorization dataset, the median value of keywords is three, and the ratio of documents with more than ten keywords is approximately one per thousand (cf. also Fig. 2 later in this paper).

The above results that show that the effort is linear in the total number of labels in the training set only apply to the training time. We still have to store a quadratic number of classifiers, and, in principle, all of them have to be queried at classification time. Park and Fürnkranz (2007) have recently proposed an efficient algorithm that allows to compute the top-ranked class for prediction in essentially linear time. We are currently working on an adaptation of this algorithm for multilabel classification, which will compute the ranking from top to bottom, and stop as soon as the calibration label has been found. However, even with this method, we still have to store all pairwise classifiers, which may become a practical burden for large number of classes. For some types of learning algorithms (like decision trees or rule learners), we think that the fact that the pairwise models tend to be simpler than the one-against-all models balances this to some extent, but other types of classifiers, such as perceptrons or SVMs, have to store a constant number of parameters for both types of problems, and thus, for such base classifiers, the memory demand of the pairwise method is quadratic in the number of classes. This problem is next on our research agenda.

5 Experimental setup

The purpose of this section is to provide an empirical evaluation of calibrated RPC. Our goal is to show that the calibration procedure adds good multilabel classification abilities to the already strong ranking performance of pairwise ranking algorithms. Thus, we have to establish two results: on the one hand, we want to demonstrate that the calibration procedure proposed above yields a good selection of relevant labels, i.e., that it outperforms the binary relevance approach. On the other hand, we intend to show that it does not lose in ranking performance compared to conventional pairwise ranking, which in turn outperforms ranking approaches that are based on a one-against all decomposition of the problem.

5.1 Learning algorithms

As a base classifier, we selected linear perceptrons, for which Crammer and Singer (2003) have shown that they are an efficient alternative to linear support-vector machines on large-scale learning multilabel ranking problems such as the *Reuters-RCV1* collection. More precisely, their *multiclass multilabel perceptrons (MMP)* train one perceptron for each possible label, but, unlike the conventional binary-relevance scenario, these classifiers are not trained independently, but in a way such that they collectively produce a reasonable ranking for a given ranking loss function. We will follow the recommendations of Crammer and Singer (2003) and train the perceptrons by optimizing the IsError-loss function (the 0/1-loss indicating whether the ranking is correct or not) with uniform updates (where all misranked pairs receive an equal weight). We have also tried to optimize other ranking losses (e.g., to directly optimize RankingLoss), but the results were qualitatively the same, and we omit them here for clarity. Crammer and Singer (2003) have shown that MMPs outperform conventional *binary relevance ranking (BR)*.

Subsequently, Loza Mencía and Fürnkranz (2008) have introduced *pairwise multilabel perceptrons (MLPC)*. MLPC train one perceptron for each pair of classes, each perceptron

is trained independently of all others. While this approach is somewhat less efficient than MMPs, Loza Mencía and Fürnkranz (2008) have shown that tackling large datasets such as *Reuters-RCV1* is still feasible with the pairwise approach, and results in a gain of accuracy.

However, both MMPs and MLPCs are only able to produce a ranking of all possible labels. In the following, we will compare all of the above options (BR, MMP, MLPC) to *calibrated multilabel perceptrons (CMLPC)*, a calibrated variant of MLPC. Calibrated MLPCs employ the calibration technique introduced in this paper and use MLPC for learning the pairwise models for the new, enlarged label set.

Even though we only report experiments with a single classifier in this paper, we are convinced that the qualitative results are typical, independent of the chosen base classifier. Pairwise methods have previously been shown to improve performance for neural networks (Knerr et al. 1990, 1992; Price et al. 1995; Lu and Ito 1999), support vector machines (Schmidt and Gish 1996; Hastie and Tibshirani 1998; Kreßel 1999; Hsu and Lin 2002), statistical learning (Bradley and Terry 1952; Friedman 1996), rule and decision tree learning (Fürnkranz 2002, 2003) and others. Moreover, for two of the datasets used in this study, we have also experimented with calibrated pairwise ranking using support vector machines as base classifiers, and obtained similar results (Brinker et al. 2006). However, in these experiments we had to restrict ourselves to a small subset of the REUTERS dataset, so we decided to use the more efficient perceptron classifiers for the more extensive set of experiments reported in this paper.

5.2 Datasets

The datasets that were included in the experimental setup cover three application areas in which multilabeled data are frequently observed: *text categorization* (the *Reuters-RCV1* and *Reuters-21578* datasets), *image classification* (the *scene* dataset) and *bioinformatics* (the *yeast* dataset).[6]

The *Reuters Corpus Volume I (Reuters-RCV1)* is one of the most widely used test collection for text categorization research. It contains 804,414 newswire documents, which we split into 535,987 training documents (all documents before and including April 26th, 1999) and 268,427 test documents (all documents after April 26th, 1999). We used the token files of Lewis et al. (2004), which are already word-stemmed and stop word reduced. However we repeated the stop word reduction as we experienced that there were still a few occurrences. The 25,000 most frequent features on the training set were selected and weighted with TF-IDF weights (Salton and Buckley 1988). We did not restrict the set of 103 categories although one class does not contain any examples in the training set.

