Multilabel Ranking with Inconsistent Rankers

Xin Geng* and Longrun Luo
School of Computer Science and Engineering
Southeast University, Nanjing, China
{xgeng, luofan}@seu.edu.cn

Abstract

While most existing multilabel ranking methods assume the availability of a single objective label ranking for each instance in the training set, this paper deals with a more common case where subjective inconsistent rankings from multiple rankers are associated with each instance. The key idea is to learn a latent preference distribution for each instance. The proposed method mainly includes two steps. The first step is to generate a common preference distribution that is most compatible to all the personal rankings. The second step is to learn a mapping from the instances to the preference distributions. The proposed preference distribution learning (PDL) method is applied to the problem of multilabel ranking for natural scene images. Experimental results show that PDL can effectively incorporate the information given by the inconsistent rankers, and perform remarkably better than the compared state-of-the-art multilabel ranking algorithms.

1. Introduction

As a particular scenario of preference learning [12], *label ranking* [11, 7, 8, 9] has recently attracted much attention in the machine learning and pattern recognition community. The goal of label ranking is to learn a mapping from the instances to the rankings over a finite set of labels. The increasing interests in label ranking are due to not only many potential applications in image labeling, object recognition, natural language processing, text categorization, etc., but also the fact that it subsumes a number of conventional machine learning paradigms, such as multiclass classification and multilabel classification. *Multilabel ranking*[1] is a combination of multilabel classification and label ranking. Accordingly, there are two main tasks involved in this problem. First, for a given instance, decide a bipartition of the relevant (positive) and irrelevant (negative) labels. Second, for the relevant labels, predict a proper ranking over them.

Existing work on multilabel ranking [4, 5] usually assumes the availability of a ground truth ranking of the relevant labels for each instance in the training set. The ranking reflects some kind of *objective* preference of the relevant labels. However, in many real-world applications, the preference is usually quite *subjective*, i.e., it might be different for different persons. Interpersonal inconsistency in the bipartition of the relevant and irrelevant labels has been observed in multilabel classification. The inconsistency becomes even more severe when the annotators are further required to rank the relevant labels. For example, we have collected the multilabel ranking data on a set of natural scene images from ten human rankers. For each image, the rankers are asked to select from nine candidate labels whatever they think are relevant to the image and rank them according to their relevance. It turns out that the results given by different rankers might be considerably different. Fig. 1 gives one typical example in this data set. The relation $y_i \succ y_j$ means that the ranker thinks that the label y_i is more relevant to the image than y_j is. As can be seen, although there is some common sense (e.g., all of the rankers agree that "water" is a relevant label), obvious inconsistency exists among different rankers, either in the selection of the relevant labels or in the order of the relevant labels.

One straightforward way to solve the inconsistency problem is to aggregate the rankings from different rankers into one ranking by, say, the Borda count voting [16] or the mean rank ordering [5], and then any multilabel ranking algorithms can be applied to the aggregated rankings. This approach might answer the question like "whether the label y_i is preferable to the label y_j?", but cannot answer the question like "how preferable is y_i to y_j?". The latter question becomes more important in case of multiple inconsistent rankers. For example, 6 out of 10 rankers give $y_i \succ y_j$ to the instance $\boldsymbol{x}_1$, and 9 out of 10 rankers give $y_i \succ y_j$ to the instance $\boldsymbol{x}_2$. Through the Borda count voting, the same aggregated result $y_i \succ y_j$ can be derived for both $\boldsymbol{x}_1$ and

*This research was supported by NSFC (61273300, 61232007) and the Key Lab of Computer Network and Information Integration, MOE.

[1]Due to abuse of terminology, "multilabel ranking" in some literature refers to label ranking or multilabel classification, which is out of the scope of this paper.

Ranker	Relevant Label Ranking
01	water ≻ cloud ≻ sky
02	cloud ≻ sky ≻ water ≻ building
03	water ≻ cloud ≻ sky ≻ building
04	water ≻ cloud ≻ sky
05	water ≻ sky ≻ cloud
06	water ≻ cloud ≻ sky
07	water ≻ cloud ≻ sky
08	sky ≻ water
09	water ≻ cloud
10	water ≻ cloud ≻ sky

Figure 1. The relevant label ranking results for one image from ten different human rankers.

