A Generic Multilabel Learning-Based Classification Algorithm Recommendation Method

GUANGTAO WANG, QINBAO SONG, XUEYING ZHANG, and KAIYUAN ZHANG,
Xi'an Jiaotong University, China

As more and more classification algorithms continue to be developed, recommending appropriate algorithms to a given classification problem is increasingly important. This article first distinguishes the algorithm recommendation methods by two dimensions: (1) meta-features, which are a set of measures used to characterize the learning problems, and (2) meta-target, which represents the relative performance of the classification algorithms on the learning problem. In contrast to the existing algorithm recommendation methods whose meta-target is usually in the form of either the ranking of candidate algorithms or a single algorithm, this article proposes a new and natural multilabel form to describe the meta-target. This is due to the fact that there would be multiple algorithms being appropriate for a given problem in practice. Furthermore, a novel multilabel learning-based generic algorithm recommendation method is proposed, which views the algorithm recommendation as a multilabel learning problem and solves the problem by the mature multilabel learning algorithms. To evaluate the proposed multilabel learning-based recommendation method, extensive experiments with 13 well-known classification algorithms, two kinds of meta-targets such as algorithm ranking and single algorithm, and five different kinds of meta-features are conducted on 1,090 benchmark learning problems. The results show the effectiveness of our proposed multilabel learning-based recommendation method.

Categories and Subject Descriptors: H.2.8 [**Database Applications**]: Data Mining

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Algorithm automatic recommendation, multiple comparison procedure, multilabel classification, multilabel k nearest neighbors

ACM Reference Format:
Guangtao Wang, Qinbao Song, Xueying Zhang, and Kaiyuan Zhang. 2014. A generic multilabel learning-based classification algorithm recommendation method. ACM Trans. Knowl. Discov. Data 9, 1, Article 7 (October 2014), 30 pages.
DOI: http://dx.doi.org/10.1145/2629474

1. INTRODUCTION

With the increasing requirements on learning interesting knowledge from data, researchers have proposed many learning algorithms based on different hypothesis spaces. Meanwhile, more and more novel or improved algorithms are being proposed continuously.

However, both the published experimental results [Brazdil et al. 2003; Bensusan 1998; King et al. 1995; Brazdil and Soares 2000; Kalousis and Theoharis 1999; Smith-Miles 2008; Lindner and Studer 1999; Ali and Smith 2006; Yang and Jiu 2006; Song et al. 2012; Prudêncio et al. 2011a] and the famous NFL (No Free Lunch) theory

This work is supported by the National Natural Science Foundation of China under grant 61373046.
Authors' address: Department of Computer Science and Technology, No. 28, Xianning West Road, Xi'an, Shaanxi, 710049, P.R. China.

DOI: http://dx.doi.org/10.1145/2629474

[Wolpert 2001] have demonstrated that no specific classification algorithm performing well on all learning problems exists. Thus, in practical applications, it is hard to decide which classification algorithm(s) should be selected for a given problem. A straightforward method is to apply all the candidate algorithms on the given problem and choose the one(s) with better performance as the appropriate one(s). However, the method is untutored, so it is time-consuming and can be impractical when the number of candidate algorithms is large and/or the time complexities of the algorithms are high. Thus, it is necessary to explore an algorithm automatic recommendation method to assist users in choosing appropriate algorithms.[1] With the algorithm automatic recommendation method, we can:

(1) understand how the characteristics of a learning problem affect the performance of algorithms, then pick up the appropriate ones to effectively address the problem at hand, and save time and computing resources; and
(2) construct better methods by combining different algorithms in an innovative way according to the characteristics of learning problems, such as assisting in the selection of more appropriate base learners in a boosting algorithm.

Algorithm recommendation is based on the interaction between the characteristics of a problem and the performance of the candidate algorithms [Smith-Miles 2008; Rice 1976; Brazdil et al. 2009]. It can be viewed as an application of meta-learning [Vilalta and Drissi 2002]; that is, it is a meta-level learning problem whose learning target is the relative performance of candidate algorithms on a given problem.

In this article, we first distinguish different meta-level learning-based algorithm recommendation methods by the following two critical aspects: (1) which kind of features are used to characterize a given problem (i.e., meta-features) and (2) the expression form of the learning target in the meta-level (i.e., meta-target), that is, how to describe the proper candidate algorithms of the given problem. For example, of the existing algorithm recommendation methods, some of them assume that there is a single optimal algorithm for a given problem. Here, the meta-target is in the form of a single algorithm [Ali and Smith 2006; Michie et al. 1994; Kalousis and Hilario 2000]; others believe that each of the candidate algorithms has the possibility to be an appropriate one, weigh all the candidate algorithms, and provide a ranking of the algorithms on the given problem [Brazdil et al. 2003; Brazdil and Soares 2000; Prudêncio et al. 2011a; De Souto et al. 2008; Soares et al. 2009; Aiguzhinov et al. 2010]. Obviously, this time the meta-target is in the form of the ranking of algorithms. All of this research work gives us some useful guidelines for algorithm recommendation.

However, either in theory or in practice, there could be multiple algorithms appropriate for a given problem, and the number of appropriate algorithms varies with different problems (see details in Section 2.3.3). Thus, algorithm recommendation can be viewed as a multilabel learning problem where the meta-target can be in the form of multiple algorithms. Inspired by this, we propose a novel multilabel learning-based algorithm recommendation method, which is not only a more reasonable way to address algorithm recommendation problems but also an expanded application of multilabel learning.

For the purpose of demonstrating how well the proposed multilabel learning-based algorithm recommendation method works, we constructed the recommendation model for classification algorithm recommendation based on an instance-based multilabel learning algorithm MLkNN (Multilabel k Nearest Neighbors) and compared it with several well-known recommendation methods over 1,090 classification problems.

[1]In this article, the phrase "algorithm recommendation" refers to "classification algorithm recommendation" if there is no special statement.

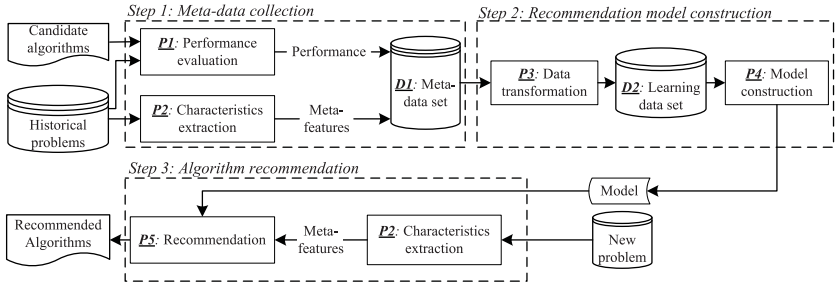

Fig. 1. Framework of algorithm recommendation.

The rest of the article is organized as follows. Section 2 presents a framework for algorithm recommendation and distinguishes different recommendation methods by two dimensions. Section 3 introduces the novel multilabel learning-based method for algorithm recommendation. Section 4 experimentally demonstrates the performance of the proposed method. Finally, Section 5 concludes the article.

2. ALGORITHM RECOMMENDATION FRAMEWORK

In this section, we first present a framework to describe the process of algorithm recommendation. Then, based on this framework, we review the commonly used meta-features in algorithm recommendation in Section 2.2, and we group the meta-targets in algorithm recommendation into three categories and state the rationality of the multilabel-based meta-target in Section 2.3.

2.1. The Framework

Algorithm recommendation was first formally described by Rice [1976]. Later, the formalism had a widespread application in the field of algorithm recommendation [Smith-Miles 2008]. In this article, we follow this formalism as well. Specially, the algorithm recommendation is viewed as a process to explore the interaction between characteristics of the problem (referred to as "meta-features" in this article) and the performance of algorithms over the problem (referred to as "meta-target" in this article) and utilizes this interaction to select appropriate algorithm(s) for a new problem.

By analysis of the existing algorithm recommendation methods [Brazdil et al. 2003; King et al. 1995; Hilario and Kalousis 2001; Smith-Miles 2008; Song et al. 2012; Rice 1976; Brazdil and Soares 2000; Brazdil et al. 1994; Sohn 1999; Kalousis and Theoharis 1999; Ali and Smith-Miles 2006; Ali and Smith 2006; De Souto et al. 2008; Prudêncio et al. 2011a, 2011b; Peng et al. 2002; Kalousis and Hilario 2000; Köpf et al. 2000; Kalousis et al. 2004; Kalousis 2002; Brodley 1993; Pfahringer et al. 2000; Arinze et al. 1997; Guo 2003], we can see that algorithm recommendation can be viewed as a learning problem, which consists of three steps: (1) meta-data collection, (2) recommendation model construction on the meta-data, and (3) algorithm recommendation for a new problem with the constructed model. Figure 1 shows the details.

(1) *Meta-data collection*
The meta-data is collected from a set of historical problems (denoted as *PSet*) and a number of candidate algorithms (denoted as *ASet*). Each instance of the meta-data corresponds to a problem $p \in PSet$. Similar to an instance of a supervised learning problem, the instance consists of two parts: (1) meta-features and (2) a learning target, that is, meta-target. Here, the meta-features are a number of specific characteristics extracted from p (see Procedure *P2*, "Characteristics extraction," in Figure 1), and the learning target is the performance metrics evaluated on p

with all the algorithms in *ASet* (see Procedure *P1*, "Performance evaluation," in Figure 1).

(2) *Recommendation model construction*
The learning target achieved from the former procedure is a set of performance metrics. However, our intent is to construct the recommendation model that can be used for choosing appropriate algorithm(s) rather than predicting the absolute performance metrics. Therefore, we need to transform the absolute metric values into another form: the appropriate algorithm with the optimal performance, or the algorithms with no significant difference to the optimal one. At the end of this step, the meta-data will be transformed into a learning dataset (see Procedure *P3*, "Data transformation," in Figure 1). Finally, we can utilize a specific learning algorithm to construct the learning model on the transformed dataset. The algorithm recommendation model is obtained (see Procedure *P4*, "Model construction," in Figure 1).

(3) *Algorithm recommendation for a new problem*
Once the recommendation model is constructed, it is straightforward to recommend appropriate algorithms for a new problem. That is, first, the meta-features of the problem are extracted using Procedure *P2*, "Characteristics extraction"; then, the appropriate algorithms are recommended according to the constructed recommendation model and the meta-features with Procedure *P5*, "Recommendation."