We also experimented with the older *Reuters-21578* corpus (Lewis 1997), which has 11,367 examples and 120 possible labels. Through similar pre-processing as in the *Reuters-RCV1* dataset, we obtained 10,000 features for this dataset.

The learning task in the *Yeast* gene functional multiclass classification problem is to associate genes with a subset of 14 functional classes from the Comprehensive Yeast Genome Database of the Munich Information Center for Protein Sequences.[7] Each of 2417 genes is represented with 103 features. In previous experiments (Loza Mencía and Fürnkranz 2008), we found that even the pairwise problems are hard to separate with a linear classifier (much

[6]The *Reuters-RCV1* dataset is available from http://trec.nist.gov/data/reuters/reuters.html, the *Reuters-21578* dataset from http://www.daviddlewis.com/resources/testcollections/reuters21578/, and the *yeast* and *scene* datasets from http://mlkd.csd.auth.gr/multilabel.html.

[7]http://mips.gsf.de/genre/proj/yeast/.

more so in the binary relevance setting). Thus, in this set of experiments, we added all pairwise feature products to the original feature representation, in order to simulate a quadratic kernel function.

The task in the *Scene* dataset (Boutell et al. 2004) is to recognize which of six possible scenes (*beach, sunset, field, fall foliage, mountain, urban*) can be found in 2407 pictures. Many pictures contain more than one scene. For each image, spatial color moments are used as features. Each picture is divided into 49 blocks using a 7×7 grid. A picture is then represented using the mean and the variance of each color band of each block, i.e., using a total of $2 \times 3 \times 7 \times 7 = 294$ features. Like in the *Yeast* dataset, we enriched the feature set with all pairwise feature products.

All algorithms are trained incrementally. For the *Reuters-RCV1* datasets, a single, chronological pass through the data was used (one epoch) because our previous results have shown that multiple iterations are not necessary (Loza Mencía and Fürnkranz 2008). On the other datasets, we trained for multiple epochs. We report the results for training after 100 epochs. However, in terms of the relative order of the tested methods, we found that the results are quite insensitive to the exact numbers of epochs.

Except for the large Reuters RCV1 data, where we used the dedicated test set, all reported results are estimated from 10-fold cross-validation.

5.3 Evaluation measures

There is no generally accepted procedure for evaluating multilabel classifications. Our approach is to consider a multilabel classification problem as a meta-classification problem where the task is to separate the set of possible labels into relevant labels and irrelevant labels. Let $\hat{P}_x$ denote the set of labels predicted by the multilabel classifier and $\hat{N}_x = \mathcal{L} - \hat{P}_x$ the set of labels that are not predicted by the classifier. Thus, we can, for each individual instance x, compute a two-by-two confusion matrix C_x of relevant/irrelevant vs. predicted/not predicted labels:

C_x	predicted	not predicted	
relevant	$\lvert P_x \cap \hat{P}_x \rvert$	$\lvert P_x \cap \hat{N}_x \rvert$	$\lvert P_x \rvert$
irrelevant	$\lvert N_x \cap \hat{P}_x \rvert$	$\lvert N_x \cap \hat{N}_x \rvert$	$\lvert N_x \rvert$
	$\lvert \hat{P}_x \rvert$	$\lvert \hat{N}_x \rvert$	$\lvert \mathcal{L} \rvert$

From such a confusion matrix C_x, we can compute several well-known measures:

- The *Hamming loss* (HAMLOSS) computes the percentage of labels that are misclassified, i.e., relevant labels that are not predicted or irrelevant labels that are predicted. This basically corresponds to the error in the confusion matrix. In order to be consistent with the following measures, we report 1− the Hamming loss, which corresponds to the accuracy on the predicted labels.

$$\text{HAMLOSS}(C_x) \stackrel{\text{def}}{=} 1 - \frac{1}{|\mathcal{L}|} \left| \hat{P}_x \triangle P_x \right|. \tag{5}$$

- *Precision* (PREC) computes the percentage of predicted labels that are relevant, *recall* (REC) computes the percentage of relevant labels that are predicted, and the *F1*-measure

is the harmonic mean between the two.

$$\textsc{Prec}(C_x) \stackrel{\text{def}}{=} \frac{|\hat{P}_x \cap P_x|}{|\hat{P}_x|}, \tag{6}$$

$$\textsc{Rec}(C_x) \stackrel{\text{def}}{=} \frac{|\hat{P}_x \cap P_x|}{|P_x|}, \tag{7}$$

$$\text{F1}(C_x) \stackrel{\text{def}}{=} \frac{2}{\frac{1}{\textsc{Rec}(C_x)} + \frac{1}{\textsc{Prec}(C_x)}} = \frac{2\textsc{Rec}(C_x)\textsc{Prec}(C_x)}{\textsc{Rec}(C_x) + \textsc{Prec}(C_x)}. \tag{8}$$

To average these values, we compute a micro-average over all values in a test set, i.e., we add up the confusion matrices C_x for examples in the test set and compute the measure from the resulting confusion matrix. Thus, for any given measure f, the average is computed as:

$$f_{\text{avg}} = f\left(\sum_{i=1}^{n} C_{x_i}\right). \tag{9}$$

If we use a cross-validation, the measures f_{avg_j}, $j = 1, \ldots, 10$ are then (macro-)averaged over all 10 folds.