$\boldsymbol{x}_2$. However, it is clear that y_i is more preferable to y_j for $\boldsymbol{x}_2$ than for $\boldsymbol{x}_1$. Such information is crucial for a multilabel ranking system to predict rankings that can satisfy as many users as possible. Surprisingly, to the best of our knowledge, this issue has barely been studied up to the present.

In order to explicitly quantize how preferable one label is to another, this paper proposes to learn from the inconsistent rankers a *latent preference degree* $d_{\boldsymbol{x}}^y$, which is a numerical indicator of how preferable the label y is for the instance $\boldsymbol{x}$. Assume that $d_{\boldsymbol{x}}^y \in [0,1]$ and $\sum_y d_{\boldsymbol{x}}^y = 1$. Then, for a particular instance, the latent preference degrees of all the labels constitute a data form similar to probability distribution. So, it is called *preference distribution*. Note that it is impractical for the rankers to directly give their preference distributions toward an instance. Instead, this paper proposes to generate a preference distribution from multiple inconsistent rankings via an optimization process. Then, a learning process is invoked to learn the mapping from the instances to the preference distributions. In the test process, given an instance, its preference distribution is first predicted, and then the labels are ranked according to their latent preference degrees.

The rest of the paper is organized as follows. Section 2 formulates the problems of label ranking and multilabel ranking, and introduces several representative algorithms for them. Section 3 proposes the method to learn from inconsistent rankers based on latent preference distributions. In Section 4, the proposed method is applied to the problem of multilabel ranking for natural scene images and compared with several state-of-the-art multilabel ranking algorithms. Finally, conclusions are drawn in Section 5.

2. Preliminaries

The problem of *label ranking* is to learn a mapping from an instance space $\mathcal{X} = \mathbb{R}^q$ to the rankings over a finite set of labels $\mathcal{Y} = \{y_1, y_2, \cdots, y_c\}$. The ranking is based on a binary order relation $\succ_{\boldsymbol{x}}$, where the subscript $\boldsymbol{x} \in \mathcal{X}$ indicates that the order is determined by $\boldsymbol{x}$. For any two labels $y_i, y_j \in \mathcal{Y}$, $y_i \succ_{\boldsymbol{x}} y_j$ means that, for the instance $\boldsymbol{x}$, y_i is preferable to y_j, and thus y_i is ranked higher than y_j.

While there are many algorithms proposed for label ranking, here we introduce two well-known ones. The first is ranking by pairwise comparison (RPC) [11, 17], which reduces the problem into a number of binary classification problems. The basic idea of RPC is to learn a binary classifier for each pair of labels (y_i, y_j) in $\mathcal{Y}$, resulting in $c(c-1)/2$ models. Each model M_{ij} decides for a given instance $\boldsymbol{x}$ whether $y_i \succ_{\boldsymbol{x}} y_j$ or $y_j \succ_{\boldsymbol{x}} y_i$. During the test process, the test instance is submitted to all the $c(c-1)/2$ models, and the final ranking is obtained through the Borda count voting from the predictions of all the binary classifiers. The second algorithm is label ranking tree (LRT) [7], which is an extension of decision tree. The main modification to conventional decision tree concerns the split criterion at the inner nodes and the stopping criterion for the partitioning. A split is determined by first fitting a Mallows model [18] to each branch node, and then selecting the split that can maximize a weighted average of the within-node variances. A branch node stops growing if the examples in it are completely pure (i.e., with the same ranking) or the number of labels in the node becomes too small.

The basic assumption behind *multilabel ranking* is that there might be multiple labels associated with one instance. Thus, the problem of multilabel ranking involves two equally important targets. The first is a bipartition of $\mathcal{Y}$ into a relevant (positive) label set $P_{\boldsymbol{x}}$ and a irrelevant (negative) label set $N_{\boldsymbol{x}}$, where $P_{\boldsymbol{x}} \cap N_{\boldsymbol{x}} = \emptyset$ and $P_{\boldsymbol{x}} \cup N_{\boldsymbol{x}} = \mathcal{Y}$. The second is a ranking $\succ_{\boldsymbol{x}}$ over $\mathcal{Y}$. Specially, $\succ_{\boldsymbol{x}}$ must satisfy that for $\forall y_i \in P_{\boldsymbol{x}}$ and $\forall y_j \in N_{\boldsymbol{x}}$, there is $y_i \succ_{\boldsymbol{x}} y_j$.