2.2. Meta-Features

Meta-features are a set of measures extracted from the learning problem and used to uniformly characterize different problems. In the field of algorithm recommendation, the meta-features should (1) have a unified form for different problems, (2) be convenient for calculating, and (3) be tightly related to the performance of classification algorithms.

For algorithm recommendation, problem characterization is a very challenging job [Smith-Miles 2008; Vilalta and Drissi 2002], and researchers have proposed a number of features to characterize various problems. However, to the best of our knowledge, there still does not exist an acknowledged method to characterize different problems or tell us which features are salient.

The meta-features in the existing literature can be summarized into four categories. Moreover, Song et al. [2012] recently proposed a novel dataset feature extraction method that uses structural information to characterize datasets and is quite different from the existing ones. Next, we briefly introduce these five different kinds of meta-features one by one. For the details of these meta-features, please refer to the appendix.

(1) *Statistical and Information-Theory-Based Measures*
The statistical and information-theory-based measures are the most widely used in the field of algorithm recommendation [Brazdil et al. 2003; King et al. 1995; Kalousis and Theoharis 1999; Lindner and Studer 1999; Michie et al. 1994; Brazdil et al. 1994; Sohn 1999; Henery 1994; Aha 1992; Gama and Brazdil 2000; Engels and Theusinger 1998]. The prominent examples based on these measures are the projects ESPRIT Statlog (1991–1994) and METAL (1998–2001). These measures generally include such dataset characteristics as number of features, number of instances, number of target concepts, ratio of instances to features, ratio of missing values, ratio of binary features, entropy of the target concept, information gain between the feature and the target concept, and correlation coefficient between features.

(2) *Model-Structure-Based Measures*
For this kind of measure, a learning problem is represented in a special data structure that can embed the complexity of a problem. Then, the characteristics of the structure are exploited to describe the learning problem.

In the field of algorithm recommendation, the induced decision tree is a well-known and commonly used structure to model a learning problem. Bensusan [1998] proposed capturing the information from the induced decision tree for describing the learning complexity. He extracted 10 measures from the decision tree, such as the ratio of the number of nodes to the number of features, the ratio of the number of nodes to the number of instances, and so forth. Afterward, Peng et al. [2002] reanalyzed the characterization of decision trees and proposed some new measures to characterize the structural properties of decision trees.

(3) *Landmarking-Based Measures*
This kind of measure falls within the concept of landmarking [Pfahringer et al. 2000; Bensusan and Giraud-Carrier 2000; Jain et al. 2000; Duin et al. 2004]. This idea was proposed based on the assumption that the performance of the candidate algorithms could be predicted by the performance of a set of simple learners (also called landmarkers). The performance (e.g., accuracy) of these landmarkers is used to describe a learning problem. Evidently, this kind of measure depends on the choice of landmarkers. In practice, it should be ensured that the chosen landmarkers have significant differences in terms of learning mechanism.

(4) *Problem-Complexity-Based Measures*
In addition to the previous problem characterization methods, the problem-complexity-based measures also were exploited to describe the learning problems in Bernadó-Mansilla and Ho [2005], Ho and Basu [2002], Elizondo et al. [2009], and Ho [2000]. By analyzing the source of difficulty in solving a learning problem, they focus on the description of the geometrical complexity of the problem and emphasize the geometrical characteristics of the distributions of the classes. The features reflecting the way in which different classes are separated or interleaved (and being relevant to learning performance) are identified as the measurement of the problem's complexity. Examples include Fisher's discriminant ratio, the percentage of instances in the problem that lie next to the class boundary, and the nonlinearity of linear/nonlinear learning algorithm.

(5) *Structural-Information-Based Measures*
Recently, Song et al. [2012] proposed a novel problem characterization method to facilitate the algorithm recommendation. The method uses structural-information-based feature vectors to characterize the learning problems, which is quite different from the existing methods. Specially, they first calculate the frequencies of the 1-itemsets and 2-itemsets of a learning problem, and then extract the quantiles of these frequency sequences as the feature vectors of the learning problem. In this article, we name this kind of problem characteristic as structural-information-based measures.

2.3. Meta-Target

As we know, algorithm recommendation can be viewed as a learning problem whose independent variables are meta-features and learning target is the meta-target. Here, the meta-target represents the relative performance of the candidate classification algorithms on the problem at hand. The relative performance can be expressed in different forms. The existing frequently used meta-target is either in the form of single label or in the form of algorithm ranking.

However, in this article, we suppose that the meta-target could be in a multiple-label form, as it is possible that there are multiple algorithms being appropriate for a problem. Moreover, for different forms of meta-targets, the methods used to construct a recommendation model on the meta-data are usually different as well. Thus, we introduce these three different forms of meta-target in the following subsections respectively.

2.3.1. Single Label. In this situation, for a given learning problem, the algorithm with the best performance is usually identified as the meta-target. That is, the meta-target is in the form of a single algorithm. Thus, the algorithm recommendation can be viewed as a single-label classification problem. So, theoretically, all the single-label learning algorithms can be used to construct a recommendation model. In practical applications, to get more straightforward and understandable recommendation models, the tree-based or rule-based learning algorithms are usually preferred.

Furthermore, in order to measure the performance of the constructed recommendation model, the metrics evaluating the performance of the single-label learning algorithm are usually employed; for example, the ratio of the incorrectly recommended problems represents the error rate.

2.3.2. Ranking. Algorithm ranking is a set of sorted values corresponding to a set of candidate algorithms. These rank values are achieved based on the performance metrics of these algorithms on a given problem. The algorithm with the highest metric achieves the top rank, the one with the second highest metric gets the second rank, and so forth. The algorithm ranking is first employed as the meta-target for algorithm recommendation in Brazdil et al. [1994]. After that, Brazdil et al. [2003], Brazdil and Soares [2000], and Nakhaeizadeh and Schnabl [1997] addressed algorithm recommendation based on a similar approach.

Because the meta-target is a set of ranking values instead of a single symbolic value, it is impossible to directly employ the single-label learning algorithms to construct recommendation models. In this case, researchers usually resort to the instance-based methods or their variations [Brazdil et al. 2003] for algorithm recommendation.

Meanwhile, since the meta-target is represented as the rank of algorithms, the recommendation result for a new problem is in the form of a rank list. In order to evaluate the performance of the recommendation model, the *Spearman Correlation Coefficient* [Brazdil et al. 2003; Fürnkranz and Hüllermeier 2010], which is defined as the Pearson correlation coefficient between two ranked variables, is employed.

For a given learning problem, this kind of algorithm recommendation method just provides a ranking list of the candidate algorithms. It inherently suggests to us that the top-ranked algorithms should be picked up as the appropriate ones. Yet, not all the candidate algorithms are appropriate, and it could not tell us the exact number of recommended algorithms. Also, there is still no effective method to preassign the number of algorithms that should be recommended from a ranking list in advance. Moreover, in practice, users may use their favorite algorithms (e.g., the tree-based learning algorithms supply more interpretative results than neural network-based ones) whose performance is slightly below the top-ranked algorithm. But the algorithm ranking-based recommendation methods do not show whether the recommended algorithms with slight performance reduction are very effective. Thus, in order to handle this problem and recommend all appropriate algorithms for users, the multilabeled meta-target is presented in the next section.

2.3.3. Multiple Label. For a given problem and a set of candidate algorithms, it is reasonable to assume that there are multiple algorithms appropriate for the problem. This can be demonstrated as follows:

(1) There exist multiple algorithms performing equivalently on a given problem because of the intrinsic properties of the problem. For example, for a given linearly separable classification problem, learning algorithms RBF (Radial Basis Function), MLP (Multiple Layer Perceptrons), and SVM (Support Vector Machine) are able to handle the problem well. All these algorithms can find the optimal splitting line for the problem and construct the optimal prediction model. Therefore, if we pursue correct prediction on the new instances, all of these learning algorithms are the appropriate choices.

Recently, Lee and Giraud-Carrier [2013] clustered algorithms based on their classification behavior and proposed an algorithm recommendation method that can recommend a cluster of algorithms rather than an individual algorithm for a given problem and that works well on real-world problems. Since each cluster can be viewed as a form of multilabel, this offers further support to our work where we can assign different multilabels for each dataset rather than a fixed one based on prior clustering.

(2) In practical application, it is difficult for us to obtain enough prior information about the problem at hand, such as whether the problem is linearity separable. Moreover, for most new proposed algorithms, their processing ability (i.e., the scope of the problems that they can effectively solve) still needs further investigation. Thus, it is difficult to select the appropriate algorithms by matching the properties of the problem with the processing abilities of the algorithms.

Because of those reasons, in order to identify the most appropriate algorithm for a given problem, we usually turn to experimental investigation. That is, through sophisticatedly designed experiments, we experimentally evaluate the candidate algorithms by estimating the performance metrics and select the algorithm with the best estimated performance as the appropriate one. However, picking up the algorithm with the best performance as the appropriate one is not sufficient. Because the performance metrics are usually estimated based on a special sample instead of overall sample space, these estimated metrics are inherently statistical. Further, the simple comparison of these metrics is not sufficient since the observed differences might not be significant in a statistical context. This means the differences might be random. Furthermore, identifying the algorithm with the best metric value as the optimal one could be arbitrary as there might be no statistically significant differences among some algorithms in terms of the metrics. In this case, it is more reasonable to maintain all the algorithms that are not significantly different from the algorithm with the best metric as the appropriate ones. Thus, there will be multiple algorithms being appropriate for a given problem.

(3) Empirical results of appropriate algorithm identification from 13 candidate learning algorithms over 1,090 benchmark learning problems as follows also provide us strong evidence: the number of appropriate algorithms on most problems is multiple. This conclusion is illustrated by Figure 2.

An algorithm is deemed appropriate for a learning task if the statistical test shows that it has no significant difference with the best algorithm in terms of the given performance metric (e.g., classification accuracy or ARR).

Figure 2 shows the distribution of the frequency of the number of appropriate algorithms on the 1,090 learning problems. The frequency of a specific number q is denoted as f_q and computed as follows: suppose the number of the problems with q appropriate algorithms is n_q, and then $f_q = n_q/1090$.