Some previous works on multilabel classification, in particular the work on MMPs to which we compare, evaluated the ranking performance and neglected the calibration. For this reason, we also employ three previously used ranking measures (Crammer and Singer 2003). Using these measures allows us to compare the ranking performance of our calibrated methods to previous methods that do not use calibration and cannot be evaluated with the above multilabel loss functions.

We use the following notational conventions: For a given instance x, let $\tau(\lambda_i)$ denote the position of λ_i in the predicted ranking (with the calibrating label λ_0 being removed from the ranking) and $\tau^{-1}(i)$ the label λ that is assigned to position i.

- *Average precision* (AVGPREC) computes for each relevant label the percentage of relevant labels among all labels that are ranked before it, and averages these percentages over all relevant labels.

$$\textsc{AvgPrec}(P_x, \tau) \stackrel{\text{def}}{=} \frac{1}{|P_x|} \sum_{\lambda \in P_x} \frac{|\{\lambda' \in P_x | \tau(\lambda') \leq \tau(\lambda)\}|}{\tau(\lambda)}. \tag{10}$$

- The *ranking loss* (RANKLOSS) computes the average fraction of pairs of labels which are not correctly ordered:

$$\textsc{RankLoss}(P_x, \tau) \stackrel{\text{def}}{=} \frac{|\{(\lambda, \lambda') \in P_x \times N_x : \tau(\lambda) > \tau(\lambda')\}|}{|P_x||N_x|}.$$

- The *one-error loss* (ONEERROR) determines whether the top-ranked label is relevant or not, and ignores the relevancy of all other labels.

$$\textsc{OneError}(P_x, \tau) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } \tau^{-1}(1) \notin P_x, \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

These measures are computed for each example and then averaged over all examples.

6 Results

Reuters-RCV1 with 103 classes and 804,414 examples is the largest and, in our opinion, most interesting test bed for the pairwise approach, because it puts its scalability to test. We will present a detailed analysis for the results for this dataset, and, in Sect. 6.4, show a brief summary of results on the other datasets, which will essentially confirm the results on the *Reuters-RCV1* benchmark.

6.1 Ranking performance

The first three columns of Table 1 show the ranking performance of the three algorithms MMP, its pairwise counter-part MLPC, and its calibrated version CMLPC. The binary relevance classifier BR, a one-against-all ensemble of conventional perceptrons, is clearly outperformed by all other approaches. This is not surprising, as it is the only method that is not concerned with optimizing a ranking. However, the pairwise algorithms also outperform MMPs that improve binary relevance ranking with a training procedure that aims at optimizing their ranking performance. For example, the pairwise classifiers have a $\approx 0.8\%$ ($\approx$2,100 documents) advantage in the percentage of examples for which the top rank is correct (column ONEERROR).

More surprisingly, CMLPC also improves the ranking performance over MLPC. Due to the large number of test examples, these seemingly small improvements are statistically highly significant ($p \ll 0.0001$), as are all other pairwise differences in this table.[8] Apparently, the information provided by the introduction of the artificial calibration label not only allows to split the classes into relevant and irrelevant, but the additional c binary models that are learned by CMLPC also help to somewhat improve the ranking of the other classes. However, large improvements cannot be expected, because the additional preferences involving the calibration label can increase the voting count of each label by at most 1, which allows only minor changes in the ranking positions.

Table 1 Comparison of binary relevance ranking (BR), one-against-all ranking (MMP), pairwise ranking (MLPC) and calibrated pairwise ranking (CMLPC) on the full Reuters-RCV1 dataset

	AVGPREC	RANKLOSS	ONEERROR	HAMLOSS
BR	88.23%	2.529%	5.02%	98.74%
MMP	92.82%	0.687%	3.75%	–
MLPC	93.67%	0.478%	2.96%	–
CMLPC	**93.81**%	**0.472**%	**2.90**%	98.97%

[8] In this case, where we have a ranking for each example in the test case, we used a version of the sign test that is particularly suited for problems with a high number of ties, and seems to be a variant of the widely used McNemar test (McNemar 1947). We chose a sign test because it is simple, it does not make any assumptions about the distribution of the observations, and it is quite conservative (significance with the sign test implies significance with more sensitive tests but not vice versa). The particular version we used computes the test statistic $z = (N_A - N_B)/\sqrt{N_A + N_B}$, where N_A is the number of examples for which algorithm A produces a better ranking than algorithm B and N_B is the number of times for which B ranks better than A (Putter 1955). This statistic is asymptotically distributed like a standard normal distribution. It is based on the idea of distributing the ties randomly between A and B, but this version has a higher asymptotic relative efficiency. It was therefore recommended in a comparative study of several sign tests that allow for the possibility of large numbers of ties (Coakley and Heise 1996).