Brinker et al. [4] proposed a method called calibrated label ranking, which can unify the two tasks of multilabel ranking into one framework. By introducing a virtual label y_0 as a split point between the relevant and irrelevant labels, calibrated ranking transforms the multilabel ranking problem into a standard label ranking problem over the extended label set $\mathcal{Y}' = \mathcal{Y} \cup \{y_0\}$. All the labels in $P_{\boldsymbol{x}}$ are ranked before y_0, and all the labels in $N_{\boldsymbol{x}}$ are ranked after y_0. In this way, any algorithm for label ranking can be applied to calibrated ranking as a multilabel ranking algorithm. For example, the two algorithms RPC and LRT mentioned before can be extended to their calibrated versions, namely CRPC and CLRT, respectively.

Brinker and Hüllermeier [5] later proposed a case-based multilabel ranking method as a special case of aggregating rankings. They consider each ranking as a bucket or-

der [10] $(B_1, \cdots, B_{i-1}, \{y_0\}, B_{i+1}, \cdots, B_b)$, where each bucket B_j contains the labels with tie ranks. The special bucket containing the virtual label $\{y_0\}$ is used as a split point between the relevant label set $P_{\boldsymbol{x}} = B_1 \cup \cdots \cup B_{i-1}$ and the irrelevant label set $N_{\boldsymbol{x}} = B_{i+1} \cup \cdots \cup B_b$. A generalized rank $\sigma(i)$ is defined for each label $y_i \in B_j$ as the average overall position $\sigma(i) = \sum_{l<j} |B_l| + \frac{1}{2}(|B_j| + 1)$. Given a query instance $\boldsymbol{x}$, its ranking is obtained by ordering the labels according to the mean generalized ranks of the k nearest neighbors of $\boldsymbol{x}$. It was proved in [5] that such aggregation is optimal in sense of maximizing the sum of the Spearman rank correlation coefficients between the rankings of the k nearest neighbors and the aggregated ranking. This kNN-based multilabel ranking method is denoted by kNN-MLR.

3. Learning from Inconsistent Rankers

For each human ranker, it is reasonable to assume that he/she makes decision according to a latent preference distribution with respect to the given instance, consciously or unconsciously. In practice, it is quite common that for the same instance, different rankers might have different preference distributions. Also, different rankers might prefer different thresholds that distinguish the relevant labels from the irrelevant labels. These differences cause the inconsistency of the rankings given by different rankers for the same instance, in not only the bipartition of the relevant and irrelevant labels, but also the rankings of the relevant labels.

Given an instance $\boldsymbol{x}$, the goal of the proposed method is to predict a label ranking for $\boldsymbol{x}$ that can satisfy the inconsistent rankers as much as possible. We solve the problem via two steps. The first step is to generate a *common* preference distribution for each instance, which is most compatible with the *personal* preference distributions. The second step is to learn a mapping from the instance space to the preference distribution space. The following two sections will introduce these two steps respectively.

3.1. Preference Distribution Transformation

For an instance $\boldsymbol{x}$, suppose that the ranking σ_i results from the i-th ranker's personal preference distribution P_i, then, the common preference distribution $\hat{P}$ for $\boldsymbol{x}$ that generates the common ranking $\hat{\sigma}$ should be most compatible with all P_i, i.e.,

$$\hat{P} = \operatorname*{argmin}_{P} \sum_{i=1}^{r} D(P, P_i), \tag{1}$$

where D is a function measuring the distance between two distributions. P_i should be constrained by the actual ranking given by the i-th ranker, i.e., $P_i^j + \varepsilon \leq P_i^k$ if $y_k \succ_{\boldsymbol{x},i} y_j$, where $P_i^j = d_{\boldsymbol{x},i}^{y_j}$ is the preference degree of y_j for the i-th ranker, $y_k \succ_{\boldsymbol{x},i} y_j$ means that the i-th ranker prefers y_k to y_j for $\boldsymbol{x}$, ε is a predefined margin to avoid too close preference degrees. Specially, if the i-th ranker regards y_j as an irrelevant label, then $P_i^j = 0$. In addition, there should be constraints that ensure both P and P_i to be distributions, i.e., $P^j \geq 0$, $P_i^j \geq 0$, $\sum_j P^j = 1$, and $\sum_j P_i^j = 1$, $j = 1, \cdots, c$, $i = 1, \cdots, r$.