The four subfigures in Figure 2 correspond to the distributions of the metatarget in terms of four different metrics used to evaluate the learning algorithms: classification accuracy and ARR with $\alpha = 0.1\%$, 1%, and 10%, respectively.

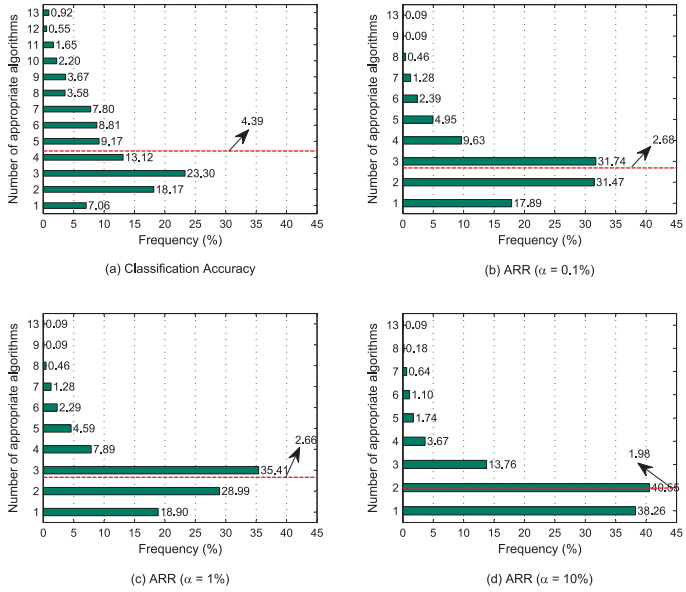

Fig. 2. Distribution of the number of the appropriate algorithms.

For each subfigure, the red line denotes the label cardinality of the multilabeled meta-data. The label cardinality is an important metric to characterize a multilabel dataset [Tsoumakas et al. 2010], and it indicates the average number of labels of instances in a dataset. In this figure, it denotes the average number of appropriate algorithms on the learning problems and is calculated by $\sum_{q=1}^{k}(f_q \times q)$, where k is the number of all possible target labels (e.g., in this experiment, $k = 13$ reveals the number of candidate algorithms).

From Figure 2, we can observe that the proportion of the problems with only one appropriate algorithm is only 7.06% in terms of classification accuracy (see the bottom bar in Figure 2(a)), 17.89% in terms of the multiple criteria metric ARR with $\alpha = 0.1\%$ (see the bottom bar in Figure 2(b)), 18.9% in terms of ARR with $\alpha = 1\%$ (see the bottom bar in Figure 2(c)), and 38.26% in terms of ARR with $\alpha = 10\%$ (see the bottom bar in Figure 2(d)). This reveals that, for most problems, there would be more than one algorithm being appropriate. Thus, it is rational to employ multilabel learning methods for algorithm recommendation.

According to previous discussion, we believe that the algorithm recommendation in nature is closer to a multilabel learning problem. Therefore, the multilabel classification, which is drawing more and more attention in the field of data mining [Tsoumakas and Katakis 2007; Tsoumakas et al. 2010; Spyromitros Xioufis et al. 2011; Katakis et al. 2008; Clare and King 2001; Veloso et al. 2007; Esuli et al. 2008; Zhang 2009; Godbole and Sarawagi 2004; Zhang and Zhou 2007b; Brinker and Hüllermeier 2007; Thabtah et al. 2004; Spyromitros et al. 2008; Cheng and Hüllermeier 2009], can be employed to construct algorithm recommendation models. Furthermore, we can borrow the systematic assessment criteria used in multilabel classification to uniformly measure different recommendation methods (see details in Section 3.3). This will be quite useful for us to identify which data characterizing method or recommendation procedure is more effective and get more insightful knowledge to guide the algorithm recommendation.

3. A GENERIC MULTILABEL LEARNING-BASED ALGORITHM RECOMMENDATION METHOD

According to Section 2.3.3, it is reasonable and practical to view algorithm recommendation as a multilabel learning problem, although multilabel learning algorithms have not been used for classification algorithm recommendation up to now.

Certainly, the proposed multilabel learning-based algorithm recommendation method is a specific implementation of the framework shown in Figure 1. The method consists of two procedures: (1) collecting the multilabel-based meta-data and (2) constructing a recommendation model by multilabel learning algorithm. For the first one, how to identify the multiple appropriate algorithms for the given problem is critical, and this is addressed by the well-known *multiple comparison procedure* [Toothaker 1993; Dunnett 1955], which is briefly introduced in Section 3.1. For the second procedure, we briefly introduce the multilabel learning algorithms in Section 3.2.

Finally, based on the metrics used to evaluate the performance of multilabel learning algorithms, we define several measures to assess the performance of different algorithm recommendation methods in Section 3.3.

3.1. Multiple Comparison Procedure

For algorithm recommendation, the meta-target collection is the foundation. The quality and correctness of the collected meta-target will govern the reliability and further the practical applicability of the constructed recommendation models.

For a given problem, the meta-target collection is a process of picking up the appropriate algorithms from a set of candidate algorithms based on the specific performance evaluation metrics. The metrics are generally divided into two categories, single-criterion metric (such as classification accuracy, AUC, runtime, etc.) and multicriteria metric [Nakhaeizadeh and Schnabl 1997] (such as ARR (Adjusted Ratio of Ratios) [Brazdil et al. 2003] considering both the accuracy and the runtime of an algorithm).

In order to identify the appropriate algorithms (i.e., meta-target) for a given problem from a set of candidate algorithms in terms of a given performance metric (e.g., classification accuracy), the statistical algorithm selection is a reasonable and commonly used approach [Pizarro et al. 2002].

For algorithm recommendation, the number of candidate algorithms is usually large. To find out the superior algorithms from three or more candidate algorithms, the traditional statistical methods usually resort to multiple paired t-tests. However, it has been proved that this approach usually leads to high type I error[2] [Dunnett 1955; Pizarro et al. 2002; Salzberg 1997; Hommel 1988; Demšar 2006]. That means the probability that we falsely exclude the algorithms not significantly differing from the best one(s) will be high. If the collected meta-target does not depict the actual situation, it will go against constructing the accurate recommendation models.

For solving this problem, we turn to the multiple comparison procedure. The multiple comparison procedure is a statistical test technique that helps us compare three or more groups of metrics (e.g., classification accuracy) while controlling the probability to make the statistical type I error [Pizarro et al. 2002; Salzberg 1997]. Moreover, it allows us to be concerned with a set of learning algorithms not significantly different from the best one rather than a single learning algorithm. Therefore, the multiple comparison procedure is an effective method for multilabeled meta-target collection.

Figure 3 shows the guidelines for comparing multiple learning algorithms, which is adapted from Pizarro et al. [2002].

[2]The probability that we make a mistake to reject the null hypothesis, that is, a misjudgment to say there exists significant difference but actually there does not.

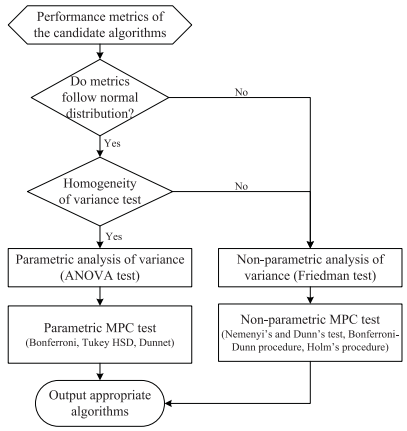

Fig. 3. Statistical procedure for appropriate algorithm identification.

Figure 3 indicates that the parametric statistical procedure ANOVA (Analysis of Variation) should be performed under a much stricter condition: meeting both the normal distribution and variance homogeneity. Otherwise, its test power[3] will be reduced (i.e., failing to distinguish the real differences among the candidate algorithms) and will even draw the wrong conclusion when the distribution of the metrics is highly skewed or the variances of metrics of the candidate algorithms are very different. Thus, in real-world applications, one should be careful when applying the parametric statistical tests.

Unfortunately, the stricter condition is most probably violated when analyzing the performance of machine-learning algorithms [Demšar 2006]. In the case where the condition is not met, nonparametric Friedman's analysis of variance by ranks can be used since it is distribution independent.

Moreover, it has been demonstrated that, in the case where the distribution is normal, the nonparametric procedures have comparable test power with the parametric procedures. On the other hand, when the distribution is nonnormal, the nonparametric procedures will be more powerful than the parametric ones [Conover and Iman 1981, 1982]. Therefore, when the distribution of the metrics is unknown or difficult to be tested, the nonparametric statistical procedure will be a good choice.

Therefore, in order to find all the appropriate algorithms for a given problem at hand, we should choose the test with the higher test power as the post hoc test procedure. In this article, as suggested in Demšar [2006], we employ Friedman's followed by Holm's procedure test to identify the appropriate algorithms for a given problem. If the result of the Friedman test shows that there is no significant performance difference among the candidate learning algorithms, all these algorithms are viewed as the appropriate ones. Otherwise, the learning algorithm with the best performance is viewed as a reference, and Holm's procedure test is performed to identify the appropriate algorithms from the rest. The algorithms that have no significant differences with the reference are viewed as the appropriate algorithms. Of course, the reference is an appropriate one as well.

3.2. Recommendation Model Construction Based on Multilabel Learning

Once we obtained the multilabel-based meta-data where multiple appropriate classification algorithms are assigned to each learning problem, the recommendation model,

[3]The power of a test refers to the ability of the test to find out the source of differences, which leads to the rejection of the null hypothesis.

which is in the form of a multilabel classifier, is conducted via multilabel learning on the meta-data collected from the historical learning problems.

There are various multilabel learning algorithms proposed with different assumptions. These algorithms are usually grouped into two general categories: problem transformation and algorithm adaptation [Tsoumakas and Katakis 2007; Tsoumakas et al. 2010].