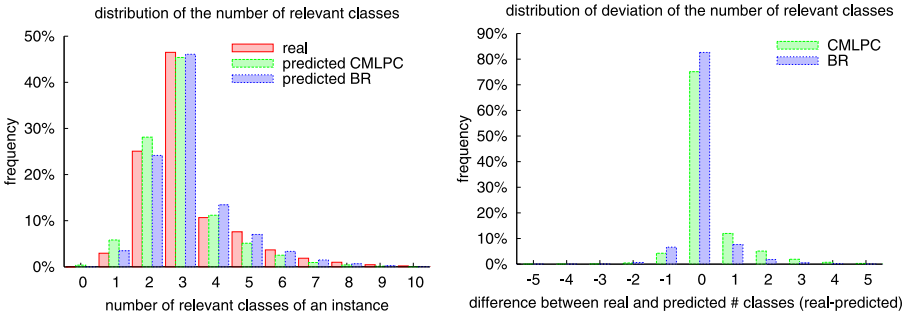


Fig. 2 Predicted and actual number of relevant classes for the calibrated ranking and for the binary relevance approach on the full Reuters-RCV1 dataset

6.2 Calibration performance

The Hamming Loss, shown in the last column of Table 1, can only be computed for the binary relevance classifiers and the calibrated pairwise multilabel perceptrons, because the other approaches are restricted to rank the labels and do not separate them into relevant and irrelevant labels. The results show that CMLPC competently splits between relevant and irrelevant classes: for every prediction there is on average approximately only one class that is placed on the wrong side of the boundary (either via bad ranking or via a bad setting of boundary). For BR, the performance is similar but somewhat worse.

Figure 2 shows the actual distribution of the number of labels for each example, and the distributions that result from the predictions of BR and CMLPC (irrespective of whether the predictions are correct or not). Obviously, both algorithms follow the original distribution quite closely. CMLPC predicts the correct number of classes in about 75% of the cases, in more than 90% the deviation was one class or less. Interestingly, these numbers are even higher for the binary relevance ranking, even though its overall performance is worse because of its bad ranking performance as discussed above. Thus, the calibration point of the CMLPC rankings is closer tied to the ranking performance of the algorithm, while for BR the two appear to be more independent. In general, a small bias towards underestimating the number of labels is noticeable for CMLPC.

In order to get an idea on the quality of the predicted boundary between relevant and irrelevant examples, Table 2 shows a comparison of the boundaries predicted by BR and CMLPC to a fixed boundary of three (i.e., we always predict the median value of three labels), and the real boundary (i.e., we "cheat" by looking at the actual number of relevant labels of the test instances and draw the same boundary in the predicted ranking). As the median and the real boundaries are predetermined, we can also—for these cases—include the ranking algorithms into the comparison.

The results show that CMLPC clearly improves over the median, but it also does not come near the performance using the real boundary. The comparison to MLPC and MMP confirms once more that CMLPC produces the best rankings, with a clear advantage over MMP and BR. The binary relevance ranking BR has, for the predicted boundary, a small advantage in terms of recall, but clearly suffers from a lack in precision. This confirms our conclusions above that CMLPC predicts the boundaries somewhat more cautiously than BR, which suffers from a bad ranking performance, as can again be seen from the results for the median and real boundaries.

Table 2 Average recall, precision and F1 for various boundaries between relevant and irrelevant labels

		PREC	REC	F1
Median (3)	BR	78.86%	73.24%	75.95%
	MMP	81.92%	76.08%	78.89%
	MLPC	82.56%	76.67%	79.51%
	CMLPC	82.74%	76.85%	79.68%
Predicted ($\lvert\hat{P}_x\rvert$)	BR	80.15%	79.70%	79.93%
	CMLPC	86.77%	79.33%	82.88%
Real ($\lvert P_x\rvert$)	BR	81.40%	81.40%	81.40%
	MMP	86.74%	86.74%	86.74%
	MLPC	87.74%	87.74%	87.74%
	CMLPC	87.99%	87.99%	87.99%

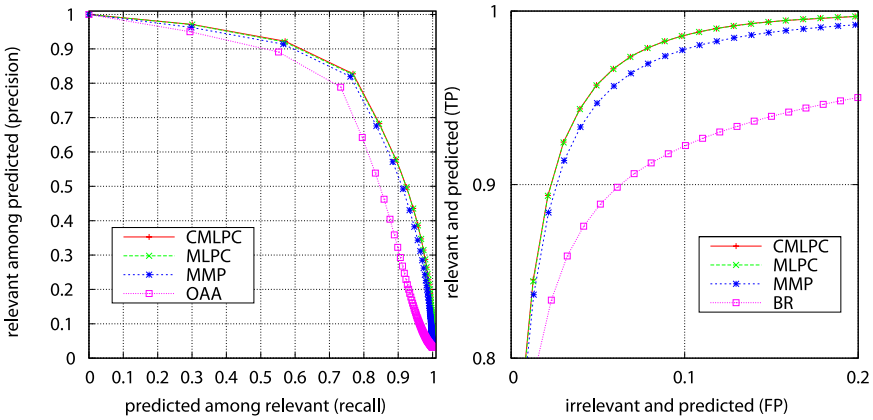


Fig. 3 Micro averaged recall/precision and ROC curves on the full Reuters-RCV1 dataset

Note that the performance that we called "real" in Table 2 is not necessarily the optimal choice in the sense that it is the highest achievable value. Higher F1-values are achievable if we deviate from the original number of labels, because of a suboptimal ranking of the labels. The optimal boundary for each example depends on both the example and the predicted ranking. Also note that choosing always the correct number of relevant classes necessarily results in equal values for precision and recall as the denominators in (7) coincide.