In order to divide the relevant and irrelevant labels, we also insert a virtual label y_0 between $P_{\boldsymbol{x}}$ and $N_{\boldsymbol{x}}$. One problem of this approach is that it penalizes misplaced labels in the ranking equally for all the labels including y_0. However, as the split point of relevant and irrelevant labels, y_0 plays a more important role than other labels, and therefore misplacement of y_0 should be penalized more. In [5], this problem is solved by inserting a set of virtual labels $\{y_{0,1}, \cdots, y_{0,v}\}$ instead of a single one. The virtual labels compose a split bucket, and the size of the bucket v provides a means to increase the penalty of misplacing the virtual labels. Analogously, we also use a set of virtual labels, and in the preference distribution, all the virtual labels have the same preference degree, i.e., $d_{\boldsymbol{x}}^{y_{0,1}} = \cdots = d_{\boldsymbol{x}}^{y_{0,v}} = P^0$.

There are many measures for the distance/similarity between probability distributions [6], which can be used to define D in Eq. (1), such as the distance measures of Euclidean, Sørensen, Squared χ^2, and Kullback-Leibler (K-L), or the similarity measures of Intersection and Fidelity. If the commonly used K-L divergence is adopted here, then the problem can be formulated as the following nonlinear programming process:

$$\begin{aligned}
min \quad & \sum_{i=1}^{r} \left(\sum_{j=1}^{c} P^j \log \frac{P^j}{P_i^j} + vP^0 \log \frac{P^0}{P_i^0} \right) \\
w.r.t. \quad & P^j, P_i^j, i = 1, \cdots, r, j = 0, \cdots, c \\
s.t. \quad & P^j \geq 0, P_i^j \geq 0, \\
& \sum_{j=1}^{c} P^j + vP^0 = 1, \sum_{j=1}^{c} P_i^j + vP_i^0 = 1, \\
& P_i^j + \varepsilon \leq P_i^k, \text{if } y_k \succ_{\boldsymbol{x},i} y_j, \\
& P_i^j = 0, \text{if } y_0 \succ_{\boldsymbol{x},i} y_j, \\
& i = 1, \cdots, r, j = 0, \cdots, c
\end{aligned} \tag{2}$$

The above nonlinear programming problem can be effectively solved by the log barrier interior-point method [21]. Such process is applied to each training instance $\boldsymbol{x}$, resulting in a common preference distribution $P(\boldsymbol{x})$ which incorporates all the rankings from the inconsistent rankers.

3.2. Preference Distribution Learning

After the inconsistent rankings for each training instance have been transformed into a single preference distribution, the next step is to learn a preference distribution model. Formally speaking, the training set now becomes

$G = \{(\boldsymbol{x}_1, P(\boldsymbol{x}_1)), \cdots, (\boldsymbol{x}_n, P(\boldsymbol{x}_n))\}$, where $P(\boldsymbol{x}_i) = \{d_{\boldsymbol{x}_i}^{y_{0,1}}, \cdots, d_{\boldsymbol{x}_i}^{y_{0,v}}, d_{\boldsymbol{x}_i}^{y_1}, \cdots, d_{\boldsymbol{x}_i}^{y_c}\}$ is the preference distribution (including the virtual labels) associated with the training instance $\boldsymbol{x}_i$. The goal of preference distribution learning is to learn a conditional probability mass function $p(y|\boldsymbol{x})$ from G, where $\boldsymbol{x} \in \mathcal{X}$ and $y \in \mathcal{Y}'$. Suppose $p(y|\boldsymbol{x})$ is a parametric model $p(y|\boldsymbol{x};\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the parameter vector. Then, $p(y|\boldsymbol{x};\boldsymbol{\theta})$ is determined by finding the $\boldsymbol{\theta}$ that can generate a distribution similar to $P(\boldsymbol{x}_i)$ given the instance $\boldsymbol{x}_i$. As mentioned in Section 3.1, there are different criteria that can be used to measure the distance or similarity between two distributions. Again, if the K-L divergence is used as the distance measure, then the best parameter vector $\boldsymbol{\theta}^*$ is determined by

$$\begin{aligned}\boldsymbol{\theta}^* &= \operatorname*{argmin}_{\boldsymbol{\theta}} \sum_i \sum_j \left(d_{\boldsymbol{x}_i}^{y_j} \ln \frac{d_{\boldsymbol{x}_i}^{y_j}}{p(y_j|\boldsymbol{x}_i;\boldsymbol{\theta})} \right) \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_i \sum_j d_{\boldsymbol{x}_i}^{y_j} \ln p(y_j|\boldsymbol{x}_i;\boldsymbol{\theta}). \end{aligned} \tag{3}$$