(1) The problem transformation methods first transform a multilabel learning problem into one or more single-label learning problems, and then a single-label learning algorithm is applied. Many problem transformation methods have been proposed [Boutell et al. 2004; Chen et al. 2007; Tsoumakas and Vlahavas 2007; Hüllermeier et al. 2008; Mencía and Furnkranz 2008; Fürnkranz et al. 2008; Zhang and Zhou 2007a; Read et al. 2009], and the most popular two are LP (Label Powerset) and BR (Binary Relevance).
(2) The algorithm adaptation methods use a straightforward strategy to solve multilabel learning problems. Most of these methods are inspired from single-label learning and can be viewed as the extensions of single-label learning algorithms. For example, the decision-tree-based method [Clare and King 2001] modifies the formula of entropy calculation and allows multiple labels at the leaves of the tree; boosting-based methods [Esuli et al. 2008; Schapire and Singer 2000] are extensions of the single-label boosting methods, and they optimize the metrics used to evaluate the performance of multilabel learning algorithms; the perceptron-based method [Crammer and Singer 2003] achieves the label ranking by maintaining one perceptron and updating the corresponding weights for each label; the RBF-Network-based method [Zhang 2009] is an adaption of the popular RBF-Network by introducing a new error function considering the multiple labels; the k-NN-based methods [Zhang and Zhou 2007b; Brinker and Hüllermeier 2007; Spyromitros et al. 2008; Cheng and Hüllermeier 2009] are the same as the traditional k-NN method in retrieving the k-nearest neighbors but make some refinements for the aggregation of multiple labels of these neighbors; and for the association-rule-based methods [Veloso et al. 2007; Thabtah et al. 2004], the consequence of the association rules they used is a set of labels instead of a single learning target label.

3.3. Metrics to Evaluate Recommendation Performance

It is hard to compare different algorithm recommendation methods. One big barrier is that there are no unified metrics to evaluate the recommendation performance. Section 2.3.3 shows that it is more reasonable to view the algorithm recommendation as a multilabel learning problem; it is natural to borrow the metrics assessing multilabel learners to evaluate the performance of algorithm recommendation.

Although all the metrics used in multilabel learning can be available theoretically, not all of them are suitable to evaluate performance of algorithm recommendation. The metrics used to evaluate the recommendation performance should be intuitive and easy to understand. Thus, in this section, based on the metrics used to evaluate multilabel learners, we define several metrics for algorithm recommendation performance evaluation as follows.

In order to facilitate understanding, we first introduce some notations. Let $D = \{d_1, d_2, \ldots, d_n\}$ denote the multilabel-based meta-data achieved from a set of n problems; each d_i corresponds to a problem p_i and is a tuple $<X_i, Y_i>(1 \leq i \leq n)$, where X_i is the meta-feature extracted from p_i, $Y_i \subseteq ASet$ presents the set of the appropriate algorithms on p_i, and $ASet = \{A_1, A_2, \ldots, A_k\}$ denotes the set of all k candidate learning algorithms. Meanwhile, given the problem p_i, let Z_i be the set of appropriate algorithms recommended by a multilabel learner for p_i.

Based on these notations, the well-known metrics *Hamming Loss*, *Precision*, *Recall*, *F-measure,* and *Accuracy*, which were proposed for performance evaluation in multilabel learning [Tsoumakas and Katakis 2007; Tsoumakas et al. 2010; Godbole and Sarawagi 2004; Schapire and Singer 2000; Read et al. 2009; Sechidis et al. 2011], are employed to evaluate recommendation models.

Definition 3.1. Hamming Loss

$$Hamming\text{-}Loss(p_i) = \frac{Z_i \Delta Y_i}{k} \quad (1 \leq i \leq n), \tag{1}$$

where Δ computes the point-to-point difference of two sets. The smaller the value of Hamming Loss, the better the recommendation.

Definition 3.2. Precision

$$Precision(p_i) = \frac{|Y_i \cap Z_i|}{|Z_i|} \quad (1 \leq i \leq n). \tag{2}$$

Definition 3.3. Recall

$$Recall(p_i) = \frac{|Y_i \cap Z_i|}{|Y_i|} \quad (1 \leq i \leq n). \tag{3}$$

The main goal for multilabel learning is to improve *Recall* without hurting *Precision*. However, *Recall* and *Precision* usually can be conflicting. This is because with an increase of the cardinal number of $|Y_i \cap Z_i|$ (i.e., improving *Recall*), there is an increase of the cardinal number of $|Z_i|$, and this will result in a reduction of *Precision*. In order to consider these two metrics at the same time, *F-measure*, which provides a balance between *Precision* and *Recall*, is employed to evaluate the performance of multilabel learning algorithms. The definition of *F-measure* is as follows:

Definition 3.4. F-measure

$$F\text{-}measure = 2 \times \frac{Recall \times Precision}{Recall + Precision}. \tag{4}$$

Definition 3.5. Accuracy

$$Accuracy(p_i) = \frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \quad (1 \leq i \leq n). \tag{5}$$

The greater the values of F-measure and Accuracy, the better the recommendation.

However, in practice, what users really care about is *whether the recommended algorithms are appropriate*. The previous metrics are not so standard for this purpose. In order to determine whether these algorithms are appropriate, the following metric *Hit* [Song et al. 2012] is defined to evaluate the performance of a recommendation model for a given problem p_i.

Definition 3.6. Hit

$$Hit(p_i) = \begin{cases} 1, & if\ Z_i \cap Y_i \neq \phi \\ 0, & otherwise \end{cases}. \tag{6}$$

According to the definition of *Hit*, $Hit = 1$ means that at least one of the recommended algorithms is appropriate for learning problem p_i. This indicates that the recommendation is effective. In contrast, $Hit = 0$ means that the recommendation does not include any appropriate algorithm, which means all the recommended algorithms do not meet the user's requirements.

With the definition of *Hit*, metric *Hit_Ratio* [Song et al. 2012] can be used to evaluate the recommendation performance over a set of learning problems PSet = $\{p_1, p_2, \ldots, p_N\}$.

Definition 3.7. Hit Ratio

$$Hit\_Ratio(\text{PSet}) = \frac{\sum_{i=1}^{N} Hit(p_i)}{N}. \tag{7}$$

The greater the value of *Hit_Ratio*, the better the recommendation method.

4. EXPERIMENTAL STUDY

In this section, we experimentally evaluate the proposed multilabel learning-based algorithm recommendation method by recommending algorithms over the benchmark learning problems.

4.1. Benchmark Dataset

In order to evaluate the reliability of the recommendation procedure and soundness of the conclusion, more learning problems should be employed. For this purpose, with the help of the problem generation method, Datasetoids, which was proposed in Soares [2009] and aimed to obtain a large number of problems for algorithm selection [Prudêncio et al. 2011a; Prudencio et al. 2011; Halabi Echeverry et al. 2012; Prudêncio et al. 2011b], we extend the 84 public UCI learning problems[4] into 1,090 (84 datasets and 1006 datasetoids) different learning problems.

The Datasetoids method is quite simple; it achieves the new learning problems by exchanging the role of each nominal attribute with that of the target concept, that is, viewing the nominal attribute as the new target concept. Table I shows the summary of the 84 problems in terms of number of features (F), number of instances (I), and number of target values (T).

4.2. Experimental Setup

In order to evaluate the performance of the proposed multilabel learning-based algorithm recommendation method, further verify whether the proposed method is potentially useful in practice, and guarantee the reproducibility of our experiments, we set the experimental study as follows.

According to the framework of algorithm recommendation (see Figure 1) in Section 2, the process of algorithm recommendation consists of three steps: meta-data collection, recommendation model construction, and algorithm recommendation for a given problem. In this part, we introduce the experimental setup following these three steps as follows:

(1) *Setup for meta-data collection*
   (a) *Meta-features used to characterize the learning problem*
   Five different types of meta-features, (1) the statistical and information-theory based, (2) the model structure based, (3) the landmarking based, (4) the problem complexity based, and (5) the structural information based, are employed to characterize different learning problems. Section 2.2 and the appendix show the details.
   (b) *Meta-target collection*
      ◇ *Candidate learning algorithms*
      In order to guarantee the generality of our experimental results, 13 algorithms based on different induction biases are employed.

[4]http://archive.ics.uci.edu/ml/datasets.html.

Table I. Summary of the 84 Learning Problems

Data ID	Data Name	F	I	T	Data ID	Data Name	F	I	T
1	anneal	38	898	6	43	mfeat-karhunen	64	2000	10
2	anneal_ORIG	38	898	6	44	mfeat-morphological	6	2000	10
3	arrhythmia	279	452	16	45	mfeat-pixel	240	2000	10
4	audiology	69	226	24	46	mfeat-zernike	47	2000	10
5	autos	25	205	7	47	molecular-biology_promoters	57	106	2
6	balance-scale	4	625	3	48	monks-problems-1_test	6	432	2
7	breast-cancer	9	286	2	49	monks-problems-1_train	6	124	2
8	breast-w	9	699	2	50	monks-problems-2_test	6	432	2
9	bridges_version1	11	105	6	51	monks-problems-2_train	6	169	2
10	bridges_version2	11	105	6	52	monks-problems-3_test	6	432	2
11	car	6	1728	4	53	monks-problems-3_train	6	122	2
12	cmc	9	1473	3	54	mushroom	22	8124	2
13	colic	22	368	2	55	nursery	8	12960	5
14	colic.ORIG	27	368	2	56	optdigits	64	5620	10
15	credit-a	15	690	2	57	page-blocks	10	5473	5
16	credit-g	20	1000	2	58	pendigits	16	10992	10
17	cylinder-bands	39	540	2	59	postoperative-patient-data	8	90	3
18	dermatology	34	366	6	60	primary-tumor	17	339	22
19	diabetes	8	768	2	61	segment	19	2310	7
20	ecoli	7	336	8	62	shuttle-landing-control	6	15	2
21	flags	28	194	8	63	sick	29	3772	2
22	glass	9	214	7	64	solar-flare_1	12	323	6
23	haberman	3	306	2	65	solar-flare_2	12	1066	6
24	hayes-roth_test	4	28	4	66	sonar	60	208	2
25	hayes-roth_train	4	132	4	67	soybean	35	683	19
26	heart-c	13	303	5	68	spambase	57	4601	2
27	heart-h	13	294	5	69	spectf_test	44	269	2
28	heart-statlog	13	270	2	70	spectf_train	44	80	2
29	hepatitis	19	155	2	71	spectrometer	102	531	48
30	hypothyroid	29	3772	4	72	spect_test	22	187	2
31	ionosphere	34	351	2	73	spect_train	22	80	2
32	iris	4	150	3	74	splice	60	3190	3
33	kdd_JapaneseVowels_test	14	5687	9	75	sponge	44	76	3
34	kdd_JapaneseVowels_train	14	4274	9	76	tae	5	151	3
35	kr-vs-kp	36	3196	2	77	tic-tac-toe	9	958	2
36	labor	16	57	2	78	trains	32	10	2
37	letter	16	20000	26	79	vehicle	18	846	4
38	liver-disorders	6	345	2	80	vote	16	435	2
39	lung-cancer	56	32	3	81	vowel	13	990	11
40	lymph	18	148	4	82	waveform-5000	40	5000	3
41	mfeat-factors	216	2000	10	83	wine	13	178	3
42	mfeat-fourier	76	2000	10	84	zoo	16	101	7

They are (1) probability-based algorithm Bayes Network; (2) tree-based algorithms C4.5, RandomTree, and RandomForest; (3) rule-based algorithms PART, Ripper, and NNge; (4) Gaussian-function-based algorithm RBFNetwork, and (5) support-vector-machine-based algorithm SMO.