In order to visualize the behavior over all possible boundaries, we also computed recall/precision and ROC curves for each example. This was done by building a label confusion matrix C_x^b for each possible boundary $b = 0, \ldots, 103$ (i.e., on all positions in the ranking). The respective recall/precision and true positive/false positive values were then plotted resulting in a polygon with 103 segments. We again used micro-averaging to average these curves by computing summary matrices

$$C^b = \sum_{i=1}^{n} C_{x_i}^b$$

Table 3 Number of operations of the different algorithms for training and testing

	BR	MMP	MLPC	CMLPC
Training	4,237,467,068	4,307,889,941	13,287,753,360	17,525,186,974
Testing	2,080,697,850	2,080,697,850	106,115,590,350	108,196,288,200

for $b = 0, \ldots, 103$. Recall/precision and ROC curves were plotted using the recall/precision and TP/FP values from these confusion matrices.

The left part of Fig. 3 shows the resulting recall/precision curve. The curve of BR is completely dominated by the MMP curve, which in turn is dominated by the MLPC and the CMLPC curves.

The complete ROC curve exhibits a qualitatively similar behavior. In the right part of Fig. 3, we have enlarged the upper left area [0.0–0.2, 0.8–1.0] to make these small differences more visible. There are also regions in the graph where very small differences between CMLPC and MLPC are noticeable in the graphs, but the resolution of the graphs in this paper is too small to show them. The areas under the curves (AUC) are 0.9747 for BR, 0.9931 for MMP, 0.9952 for MLPC, and 0.9953 for CMLPC. This means that the probability that a randomly selected relevant class is ranked before a randomly selected irrelevant class is almost > 0.995 for the pairwise approaches. For example, for the median case of 3 relevant and 100 irrelevant labels, there are 3×100 possible pairs of relevant and irrelevant classes, and on average 1.5 of them are incorrectly predicted.

6.3 Computational complexity

We measured the run-time in order to compare it to our analysis of the computational complexity in Sect. 4.3. We found it the most convenient to measure an amount of processed operations instead of an amount of (CPU-)time, since in this way it is guaranteed to be independent from external factors such as logging activities and others not part of the basic algorithm, suboptimal routines in the underlying workbench, activities of the operation system etc. Since all the algorithms used on the full Reuters dataset are based on the Perceptron algorithm, a basic operation that is appropriate to compare the results of the different algorithms can easily be found. We defined the basic operation to be an arithmetic floating number operation when calculating a dot product or adding two vectors, the two most frequent operations in the Perceptron algorithm. Other operations such as comparisons and sortings were ignored.

Table 3 shows the number of arithmetic operations in the training and testing process used by the respective algorithms. The ratio of training operations between the binary relevance and the pairwise comparison approach averages 3 and therefore corresponds to the median number of relevant classes per example in the used dataset. These results confirm our analysis in Sect. 4.3. Also note that the complexity of the CMLPC approach equals the sum of the complexities of the MLPC and the BR approaches,[9] in agreement with Theorem 4.2.

6.4 Results on other datasets

Tables 4 and 5 show the ranking loss and the calibration performance of the algorithms on the *Reuters-21578*, *yeast*, and *scene* datasets. Essentially, the results confirm the key findings of the previous sections, namely that

[9]For the training complexities this match is only approximate due to different initialization vectors.

Table 4 Comparison of the ranking performance of binary relevance ranking (BR), one-against-all ranking (MMP), pairwise ranking (MLPC) and calibrated pairwise ranking (CMLPC) on the *Reuters-21578*, *scene*, and *yeast* data. The best result in each column is indicated in bold font, significant differences to CMLPC are indicated with +/- (10%), ++/- - (5%), and +++/- - - (1%)

	AvgPrec	RankLoss	OneError
Reuters-21578			
BR	91.33% - - -	8.0711 - - -	0.0750 - - -
MMP	95.49% - - -	1.1793 - - -	0.0601 -
MLPC	95.50% - - -	0.7189 - - -	0.0618 - - -
CMLPC	**95.84%**	**0.7064**	**0.0560**
scene			
BR	85.64% - -	0.4304 - - -	0.2405 - -
MMP	85.82% - -	0.4121 - - -	0.2397 -
MLPC	86.43% -	0.3922 -	0.2281 =
CMLPC	**86.70%**	**0.3831**	**0.2243**
yeast			
BR	70.41% - - -	8.5811 - - -	0.3152 - - -
MMP	71.39% - - -	7.8867 - - -	0.3099 - - -
MLPC	**75.15%** =	**6.4560** ++	**0.2495** ++
CMLPC	75.05%	6.4921	0.2540

1. the pairwise ranking methods outperform the one-against-all ranking methods (Table 4)[10]
2. adding the calibration label to the ranking does not systematically deteriorate performance (in the *yeast* dataset it lead to worse results, on the *scene* and *Reuters-21578* it helped)
3. the calibrated pairwise method outperforms the binary relevance predictor in terms of multilabel losses (Table 5)
4. the calibrated pairwise method is typically a bit more conservative in predicting the number of relevant labels (lower recall, in particular for *Reuters-21578*) but this is outweighed by its superior ranking performance (higher precision and higher F1)[11]