As to the form of $p(y|\boldsymbol{x};\boldsymbol{\theta})$, similar to the work of Geng et al. [13, 14], we assume it to be the maximum entropy model [2], i.e.,

$$p(y|\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{Z} \exp\left(\sum_k \theta_{y,k} \boldsymbol{x}^k \right), \tag{4}$$

where $Z = \sum_y \exp\left(\sum_k \theta_{y,k} \boldsymbol{x}^k\right)$ is the normalization factor, $\theta_{y,k}$ is an element in $\boldsymbol{\theta}$, and $\boldsymbol{x}^k$ is the k-th feature of $\boldsymbol{x}$. Substituting Eq. (4) into Eq. (3) yields the target function

$$\begin{aligned} T(\boldsymbol{\theta}) &= \sum_{i,j} d_{\boldsymbol{x}_i}^{y_j} \sum_k \theta_{y_j,k} \boldsymbol{x}_i^k \\ &\quad - \sum_i \ln \sum_j \exp\left(\sum_k \theta_{y_j,k} \boldsymbol{x}_i^k \right). \end{aligned} \tag{5}$$

Note that in each preference distribution, the preference degrees of the virtual labels should be exactly same. Suppose $d_{\boldsymbol{x}_i}^{y_{0,1}} = d_{\boldsymbol{x}_i}^{y_{0,2}} = \cdots = d_{\boldsymbol{x}_i}^{y_{0,v}} = P^0(\boldsymbol{x}_i)$, and the parameters corresponding to the virtual labels are $\theta_{0,k}$, then Eq. (5) can be rewritten as

$$\begin{aligned} T(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{c} d_{\boldsymbol{x}_i}^{y_j} \sum_{k=1}^{q} \theta_{y_j,k} \boldsymbol{x}_i^k + v P^0(\boldsymbol{x}_i) \sum_{k=1}^{q} \theta_{0,k} \boldsymbol{x}_i^k \right) \\ - \sum_{i=1}^{n} \ln \left(\sum_{j=1}^{c} \exp\left(\sum_{k=1}^{q} \theta_{y_j,k} \boldsymbol{x}_i^k \right) + v \exp\left(\sum_{k=1}^{q} \theta_{0,k} \boldsymbol{x}_i^k \right) \right). \end{aligned} \tag{6}$$

The minimization of $T'(\boldsymbol{\theta}) = -T(\boldsymbol{\theta})$ can be effectively solved by the quasi-Newton method BFGS [19]. The basic idea of BFGS is to avoid explicit calculation of the inverse Hessian matrix in the Newton method by approximating it with an iteratively updated matrix. The computation of BFGS is mainly related to the first-order gradient of $T'(\boldsymbol{\theta})$, which can be obtained by

$$\frac{\partial T'(\boldsymbol{\theta})}{\partial \theta_{y_j,k}} = \begin{cases} \sum_{i=1}^{n} \frac{v \exp\left(\sum_{k=1}^{q} \theta_{0,k} \boldsymbol{x}_i^k \right) \boldsymbol{x}_i^k}{\sum_{j=1}^{c} \exp\left(\sum_{k=1}^{q} \theta_{y_j,k} \boldsymbol{x}_i^k \right) + v \exp\left(\sum_{k=1}^{q} \theta_{0,k} \boldsymbol{x}_i^k \right)} \\ -v \sum_{i=1}^{n} P^0(\boldsymbol{x}_i) \boldsymbol{x}_i^k, & \text{if } j = 0; \\ \sum_{i=1}^{n} \frac{\exp\left(\sum_{k=1}^{q} \theta_{y_j,k} \boldsymbol{x}_i^k \right) \boldsymbol{x}_i^k}{\sum_{j=1}^{c} \exp\left(\sum_{k=1}^{q} \theta_{y_j,k} \boldsymbol{x}_i^k \right) + v \exp\left(\sum_{k=1}^{q} \theta_{0,k} \boldsymbol{x}_i^k \right)} \\ - \sum_{i=1}^{n} d_{\boldsymbol{x}_i}^{y_j} \boldsymbol{x}_i^k, & \text{if } j \neq 0. \end{cases} \tag{7}$$