Besides the aforementioned nine single learning algorithms, we also pick up the ensemble learning algorithms. The boosting algorithm is applied with the base classifiers Naive Bayes (NB) and C4.5, respectively. The bagging algorithm also runs with the same base classifiers. They are denoted as Boost+NB, Boost+C4.5, Bag+NB, and Bag+C4.5, respectively, in the experiment.

- ◇ *Metrics to evaluate the performance of the candidate learning algorithms*
  In our experiment, two kinds of metrics are employed to evaluate the performance of different algorithms on a learning problem: the frequently used single-criterion metric classification accuracy and the multicriteria metric ARR (Adjusted Ratio of Ratios) [Brazdil et al. 2003] considering both the accuracy and runtime of the algorithm. Tradeoff coefficient α used in ARR is set to 0.1%, 1%, and 10% following the suggestions in Brazdil et al. [2003] to balance the effect between accuracy and runtime, respectively.

  Moreover, in order to get a stable performance and make the best use of data, a 5×10-folds cross-validation procedure is performed to estimate these metrics. That is, when evaluating each learning algorithm on each problem, the 10-fold cross-validation procedure is repeated five times. This procedure helps control the variation due to the different choices of the training and test datasets [Pizarro et al. 2002]. And the performance metrics estimated by this procedure can further be used in multiple comparison procedure for finding out the appropriate learning algorithms of a given problem.
- ◇ *Multiple comparison procedure to identify appropriate algorithms for a learning problem*
  For each problem, in order to identify the appropriate algorithms, the nonparametric test-based multiple comparison procedure, the Friedman test [Friedman 1937], followed by the Holm procedure test [Hommel 1988], is conducted at the significance level 0.05. This is due to the fact that (1) the distribution of the estimated performance metric is usually unknown and it is difficult to guarantee the performance metric satisfying the normality distribution and the homogeneity of variance, and (2) the nonparametric tests are distribution free [Pizarro et al. 2002; Demšar 2006] and more powerful in detecting the differences among the candidate algorithms.

(2) *Recommendation model construction*

(a) *Algorithms used to construct the recommendation model*
According to Section 2.3, there are three different kinds of meta-data in terms of the form of meta-target: single label, ranking, and multilabel. For different forms of meta-target, the algorithms used to construct recommendation models are different as well.

In order to demonstrate whether the multilabel learning algorithms are competitive in algorithm recommendation model construction, we compare the performance of the recommendation models constructed by multilabel learning algorithms with that of the models constructed by single-label learning and ranking-based learning algorithms. The following instance-based learning algorithms are employed on the meta-data with different forms of meta-target.

- ◇ For the meta-data whose meta-target is algorithm ranking, the learning algorithms used to construct recommendation models are usually instance based [Brazdil et al. 2003; Brazdil and Soares 2000; Brazdil et al. 1994; Nakhaeizadeh and Schnabl 1997]. Thus, in our experiment, we employ the well-known k-NN algorithm, which has been used for algorithm recommendation in Brazdil et al. [2003] and Song et al. [2012] as a benchmark. And the number of the nearest neighbors k is set to 1, 5, and 10 and 10% of the number of instances in the training dataset as suggested in Brazdil et al. [2003] and Song et al. [2012].
- ◇ For the meta-data with a multilabeled meta-target, we use the well-known k-NN-based multilabel learning algorithm MLkNN [Zhang and Zhou 2007b]. To make a fair comparison, the number of nearest neighbors used in these

algorithms is set to 1, 5, and 10 and 10% of the number of instances in the training dataset as well.

- ◇ For the meta-data with a single-labeled meta-target, we also employ the k-NN-based algorithms with $k = 1$, 5, and 10 and 10% of the number of instances in the training dataset.

In addition to these recommendation methods, recently, Lee and Giraud-Carrier [2013] proposed a behavior-clustering-based algorithm recommendation method that performs recommendation on the clusters of algorithms rather than individual algorithms. And the clusters of algorithms can be viewed as a multilabel form; thus, the recommendation method is close and quite relevant to our method. Therefore, the method in Lee and Giraud-Carrier [2013] is also selected as a benchmark method being compared with ours. The parameter used to cluster algorithms in this recommendation method is set according to the suggestions in Lee and Giraud-Carrier [2013]. Note that Lee and Giraud-Carrier's method clusters the algorithms directly according to the metrics computed from the differences of the algorithms' output and only focuses on recommending the most accurate learning algorithms and does not take into consideration the runtime of a learning algorithm. Moreover, their method does not give any guideline about how to cluster algorithms while taking the runtime into account as well. Therefore, in this article, we compare the method in Lee and Giraud-Carrier [2013] with ours only under the specific situation where we need to recommend the algorithms with high classification accuracy.

(b) *Comparison between different recommendation methods*

We compare our proposed multilabel learning-based recommendation method with other recommendation methods in terms of the four metrics *Hamming Loss*, *F-measure*, *Accuracy*, and *Hit Ratio* defined in Section 3.3. These metrics are calculable when the recommendation results are represented in the form of a set of algorithms.

However, as the ranking-based recommendation method provides a ranking list of the candidate algorithms, it does not tell us how many algorithms should be recommended due to the fact that the number of appropriate algorithms on different problems is different. It is difficult to evaluate this kind of method in terms of these metrics. In this case, we usually need to predefine a threshold to identify a set of algorithms from the ranking list as the recommended algorithms. However, there does not exist any effective way to preassign this threshold. In order to handle this problem and make a fair comparison, supposing t algorithms are recommended by the multilabeled learning-based recommendation method, we extract the top t algorithms of the ranking list as the recommended algorithms and then compute the four metrics on these t algorithms to evaluate the performance of the algorithm-ranking-based recommendation method.

For the single-label learning-based recommendation method, the recommendation result is a single algorithm. That is, there is only one element in the recommended algorithm set. In this case, according to the definitions of *Hamming Loss*, *F-measure,* and *Accuracy*, it is innate that the value of *Hamming Loss* will be high and the values of *F-measure* and *Accuracy* will be low when they are employed to evaluate only one recommended algorithm. Thus, it is unfair to compare the single recommended algorithm with a set of algorithms with multiple elements in terms of these three metrics.

However, we can evaluate whether the single recommended algorithm is really appropriate by the metric *Hit Ratio*. Therefore, in our experiments, the

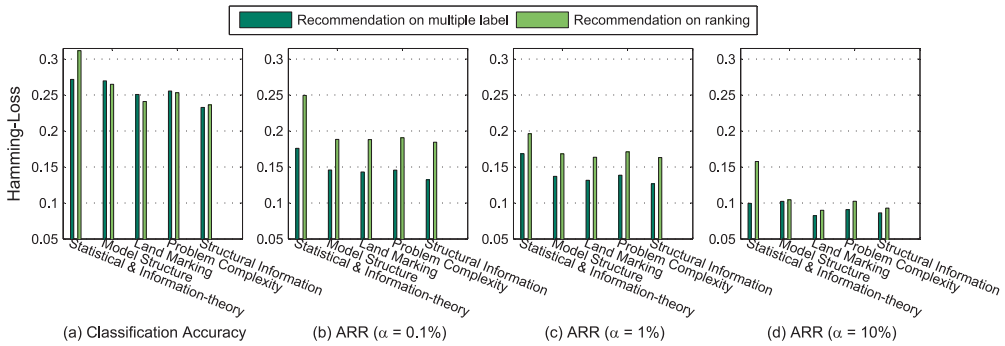

Fig. 4. Comparison of different recommendation methods in terms of Hamming Loss.

single-label learning-based recommendation method is compared with our proposed recommendation method only in terms of *Hit Ratio*.

(3) *Procedure to estimate the performance of algorithm recommendation*
The *jackknife* procedure is employed to evaluate the performance of algorithm recommendation. That is, we view each of the 1,090 problems as new learning problems and the remaining 1,089 problems as historical problems. Then, we recommend the appropriate algorithms for the new problem based on the recommendation model constructed on the historical problems.

4.3. Experimental Results and Analysis

This section reports the experimental results. First, we compare the multilabel learning-based recommendation method to the existing ones in Section 4.3.1. Then, we analyze the effect of the number of nearest neighbors on MLkNN for algorithm recommendation in Section 4.3.2.

4.3.1. Recommendation Performance Comparison. In this section, we compare the proposed multilabel learning-based algorithm recommendation method to the existing algorithm recommendation methods in terms of the metrics *Hamming Loss*, *F-measure*, *Accuracy*, and *Hit Ratio*, respectively. Here, we present the recommendation results with the number of the nearest neighbors $k = 5$ under which the recommendation methods show better recommendation performance on all the single-label, ranking, and multilabel-based meta-datasets.

(1) *Comparison on Hamming Loss*
Figure 4 compares the recommendations on multilabel-based meta-data and those on ranking-based meta-data in terms of Hamming Loss. The four subfigures respectively correspond to the comparison results when we evaluate the performance of candidate learning algorithms in terms of classification accuracy,[5] ARR with $\alpha = 0.1\%$, ARR with $\alpha = 1\%$, and ARR with $\alpha = 10\%$. Moreover, each subfigure shows the comparison results over five different kinds of meta-features. The same representations are considered in Figures 5, 6, and 7.