Essentially, the calibrated pairwise method combines the advantages of both methods: it utilizes the improved ranking capabilities of the pairwise approach, but, through its calibra-

[10]Table 4 shows results of a paired t-test for establishing statistical significance. We also computed Friedman's test and the Bonferroni-Dunn test, as recommended by Demšar (2006). Friedman's tests indicates that the average rankings for the algorithms are significantly different from random (or in other words that at least one of the medians is different from the others). With the Bonferroni-Dunn test, we can only conclude that CMLP is significantly better than BR (the average ranks over all four datasets for the four rank evaluation measures are for CMLP 1.25, MLP 1.75, MMP 3.0, and BR 4.0). However, it should be noted that the critical rank difference for four classifiers is 2.185, i.e., even if the rankings were perfectly consistent (average ranks 1.0 for CMLP and 2.0 for MLP), we could not conclude that CMLP is significantly better than MMP because their rank difference would only be $2.0 < 2.185$.

[11]The astute reader may wonder why for the results of BR on the *scene* dataset the F1 measure is lower than both the recall and the precision measure. This happens because we macro-average the results of the 10 cross-validation folds. If recall is higher than precision on some folds and lower on other folds, the macro-averaged F1 may indeed be lower than both. For example: $r_1 = p_2 = 0.7$, $p_1 = r_2 = 0.3 \rightarrow f_1 = f_2 = f_{\text{avg}} = 0.42$, whereas $r_{\text{avg}} = p_{\text{avg}} = 0.5$.

Table 5 Comparison of the calibration performance of binary relevance ranking (BR), and calibrated pairwise ranking (CMLPC) on the *Reuters-21578*, *scene*, and *yeast* data. The best result in each column is indicated in bold font, significant differences to CMLPC are indicated with +/- (10%), ++/-- (5%), and +++/--- (1%)

	HAMLOSS	PREC	REC	F1
Reuters-21578				
BR	99.60% ---	78.38% ---	**85.21%** +++	81.64% ---
CMLPC	**99.70%**	**89.55%**	80.53%	**84.78%**
scene				
BR	89.58% --	71.80% --	**71.21%** =	71.19% --
CMLPC	**90.31%**	**74.43%**	70.84%	**72.41%**
yeast				
BR	75.91% ---	60.47% ---	59.07% ---	59.76% ---
CMLPC	**77.67%**	**64.01%**	**59.95%**	**61.91%**

tion label, is able to make a good prediction where in the predicted ranking the labels should be separated into relevant and irrelevant labels.

7 Related work

Schapire and Singer (1999) derived a family of multilabel extensions in the framework of AdaBoost (referred to as AdaBoost.MH and AdaBoost.MR) which provided the algorithmic basis for the Boostexter text and speech categorization system (Schapire and Singer 2000). In a similar manner, the multilabel generalization of support vector machines advocated by Elisseeff and Weston (2002) exploits comparative preferences among pairs of labels as defined for the multilabel case in Sect. 2, while lacking a natural approach to determine the relevance zero-point in the multilabel ranking. More recently, Shalev-Shwartz and Singer (2006) presented a general framework for efficiently learning label rankings using the maximum margin methodology. The underlying concept of preference graph decompositions incorporates (Elisseeff and Weston 2002) and several other classification approaches as special cases.

Rousu et al. (2006) consider the hierarchical multilabel learning scenario, a combination of hierarchical classification (Cai and Hofmann 2004) and multilabel learning. Here, the labels are assumed to be organized in a hierarchical structure but in contrast to hierarchical classification an instance may be associated with one or more labels. The underlying classification model is an adaptation of the so-called maximum margin Markov network.

Instance-based approaches to multilabel learning provide an interesting alternative to model-based approaches, such as those discussed in the previous paragraphs, which essentially induce *global prediction models* for the entire input space from the training data. As opposed to model-based approaches which typically entail a substantially increased computational complexity (in comparison to classification learning), there exist conceptually simple and computationally very efficient generalizations to multilabel ranking using the case-based paradigm (Brinker and Hüllermeier 2007; Zhang and Zhou 2005). One of the concepts underlying the case-based generalization in (Brinker and Hüllermeier 2007) is

closely related to the calibration technique advocated in this paper and leads to a rank aggregation optimization problem. The k-nearest neighbor approach to multilabel classification in (Zhang and Zhou 2005) is inspired by Bayesian reasoning.

In a survey on multilabel classification, Tsoumakas and Katakis (2007) distinguish between two categories of approaches, namely, those which employ a *transformation* into a set of binary classification problems and approaches adapting existing methods to handle multilabeled data *directly*. In a certain sense, our approach can be seen as a transformation technique, however, not into a set of binary problems but into a single label ranking problem. A subsequent transformation into the binary domain, such as for ranking by pairwise comparison, is a viable option, but other approaches for solving label ranking problems could also be used (Hüllermeier et al. 2008 contains a brief survey).