After the preference distribution model $p(y|\boldsymbol{x};\boldsymbol{\theta})$ has been learned, given a test instance $\boldsymbol{x}'$, its preference distribution is predicted as $p(y|\boldsymbol{x}';\boldsymbol{\theta})$. The labels with a preference degree higher than that of the virtual labels are regarded as the relevant labels, and the rest are regarded as the irrelevant labels. Finally, the relevant labels are ranked in descending order of their preference degrees.

3.3. Evaluation Measures

As mentioned before, multilabel ranking involves two tasks. The first is the bipartition between relevant and irrelevant labels, and the second is the ranking of the relevant labels. Accordingly, a multilabel ranking algorithm should be evaluated by two sets of measures corresponding to the two tasks, respectively. The former task can be evaluated by those commonly used measures for multilabel classification [20], such as *hamming loss*, *one-error*, *coverage*, *ranking loss* and *average precision*. The latter task can be evaluated by those used to measure the similarity between rankings with ties (because the irrelevant labels are regarded as tied) [1], such as *Kendall's tau-b coefficient*, *Kendall's tau-c coefficient*, *Spearman's rho coefficient*, *Gamma correlation coefficient* and *Symmetrically Adjusted Gamma (SAG)*[2]. For each measure, the predicted ranking is compared to the rankings from all the rankers and the average value is calculated as the evaluation result. Note that the virtual labels are removed before evaluation.

4. Experiments

4.1. Methodology

The data set used in the experiments includes $2,000$ natural scene images [22]. There are nine possible labels associated with these images, i.e., *plant*, *sky*, *cloud*, *snow*, *build-*

[2]Due to page limit, the formulae of both the classification and ranking measures are not listed here, but can be found in the references therein.

Table 1. Comparison Results (mean±std (t-test)) of the Four Algorithms on Ten Evaluation Measures

	Measure	PDL	CRPC	CLRT	kNN-MLR
Ranking	Kendall's Tau-b ↑	**0.3869±0.0105**	0.3414±0.0117 (1)	0.3120±0.0109 (1)	0.3686±0.0079 (1)
	Kendall's Tau-c ↑	**0.3405±0.0091**	0.3003±0.0104 (1)	0.2750±0.0095 (1)	0.3243±0.0074 (1)
	Spearman's Rho ↑	**0.5660±0.0097**	0.5257±0.0090 (1)	0.5025±0.0141 (1)	0.5501±0.0074 (1)
	Gamma ↑	**0.6241±0.0215**	0.5533±0.0228 (1)	0.5002±0.0212 (1)	0.5977±0.0137 (1)
	SAG ↑	**0.2534±0.0073**	0.2227±0.0088 (1)	0.2057±0.0068 (1)	0.2404±0.0077 (1)
Classification	Hamming loss ↓	**0.2090±0.0106**	0.2282±0.0100 (1)	0.2336±0.0147 (1)	0.2204±0.0107 (1)
	One-error ↓	**0.3382±0.0262**	0.3922±0.0248 (1)	0.4375±0.0335 (1)	0.3700±0.0252 (1)
	Coverage ↓	**2.9227±0.1573**	3.2790±0.1475 (1)	3.3713±0.1615 (1)	3.0663±0.1113 (1)
	Ranking loss ↓	**0.1688±0.0110**	0.2086±0.0110 (1)	0.2327±0.0107 (1)	0.1837±0.0069 (1)
	Average precision ↑	**0.7426±0.0162**	0.6974±0.0142 (1)	0.6714±0.0145 (1)	0.7215±0.0151 (1)

ing, *desert*, *mountain*, *water* and *sun*. Ten human rankers are requested to label the images. For each image, they first select from the nine candidate labels what they think are relevant to the image, and then rank the relevant labels in descending order of relevance to the image. Each human ranker makes his/her decisions independently, i.e., each ranker can not see the results from other rankers. As expected, the rankings given by different rankers are highly inconsistent, in both the selection of relevant labels and the order of the relevant labels. The average number of relevant labels selected for each image is 2.22. There are only 20.8% images where the ten rankers all agree on the same relevant label set, and 20.6% images where the ten rankers rank the relevant labels in the same order. One typical example in this data set has already been shown in Fig. 1.