As we know, the metric Hamming Loss indicates how far the recommendation is from the set of really appropriate algorithms. The smaller the *Hamming Loss*, the better the recommendation. From this figure, we can observe that:

(a) Hamming Loss of the recommendation on multilabel-based meta-data is less than that on ranking-based meta-data under all the five different kinds of

[5]Classification accuracy is equivalent to an ARR with $\alpha = 0$.

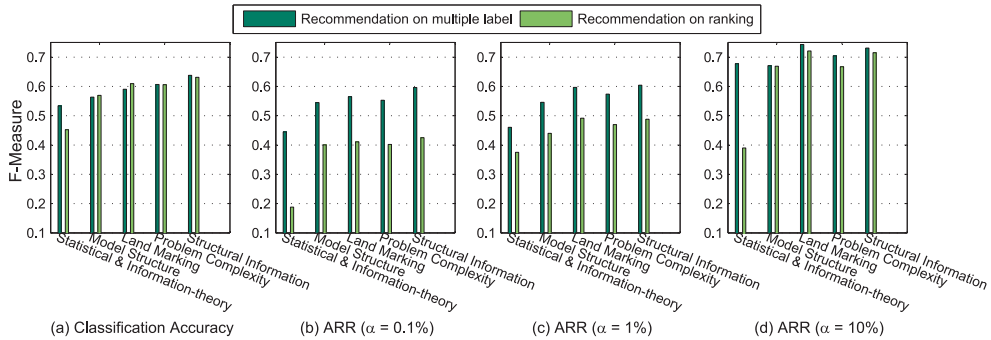

Fig. 5. Comparison of different recommendation methods in terms of F-measure.

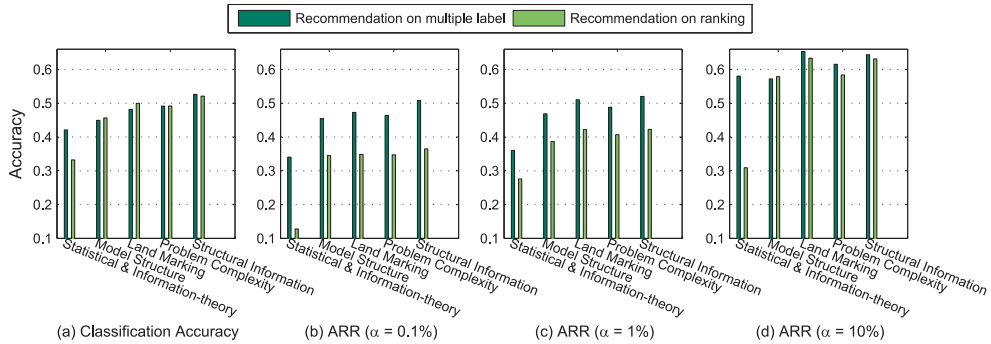

Fig. 6. Comparison of different recommendation methods in terms of Accuracy.

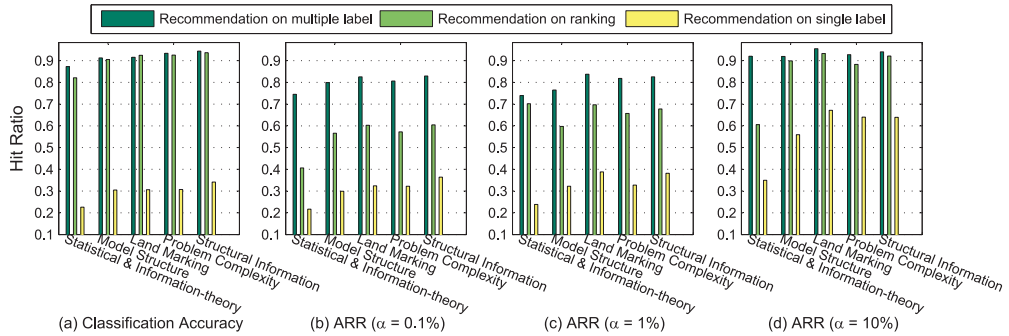

Fig. 7. Comparison of different recommendation methods in terms of Hit Ratio.

meta-features when the meta-target is in terms of ARR with $\alpha = 0.1\%$, 1%, and 10% (see Figures 4(b), (c), and (d)).

(b) For the classification accuracy-based meta-target (see Figure 4(a)), although Hamming Loss achieved on multilabel-based meta-data is not always less than that achieved on ranking-based meta-data under all five kinds of meta-features, the minimum Hamming Loss is still achieved on the multilabel-based meta-data when meta-features are structural information based.

(c) When we evaluate the performance of the candidate algorithms in terms of ARR, the value of α in ARR represents the amount of accuracy the user is willing trade for speedup of the algorithm. The greater value of α means that the user intends to find the more efficient algorithm. From these subfigures, we can find that with an increase of α, Hamming Loss decreases. This is because, with an increase of α, the number of really appropriate algorithms on a learning problem usually decreases (see Figure 2). Moreover, an algorithm is recommended as an appropriate one if and only if it is really appropriate. Thus, the number of recommended algorithms will decrease as well. As a result, the number of algorithms in the set, which is achieved by Δ (see Definition 3.1) between the set of really appropriate algorithms and the set of recommended algorithms, will decrease. Thus, according to Definition 3.1 of Hamming Loss, Hamming Loss decreases.

(2) *Comparison on F-measure*

Figure 5 compares the recommendation results between multilabel-based meta-data and ranking-based meta-data in terms of F-measure. Different from Hamming Loss, the greater the value of F-measure, the better the recommendation. From this figure, we can observe the following:

(a) The F-measure of the recommendation on multilabel-based meta-data is greater than that on ranking-based meta-data under all the five different kinds of meta-features when the meta-target is in terms of ARR with $\alpha = 0.1\%$, 1%, and 10% (see Figures 5(b), (c), and (d)).

(b) For the classification accuracy-based meta-target (see Figure 5(a)), the F-measure of the recommendation on multilabel-based meta-data is not always greater than that on ranking-based meta-data for all the five kinds of meta-features; for example, it is smaller for the meta-features extracted based on "Model structure" and "Landmarking." However, the maximum F-measure is still achieved on the multilabel-based meta-data when meta-features are structural information based.

(3) *Comparison on Accuracy*

Figure 6 depicts the recommendation comparisons between multilabel-based meta-data and ranking-based meta-data in terms of Accuracy.

From this figure, we can observe that the comparison results in terms of Accuracy are similar with those in terms of F-measure in Figure 5. That is, Accuracy of the recommendation on multilabel-based meta-data is greater than that on ranking-based meta-data under all the five different kinds of meta-features when the meta-target is in terms of ARR with $\alpha = 0.1\%$, 1%, and 10% (see Figures 6(b), (c), and (d)). However, for the classification-accuracy-based meta-target, the maximum Accuracy is still achieved on the multilabel-based meta-data when meta-features are structural information based (see Figure 6(a)).

(4) *Comparison on Hit Ratio*

Figure 7 compares the Hit Ratio of the recommendation on multilabel-based meta-data with those on ranking-based and single-label-based meta-data. From this figure, we can observe the following:

(a) The Hit Ratio of the recommendation on multilabel-based meta-data is at least more than 70%. The greatest Hit Ratio is up to 90% when the meta-target is in terms of classification accuracy and ARR with $\alpha = 10\%$ (see Figures 7(a) and (d)), and 80% when the meta-target is ARR with $\alpha = 0.1\%$ and ARR with $\alpha = 1\%$ (see Figure 7(b) and (c)). These high Hit Ratios indicate that the multilabeled meta-data can effectively depict the relationship between characteristics of the learning problems (i.e., the meta-features) and performance of the learning algorithms (i.e., the meta-target).

(b) No matter under which pair of meta-target and meta-features, the Hit Ratio of the recommendation on single-label-based meta-data is the lowest. This is because the single-labeled meta-data assumes there is only one appropriate algorithm for a given dataset. The appropriate algorithm of the given dataset is identified as the one with the highest performance metric (e.g., classification accuracy) estimated by cross-validation procedure. And this excludes the other candidate algorithms whose performance has no statistical difference with the highest performance. Moreover, recommendation models that are constructed on the single-labeled meta-data require that the recommended algorithm is just the algorithm with the highest performance on the given dataset. That is, the recommended algorithm does not hit even its performance is approximately equal to and has no significant difference with the highest performance. This is a much stricter condition compared with the definition of hit on recommendations of multilabel learning/algorithm-ranking-based recommendation models. Therefore, the Hit Ratio of the recommendations on single-labeled meta-data is lower.

Moreover, the Hit Ratio of the recommendation on multilabel-based meta-data is greater than that on ranking-based meta-data under all the pairs of meta-target and meta-features, except for one case where the meta-target is in terms of classification accuracy and meta-features are extracted based on landmarking. However, in this case, the difference of the Hit Ratio is quite small.

According to these analyses, we can conclude that the recommendation performance on the multilabel-based meta-data is superior to or competes with that on the ranking/single-label-based meta-data.

In order to further explore whether or not the improvements are statistically significant, the nonparametric Wilcoxon signed-rank test, which is suggested in Demšar [2006] to compare two methods (i.e., multilabel based vs. ranking based/single-label based) on multiple problems, is conducted at significance level 0.05. And then we give the statistical test results in terms of p-values and corresponding analysis.

(5) *Statistical Test Results*

Table II shows the p-values obtained by Wilcoxon signed-rank test on comparing the multilabel learning-based recommendation method with the existing methods in terms of *Hamming Loss* (see Table II(a)), *F-measure* (see Table II(b)), *Accuracy* (see Table II(c)), and *Hit Ratio* (see Table II(d)). The alternative hypothesis is that the recommendations on the multilabel-based meta-data are statistically better than those on the ranking or single-label-based meta-data in terms of each of the four recommendation performance metrics.

In these tables, a p-value < 0.05 indicates that the alternative hypothesis is valid; that is, the multilabel learning-based recommendation method is statistically better than the compared methods. This is marked by the symbol "+."