Yang (1999) discusses and evaluates a few straight-forward approaches for determining a zero-point. The first one, *RCut*, uses a fixed threshold for all examples (the one that optimizes the $F1$-score), the second approach, *PCut*, suggests to optimize the threshold for each possible label independently, and the final one, *SCut*, proposes to optimize these label thresholds on a separate validation set. Our calibration method compared favorably to the simple *RCut* approach,[12] but we did not compare to the label thresholds. However, none of these approaches allow one to adjust the thresholds for each individual example, which we think is the key distinguishing feature of our proposal.

8 Concluding remarks

We have proposed a generic and unified extension to overcome the severe restriction on the expressive power of previous approaches to label ranking induced by the lack of a calibrated scale. The key idea of this approach is to introduce a calibration label that represents the boundary between relevant and irrelevant labels. This generic approach to the calibrated ranking setting enables general ranking techniques, such as ranking by pairwise comparison and constraint classification, to incorporate and exploit partition-related preference information and to generalize to settings for which predicting the zero-point is required.

In particular, the calibrated extension suggests a conceptually novel technique for extending the common learning by pairwise comparison technique to the multilabel scenario, a setting previously not being amenable to pairwise decomposition. We should mention that in this case the introduction of this calibration label effectively produces an ensemble that combines the models learned by the conventional binary relevance ranking approach and those learned by the pairwise classification approach.

The main conclusion from our experimental results in the areas of text categorization and gene analysis is that the calibrated pairwise approach clearly outperforms the binary relevance approach. We have also seen some evidence that the calibration not only allows one to calibrate the ranking, but that it can also improve the quality of the ranking itself because of the increased redundancy provided by the additional classifiers in the ensemble. We plan a systematic investigation of this issue for its potential of improving the performance in general ranking problems.

A particular limitation of the binary-relevance extension to multilabel ranking, which is not shared by the calibrated framework proposed in this paper, lies in the fact that it only applies to soft classifiers, which are able to provide confidence scores with the prediction.

[12] We had selected the fixed threshold of 3 because it is the median value of classes, but we later confirmed that this is also the one that optimizes the F1-score.

Exploring the benefits of calibrated ranking with binary-relevance classifiers is a promising aspect of future work.

Acknowledgements This research was supported by the German Research Foundation (DFG). We would like to thank the anonymous reviewers and the editor for their helpful suggestions. We also thank Janez Demšar for an interesting discussion on significance tests.

References

Altun, Y., McAllester, D., & Belkin, M. (2006). Margin semi-supervised learning for structured variables. In Y. Weiss, B. Schölkopf, & J. Platt (Eds.), *Advances in neural information processing systems 18*. Cambridge: MIT Press.

Boutell, M. R., Luo, J., Shen, X., & Brown, C. M. (2004). Learning multi-label scene classification. *Pattern Recognition*, *37*(9), 1757–1771.

Bradley, R. A., & Terry, M. E. (1952). The rank analysis of incomplete block designs, I: the method of paired comparisons. *Biometrika*, *39*, 324–345.

Brinker, K., & Hüllermeier, E. (2005). Calibrated label-ranking. In S. Agarwal, C. Cortes, & R. Herbrich (Eds.), *Proceedings of the NIPS-2005 workshop on learning to rank* (pp. 1–6), Whistler, BC, Canada.

Brinker, K., & Hüllermeier, E. (2007). Case-based multilabel ranking. In *Proceedings of the 20th international joint conference on artificial intelligence (IJCAI-07)* (pp. 702–707).

Brinker, K., Fürnkranz, J., & Hüllermeier, E. (2006). A unified model for multilabel classification and ranking. In G. Brewka, S. Coradeschi, A. Perini, & P. Traverso (Eds.), *Proceedings of the 17th European conference on artificial intelligence (ECAI-06)* (pp. 489–493).

Cai, L., & Hofmann, T. (2004). Hierarchical document categorization with support vector machines. In *Proceedings of the 13th ACM conference on information and knowledge management (CIKM-04)* (pp. 78–87), Washington, DC.

Coakley, C. W., & Heise, M. A. (1996). Versions of the sign test in the presence of ties. *Biometrics*, *52*, 1242–1251.

Crammer, K., & Singer, Y. (2003). A new family of online algorithms for category ranking. *Journal of Machine Learning Research*, *3*, 1025–1058.

Dekel, O., Manning, C. D., & Singer, Y. (2004). Log-linear models for label ranking. In S. Thrun, L. K. Saul, & B. Schölkopf (Eds.), *Advances in neural information processing systems 16 (NIPS 2003)* (pp. 497–504). Cambridge: MIT Press.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Elisseeff, A., & Weston, J. (2002). A kernel method for multi-labelled classification. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems 14* (pp. 681–687). Cambridge: MIT Press.

Friedman, J. H. (1996). *Another approach to polychotomous classification* (Technical report). Department of Statistics, Stanford University, Stanford, CA.

Fürnkranz, J. (2003). Round robin ensembles. *Intelligent Data Analysis*, *7*(5), 385–404.

Fürnkranz, J. (2002). Round robin classification. *Journal of Machine Learning Research*, *2*, 721–747.