The image features are extracted by the method used in [3]. In detail, each color image is first converted into the CIELUV space. Then, the image is divided into 49 blocks via a 7×7 grid. In each block, the mean and variance of each band are computed. Finally, the image is represented by a feature vector of $49 \times 3 \times 2 = 294$ dimensions.

The preference distribution learning method (denoted by PDL) proposed in this paper is compared with the three multilabel ranking algorithms described in Section 2, i.e., CRPC, CLRT and kNN-MLR. Note that none of the three baseline methods can directly deal with inconsistent rankings. So, for each training image, the mean rank ordering process [5] is first run to integrate the rankings from the ten rankers into one, and then CRPC, CLRT and kNN-MLR can be applied to the aggregated ranking. We also tried other ranking aggregation methods, such as the Borda count voting [16], but the mean rank ordering turns out to work better with the baseline methods. For each compared methods, several parameter configurations are tested and the best performance is reported. For all the methods, the number of virtual labels v is set to 5. The margin ε in PDL is set to 0.05. CRPC uses logistic regression as its base classifier, and reaches the final ranking via soft voting on the base classifiers' results. The decision tree in CLRT is learned by the J48 implementation (with its default setting) of WEKA [15]. The number of neighbors k in kNN-MLR is set to 30.

The methods are compared via ten-fold cross-validation. On the test set, the predicted rankings are compared with the actual rankings given by the ten rankers and the average result is recorded. As described in Section 3.3, the evaluation measures include five ranking similarities (*Kendall's tau-b*, *Kendall's tau-c*, *Spearman's rho*, *Gamma* and *SAG*) and five multilabel classification measures (*hamming loss*, *one-error*, *coverage*, *ranking loss* and *average precision*).

4.2. Results

The comparison results of the four algorithms on the ten evaluation measures are tabulated in Table 1. After the name of each measure, "↓" indicates "the smaller the better", and "↑" indicates "the larger the better". Each result is represented by the mean value and standard deviation of the ten-fold cross-validation. The best mean performance on each measure is highlighted by boldface. Also, the two-tailed t-tests at the 5% significance level are performed to see whether the differences between the results of PDL and other algorithms are statistically significant. The results of the t-tests are given in the brackets right after the performances of the baseline methods. The number "1" indicates significant difference, "0" indicates otherwise. As can be seen from Table 1, PDL perform significantly better than all the baseline methods on all evaluation measures. This reveals that, by effectively incorporating the rankings from different rankers, PDL can better accomplish the two tasks of multilabel ranking, i.e., the bipartition of the relevant and irrelevant labels, and the ranking of the relevant labels.

In order to show the influence of the inconsistency among the rankers, the algorithms are also tested while the number of rankers is gradually increased. The results on Kendall's tau-b (the results on other measures are similar and are not shown here due to page limit) are shown in Fig. 2. As can be seen that the ranking prediction accuracy of PDL gradually increases with the increase of rankers, while that of the baseline methods does not always get better. The performance of kNN-MLR does not change much, and that of CRPC and CLRT exhibits random variations. Each of the three baseline methods shows performance deterioration at certain points. CLRT performs even worse with ten rankers than with just one. The superiority of PDL over the baseline methods becomes more and more significant while the

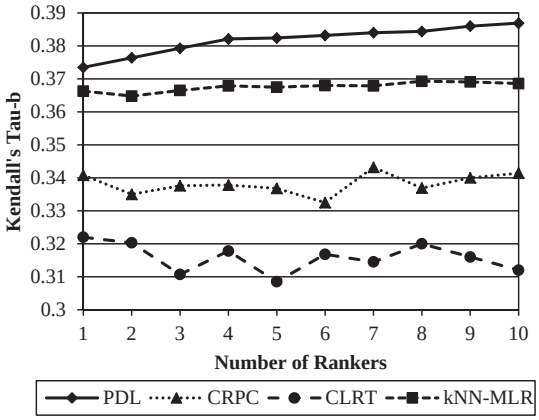

Figure 2. The variation of Kendall's tau-b with gradually increasing rankers.

number of rankers increases. This reveals that PDL can effectively utilize the additional information given by the additional rankers, while the inconsistency among different rankers may cause serious trouble to the baseline methods.