On the contrary, when the p-value is greater than 0.05, the multilabel learning-based recommendation method would be either statistically equal to or worse than the compared methods (i.e., algorithm ranking and single-label learning-based recommendation methods). In this case, in order to further distinguish the later two conclusions, when the average performance of the multilabel learning-based recommendation method is smaller than that of the compared method, the Wilcoxon signed-rank test is performed, and the corresponding alternative hypothesis is that the multilabel learning-based recommendation method is statistically worse than the compared methods in terms of each of the four recommendation performance metrics. The p-values that are underlined are the results of these tests.

Table II. Statistical Test Results on Comparison Between Recommendations on Different Meta-Data

(a) Statistical test in terms of *Hamming Loss*

Alternative Hypothesis	Meta-features	Meta-target			
		Acc	ARR($\alpha = 0.1\%$)	ARR($\alpha = 1\%$)	ARR($\alpha = 10\%$)
Multilabel > Ranking	Statistical & information theory	0.00+	0.00+	0.00+	0.00+
	Model structure	0.60	0.00+	0.00+	0.65
	Landmarking	0.17	0.00+	0.00+	0.22
	Problem complexity	0.80	0.00+	0.00+	0.04+
	Structural information	0.45	0.00+	0.00+	0.48
	Win/Draw/Loss	1/4/0	5/0/0	5/0/0	2/3/0

(b) Statistical test in terms of *F-measure*

Alternative Hypothesis	Meta-features	Meta-target			
		Acc	ARR($\alpha = 0.1\%$)	ARR($\alpha = 1\%$)	ARR($\alpha = 10\%$)
Multilabel > Ranking	Statistical & information theory	0.00+	0.00+	0.00+	0.00+
	Model structure	0.52	0.00+	0.00+	0.76
	Landmarking	0.13	0.00+	0.00+	0.17
	Problem complexity	0.92	0.00+	0.00+	0.03+
	Structural information	0.82	0.00+	0.00+	0.41
	Win/Draw/Loss	1/4/0	5/0/0	5/0/0	2/3/0

(c) Statistical test in terms of *Accuracy*

Alternative Hypothesis	Meta-features	Meta-target			
		Acc	ARR($\alpha = 0.1\%$)	ARR($\alpha = 1\%$)	ARR($\alpha = 10\%$)
Multilabel > Ranking	Statistical & information theory	0.00+	0.00+	0.00+	0.00+
	Model structure	0.49	0.00+	0.00+	0.70
	Landmarking	0.13	0.00+	0.00+	0.17
	Problem complexity	0.93	0.00+	0.00+	0.03+
	Structural information	0.69	0.00+	0.00+	0.42
	Win/Draw/Loss	1/4/0	5/0/0	5/0/0	2/3/0

(d) Statistical test in terms of *Hit Ratio*

Alternative Hypothesis	Meta-features	Meta-target			
		Acc	ARR($\alpha = 0.1\%$)	ARR($\alpha = 1\%$)	ARR($\alpha = 10\%$)
Multilabel > Ranking	Statistical & information theory	0.00+	0.05+	0.01+	0.00+
	Model structure	0.55	0.00+	0.00+	0.10
	Landmarking	0.43	0.00+	0.00+	0.03+
	Problem complexity	0.45	0.00+	0.00+	0.00+
	Structural information	0.47	0.00+	0.00+	0.08
	Win/Draw/Loss	1/4/0	5/0/0	5/0/0	3/2/0
Multilabel > Single-label	Statistical & information theory	0.00+	0.00+	0.00+	0.00+
	Model structure	0.00+	0.00+	0.00+	0.00+
	Landmarking	0.00+	0.00+	0.00+	0.00+
	Problem complexity	0.00+	0.00+	0.00+	0.00+
	Structural information	0.00+	0.00+	0.00+	0.00+
	Win/Draw/Loss	5/0/0	5/0/0	5/0/0	5/0/0

⋆ "Acc" is the abbreviation of "classification accuracy."

Moreover, the row "Win/Draw/Loss" records the number of cases (i.e., meta-features) under which the multilabel learning-based recommendation method is statistically better than/equal to/worse than the compared recommendation method in terms of a recommendation metric. From these tables, we can observe the following:

(a) According to the row "Win/Draw/Loss" in these tables, no matter under which pair of meta-features and meta-target, the recommendations on the multilabel-based meta-data are (1) either statistically better than or statistically equal to those on ranking-based meta-data in terms of all the four metrics *Hamming Loss*, *F-measure*, *Accuracy,* and *Hit Ratio* and (2) statistically better than those on single-label-based meta-data in terms of *Hit Ratio*.

(b) For both of the meta-targets based on classification accuracy and ARR with $\alpha = 10\%$, the number of "Draw" is not zero when we compare recommendations on the multilabel-based meta-data and those on the ranking-based meta-data in terms of the four recommendation performance metrics. This means that there are no significant differences between the multilabel learning-based recommendation method and the algorithm-ranking-based one under some kinds of meta-features, such as the recommendations under "Model structure"-based meta-features.

The recommendation method on ranking-based meta-data just predicts a ranking list of the candidate algorithms. The algorithms with the top ranks should be considered as the appropriate at first. However, not all the candidate algorithms are applicable, and it could not tell us the exact number of algorithms being recommended. Moreover, there is still no any effective method to preassign the number of recommended algorithms picked up from a ranking list in advance. Thus, in practice, it is difficult for us to choose a set of algorithms from this ranking list as the recommended ones. On the contrary, the recommendations on multilabel-based meta-data expressly output a set of algorithms as the recommended ones. Thus, from the angle of practicality, the multilabel -based meta-data is better.

(6) *Comparison with Algorithm-Clustering-Based Recommendation Method*

The algorithm-clustering-based recommendation method [Lee and Giraud-Carrier 2013] recommends a cluster of algorithms instead of individual algorithms, and the cluster can also be viewed as multilabels. This is closest to our proposed multilabel learning-based recommendation method. In this part, we present the comparison results between these two methods.

According to the experimental setup in Section 4.2, the algorithm-clustering-based recommendation method focuses on picking up the most accurate algorithms for a given problem and does not take into consideration the runtime of the algorithm. Thus, the comparison between these two methods is performed only under the situation where we evaluate the performance of the learning algorithm by classification accuracy.

Table III shows the comparison results between these two methods in terms of Hamming Loss, F-measure, Accuracy, and Hit Ratio, respectively. In this table, the columns "multi" and "cluster" represent the average recommendation performance of multilabel learning- and algorithm-clustering-based recommendation methods. Meanwhile, we also give the Wilcoxon test results between between these two methods in terms of a given recommendation metric at the significant level $\alpha = 0.05$. The column "p" shows the p-value of the test. And the mark "+" following the p-value indicates that the multilabel learning-based recommendation method is statistically better than the algorithm-clustering-based one. The row "Win/Draw/Loss" summarizes the number of different meta-features under which the multilabel

Table III. Comparison of Recommendations Achieved by Multilabel Learning and Algorithm Clustering

	Hamming Loss			F-Measure			Accuracy			Hit Ratio		
Meta-features	Multi	Cluster	p	Multi	Cluster	p	Multi	Cluster	p	Multi	Cluster	p
Statistical & information theory	0.2717	0.3663	0+	0.5377	0.4729	0+	0.4209	0.3367	0+	0.8725	0.8862	0.3241
Model structure	0.2697	0.3665	0+	0.5635	0.4689	0+	0.4492	0.3328	0+	0.9128	0.8927	0.2712
Landmarking	0.2507	0.3665	0+	0.5905	0.4689	0+	0.4816	0.3328	0+	0.9156	0.8927	0.0690
Problem complexity	0.2555	0.3717	0+	0.6066	0.4621	0+	0.4914	0.3263	0+	0.9339	0.9037	0.0097+
Structural information	0.2327	0.3625	0+	0.6381	0.4719	0+	0.5261	0.3348	0+	0.9440	0.9092	0.0018+
Win/Draw/Loss		5/0/0			5/0/0			5/0/0				2/3/0

learning-based recommendation method is statistically better/equal to/worse than the algorithm-clustering-based one in terms of a given recommendation metric.

From this table, we can observe that, under all the five different kinds of meta-features, the multilabel learning-based recommendation method is statistically better than the algorithm-clustering-based recommendation method in terms of Hamming Loss, F-measure, and Accuracy. But when comparing them in terms of Hit Ratio, the algorithm-clustering-based recommendation method is statistically equal to the proposed multilabel learning-based method under three kinds of meta-features. Moreover, for the other two kinds of meta-features, although they are statistically different, the difference is not very great. This indicates that the algorithm-clustering-based method can recommend the appropriate algorithms but introduces more improper algorithms when compared to the proposed multilabel learning-based method. This might be because the clusters of algorithms are precalculated. These clusters are static. But in practice, the number of appropriate algorithms will vary with different datasets. The precalculated clusters cannot capture this variation. This demonstrates that the new proposed multilabel learning-based method is more flexible.

In summary, these experiment results and analyses demonstrate that the recommendation model constructed on multilabel-based meta-data is more effective and practical. This means that the multilabel-based meta-data can be used to explore the relationship between the characteristics of learning problems and the performance of learning algorithms.

4.3.2. Effect of the Number of Nearest Neighbors on MLkNN. In our experiment, we employ the instance-based multilabel learning algorithm MLkNN to construct the recommendation model. Generally, a different number of the nearest neighbors k will lead to different nearest neighbor datasets and further the different recommendations.

Thus, in this section, we analyze the impact of the parameter k on the recommendation results and further provide the guideline to choose appropriate k in practical applications.

For this purpose, first, all the possible nearest neighbor settings are tested. That is, the number of the nearest neighbors k is set from 1 to the number of the historical learning problems minus 1 (e.g., 1,089 in our experiment) when identifying the neighbors for a given learning problem. Then, we summarize the average recommendation performance metrics (e.g., *Hamming Loss*, *F-measure, Accuracy,* and *Hit Ratio*) with respect to the number of nearest neighbors k over the 1,090 historical learning problems under different kinds of meta-targets. Here, for each performance metric and a given k, the average performance is the mean of the recommendation performance over five different kinds of meta-features.