Fürnkranz, J., & Hüllermeier, E. (2003). Pairwise preference learning and ranking. In N. Lavrač, D. Gamberger, H. Blockeel, & L. Todorovski (Eds.), *Lecture notes in artificial intelligence: Vol. 2837 Proceedings of the 14th European conference on machine learning (ECML-03)*, Cavtat, Croatia (pp. 145–156). Berlin: Springer.

Gärtner, T. (2003). A survey of kernels for structured data. *SIGKDD Explorations*, *5*(1), 49–58.

Har-Peled, S., Roth, D., & Zimak, D. (2002). Constraint classification: a new approach to multiclass classification and ranking. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*. Cambridge: MIT Press.

Hastie, T., & Tibshirani, R. (1998). Classification by pairwise coupling. In M. Jordan, M. Kearns, & S. Solla (Eds.), *Advances in neural information processing systems 10 (NIPS-97)* (pp. 507–513). Cambridge: MIT Press.

Hsu, C.-W., & Lin, C.-J. (2002). A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, *13*(2), 415–425.

Hüllermeier, E., Fürnkranz, J., Cheng, W., & Loza Mencía, E. (2008, in press). Label ranking by learning pairwise preferences. *Artificial Intelligence*.

Knerr, S., Personnaz, L., & Dreyfus, G. (1990). Single-layer learning revisited: A stepwise procedure for building and training a neural network. In F. Fogelman Soulié & J. Hérault (Eds.), *NATO ASI series: Vol. F68. Neurocomputing: algorithms, architectures and applications* (pp. 41–50). Berlin: Springer.

Knerr, S., Personnaz, L., & Dreyfus, G. (1992). Handwritten digit recognition by neural networks with single-layer training. *IEEE Transactions on Neural Networks*, *3*(6), 962–968.

Kreßel, U. H.-G. (1999). Pairwise classification and support vector machines. In B. Schölkopf, C. Burges, & A. Smola (Eds.), *Advances in kernel methods: support vector learning* (pp. 255–268). Cambridge: MIT Press.

Lewis, D. D. (1997). *Reuters-21578 text categorization test collection*. README file (V 1.2), available from http://www.research.att.com/~lewis/reuters21578/README.txt.

Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). RCV1: a new benchmark collection for text categorization research. *Journal of Machine Learning Research*, *5*, 361–397.

Loza Mencía, E., & Fürnkranz, J. (2008). Pairwise learning of multilabel classifications with perceptrons. In *Proceedings of the 2008 international joint conference on neural networks (IJCNN-08)*, Hong Kong (pp. 2900–2907). New York: IEEE.

Lu, B.-L., & Ito, M. (1999). Task decomposition and module combination based on class relations: a modular neural network for pattern classification. *IEEE Transactions on Neural Networks*, *10*(5), 1244–1256.

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, *12*, 153–157.

Park, S.-H., & Fürnkranz, J. (2007). Efficient pairwise classification. In J. N. Kok, J. Koronacki, R. Lopez de Mantaras, S. Matwin, D. Mladenič, & A. Skowron (Eds.), *Proceedings of 18th European conference on machine learning (ECML-07)* (pp. 658–665), Warsaw, Poland. Berlin: Springer.

Price, D., Knerr, S., Personnaz, L., & Dreyfus, G. (1995). Pairwise neural network classifiers with probabilistic outputs. In G. Tesauro, D. Touretzky, & T. Leen (Eds.), *Advances in neural information processing systems 7 (NIPS-94)* (pp. 1109–1116). Cambridge: MIT Press.

Putter, J. (1955). The treatment of ties in some nonparametric tests. *Annals of Mathematical Statistics*, *26*, 368–386.

Rousu, J., Saunders, C., Szedmák, S., & Shawe-Taylor, J. (2006). Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, *7*, 1601–1626.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, *24*(5), 513–523.

Schapire, R. E., & Singer, Y. (2000). BoosTexter: a boosting-based system for text categorization. *Machine Learning 39*(2/3), 135–168.

Schapire, R. E., & Singer, Y. (1999). Improved boosting using confidence-rated predictions. *Machine Learning*, *37*(3), 297–336.

Schmidt, M. S., & Gish, H. (1996). Speaker identification via support vector classifiers. In *Proceedings of the 21st IEEE international conference on acoustics, speech, and signal processing (ICASSP-96)* (pp. 105–108), Atlanta, GA.

Shalev-Shwartz, S., & Singer, Y. (2006). Efficient learning of label ranking by soft projections onto polyhedra. *Journal of Machine Learning Research*, *7*, 1567–1599.

Tsochantaridis, I., Hofmann, T., Joachims, T., & Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st international conference on machine learning (ICML-04)* (pp. 823–830). New York: ACM Press.

Tsoumakas, G., & Katakis, I. (2007). Multi-label classification: an overview. *International Journal of Data Warehousing and Mining*, *3*(3), 1–17.

Weskamp, N., Hüllermeier, E., Kuhn, D., & Klebe, G. (2007). Multiple graph alignment for the structural analysis of protein active sites. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *4*(2), 310–320.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval*, *1*, 69–90.

Zhang, M.-L., & Zhou, Z.-H. (2005). A *k*-nearest neighbor based algorithm for multi-label classification. In *Proceedings of the 1st IEEE international conference on granular computing (GRC-05)* (pp. 718–721).