5. Conclusion

This paper proposes to learn a latent preference distribution from multiple inconsistent rankers. The proposed preference distribution learning (PDL) method mainly includes two steps. The first step is to generate a common preference distribution for each instance, which is most compatible to the personal rankings from all rankers. The second step is to learn a mapping from the instances to the preference distributions. PDL is applied to the problem of multilabel ranking for natural scene images and performs remarkably better than the compared well-known multilabel ranking algorithms.

References

[1] A. Agresti. *Analysis of Ordinal Categorical Data*. John Wiley & Sons, New York, 2nd edition, 2010. 4

[2] A. L. Berger, S. D. Pietra, and V. J. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996. 4

[3] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004. 5

[4] K. Brinker, J. Fürnkranz, and E. Hüllermeier. A unified model for multilabel classification and ranking. In *Proc. 17th European Conf. Artificial Intelligence*, pages 489–493, Riva del Garda, Italy, 2006. 1, 2

[5] K. Brinker and E. Hüllermeier. Case-based multilabel ranking. In *Proc. of the 20th Int'l Joint Conf. Artificial Intelligence*, pages 702–707, Hyderabad, India, 2007. 1, 2, 3, 5

[6] S.-H. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *Int'l Journal of Mathematical Models and Methods in Applied Sciences*, 1:300–307, 2007. 3

[7] W. Cheng, J. C. Huhn, and E. Hüllermeier. Decision tree and instance-based learning for label ranking. In *Proc. 26th Int'l Conf. Machine Learning*, page 21, Montreal, Canada, 2009. 1, 2

[8] W. Cheng, E. Hüllermeier, W. Waegeman, and V. Welker. Label ranking with partial abstention based on thresholded probabilistic models. In *Advances in Neural Information Processing Systems 25*, pages 2510–2518, Lake Tahoe, NV, 2012. 1

[9] S. Destercke. A pairwise label ranking method with imprecise scores and partial predictions. In *Proc. European Conf. Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 112–127, Prague, Czech Republic, 2013. 1

[10] R. Fagin, R. Kumar, M. Mahdian, D. Sivakumar, and E. Vee. Comparing and aggregating rankings with ties. In *Proc. 23rd ACM Symposium Principles of Database Systems*, pages 47–58, Paris, France, 2004. 3

[11] J. Fürnkranz and E. Hüllermeier. Pairwise preference learning and ranking. In *Proc. 14th European Conf. Machine Learning*, pages 145–156, Cavtat-Dubrovnik, Croatia, 2003. 1, 2

[12] J. Fürnkranz and E. Hüllermeier, editors. *Preference Learning*. Springer, Berlin, 2011. 1

[13] X. Geng, K. Smith-Miles, and Z.-H. Zhou. Facial age estimation by learning from label distributions. In *Proc. 24th AAAI Conf. Artificial Intelligence*, pages 451–456, Atlanta, GA, 2010. 4

[14] X. Geng, C. Yin, and Z.-H. Zhou. Facial age estimation by learning from label distributions. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 35(10):2401–2412, 2013. 4

[15] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009. 5

[16] E. Hüllermeier and J. Fürnkranz. Ranking by pairwise comparison: A note on risk minimization. In *Proc. IEEE Int'l Conf. Fuzzy Systems*, pages 97–102, Budapest, Hungary, 2004. 1, 5

[17] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker. Label ranking by learning pairwise preferences. *Artif. Intell.*, 172(16-17):1897–1916, 2008. 2

[18] C. Mallows. Non-null ranking models. *Biometrika*, 44(1):114–130, 1957. 2

[19] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, New York, NY, 2nd edition, 2006. 4

[20] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *Int'l Journal of Data Warehousing and Mining*, 3(3):1–13, 2007. 4

[21] R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimizationthat combines line search and trust region steps. *Mathematical Programming*, 107:391–408, 2006. 3

[22] M.-L. Zhang and Z.-H. Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007. 4