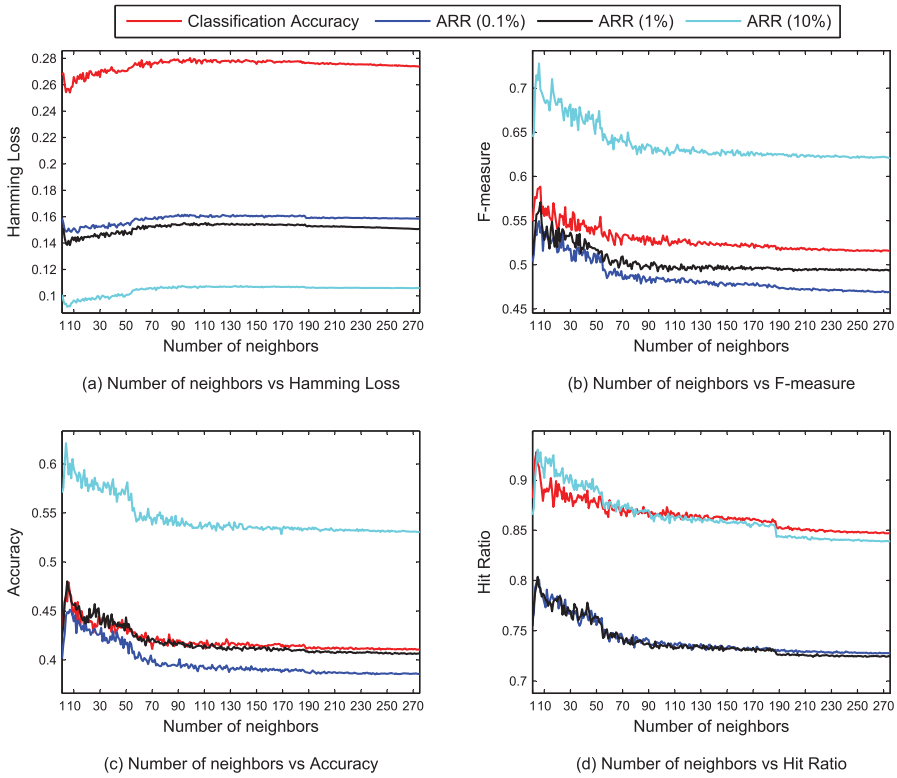

Fig. 8. Sensitive analysis of the number of neighbors on the recommendation performance.

Figure 8 shows the relationship between the number of neighbors and the recommendation performance. The four subfigures present the effect of the number of nearest neighbors k on *Hamming Loss*, *F-measure, Accuracy,* and *Hit Ratio,*respectively. In each subfigure, the four lines with different color correspond to the performance metrics achieved on the meta-data whose meta-target is in terms of classification accuracy, ARR with $\alpha = 0.1\%$, ARR with $\alpha = 1\%$, and ARR with $\alpha = 10\%$.

It is observed that the figure does not list all the 1,089 possible numbers of the nearest neighbors but only up to 270. The reason is that when the number of the nearest number k is greater than 270, with an increase of k, the average recommendation performance either becomes stable or slowly decreases. That is, the recommendation results under these larger k ($>$270) carry quite little information about selection of the appropriate k and thus are not shown in the figure. From this figure, we can observe the following:

(1) For each of the recommendation performance metrics, the number of the nearest neighbors does affect the average recommendation performance under all the four meta-targets.
(2) For all four recommendation performance metrics, their variation tendencies under different meta-targets are similar. With the increment of k, the average recommendation performance first reaches the best point with fluctuations, then worsens and becomes stable. Especially, for *Hamming Loss* (see Figure 8(a)), where a lower value means a better recommendation, it first reaches the lowest point and then increases; and for each of the other three metrics (see Figures 8(b), (c), and (d)),

where a higher value means a better recommendation, it first reaches the highest point and then decreases.

(3) For all four different meta-targets, the average recommendation performance reaches the best point very quickly with the increase of the number of the nearest neighbors k in terms of all four recommendation performance metrics, although there are fluctuations before and after the best point, and a greater k cannot improve the recommendation but seems to make it worse. This reveals that there is a small k under which the average recommendation performance reaches the optimum point. The optimum point lies between 5 and 10. Therefore, in order to achieve a better recommendation by MLkNN, the number of nearest neighbors should fall into the range of 5 to 10.

5. CONCLUSION

In this article, we first distinguished different recommendation methods by two dimensions: meta-features and meta-target. The meta-features are divided into five categories: statistical and information-theory based, model structure based, landmarking based, problem complexity based, and structural information based; meta-target is grouped into three categories: single-label form, ranking form, and multilabel form. The first two kinds of meta-target have been used in practice, but for the last one, as far as we know, there is still no recommendation method explicitly employing multilabel-based meta-data to describe the relationship between the characteristics of problems and the relative performance of learning algorithms, although the multilabel-based meta-data is a more natural and applicable form.

Inspired by this idea, we proposed a novel multilabel learning-based algorithm recommendation method. Moreover, with the help of the measures commonly used in evaluating the performance of multilabel learning algorithms, we presented several metrics that can be used to uniformly evaluate the performance of different recommendation methods.

Finally, we evaluated our new multilabel learning-based recommendation method. Experiments were conducted over 1,090 benchmark learning problems, 13 different learning algorithms, and five different kinds of meta-features. Results show that the proposed multilabel-based meta-learning is more effective.

APPENDIX

In this appendix, we introduce the five different kinds of meta-features for algorithm recommendation:

(1) Statistical and information-theory-based measures (see Table IV)
(2) Model-structure-based measures (see Table V)
(3) Landmarking-based measures following the suggestions in Pfahringer et al. [2000] and Bensusan and Giraud-Carrier [2000]; the following six classifiers are selected as the landmark learners: (1) Naive Bayes, (2) 1-NN (Nearest Neighbor), (3) Elite 1-NN, (4) a decision node learner, (5) a random chosen node learner, and (6) the worst node learner, where the last three learners can be achieved based on the well-known learning algorithm C4.5.
(4) Problem-complexity-based measures (See Table VI)
(5) Structural-information-based measures First, the two feature vectors, one-item feature vector and two-item feature vector, are extracted from the given problem. These two vectors consist of the frequencies of one-item sets and two-item sets, respectively. Afterward, the *minimum*, *1/8 quantile*, *2/8 quantile*, *3/8 quantile*, *4/8 quantile*, *5/8 quantile*, *6/8 quantile*, *7/8 quantile,* and *maximum* are computed for these two vectors and form the final set of dataset characteristics.

Table IV. Statistical and Information-Theory-Based Measures

Measures	Definitions
Ins.Num	Number of instances
Attr.Num	Number of Attributes
Target.Num	Number of target concept values
Target.Min	Proportion of minority target
Target.Max	Proportion of majority target
Pro.Bin	Proportion of binary attributes
Pro.Nom	Proportion of nominal attributes
Pro.Num	Proportion of numeric attributes
Pro.MissIns	Proportion of instances with missing values
Pro.MissValues	Proportion of missing values
Mean.Geo	Geometric mean
Mean.Harm	Harmonic mean
Mean.Trim	Trim mean excluding the highest and lowest 5%
Mad	Mean absolute deviation
Var	Variance
Std	Standard deviation
Prcitile	Percentile 75%
Int.Range	Interquartile range
Prop.AttrWithOutlier	Proportion of numerical attributes with outliers over all numerical attributes
Skewness	Skewness of data based on numerical attributes
Kurtosis	Kurtosis of data based on numerical attributes
Max.eig	Maximum eigenvalue
Min.eig	Minimum eigenvalue
Can.corr	Canonical correlation
Grav.cent	Center of gravity
MeanAbsCoef	Mean absolute coefficient of attribute pairs
$H(C)$	Entropy of classes
$\bar{H}(X)$	Mean entropy of nominal attributes
$\bar{M}(C, X)$	Mean mutual information of classes and attributes based on nominal attributes
En.attr	Equivalent number of attributes $H(C)/\bar{M}(C, X)$
Ns.ratio	Noise-signal ratio $\bar{H}(X)/\bar{M}(C, X) - 1$

Table V. Model-Structure-Based Measures

Measures	Definitions
Tree.Height	Height of tree (also referred as to number of levels in tree)
Tree.Wdith	Width of tree
Node.Num	Number of nodes in tree
Leaf.Num	Number of leaves in tree
Level.Max	Maximum number of nodes at one level
Level.Mean	Mean of the number of nodes on levels
Level.Dev	Standard deviation of the number of nodes on levels
Branch.Long	Length of the longest branch
Branch.Short	Length of the shortest branch
Branch.Mean	Mean of the branch lengths
Branch.Dev	Standard deviation of the branch lengths
Attr.Min	Minimum occurrence of attributes
Attr.Max	Maximum occurrence of attributes
Attr.Mean	Mean of the number of occurrences of attributes
Attr.Dev	Standard deviation of the number of occurrences of attributes

Table VI. Problem-Complexity-Based Measures

Measures	Definitions
Bound.Len	Length of class boundary
Adherence.Prop	Proportion of retained adherence subsets
Intra/Inter.Ratio	Ratio of average intra/interclass nearest neighbors
NN.Nonlinerity	Nonlinearity of Nearest Neighbors classifier
Linear.Nonlinerity	Nonlinearity of linear classifier
Fisher.Ratio	Maximum Fisher's discriminant ratio
Ins/Attr	Training set size relative to feature space dimensionality

ACKNOWLEDGMENTS

The authors are grateful to the contributions of the two anonymous reviewers and editor, which greatly improved the article.

REFERENCES

D. W. Aha. 1992. Generalizing from case studies: A case study. In *Proceedings of the 9th International Conference on Machine Learning*. Morgan Kaufmann, Citeseer, 1–10.

A. Aiguzhinov, C. Soares, and A. Serra. 2010. A similarity-based adaptation of naive Bayes for label ranking: Application to the metalearning problem of algorithm recommendation. In *Discovery Science*. Springer, 16–26.

S. Ali and K. A. Smith. 2006. On learning algorithm selection for classification. *Applied Soft Computing* 6, 2 (2006), 119–138.

S. Ali and K. A. Smith-Miles. 2006. A meta-learning approach to automatic kernel selection for support vector machines. *Neurocomputing* 70, 1–3 (2006), 173–186.

B. Arinze, S. L. Kim, and M. Anandarajan. 1997. Combining and selecting forecasting models using rule based induction. *Computers & Operations Research* 24, 5 (1997), 423–433.

H. Bensusan. 1998. God doesn't always shave with Occam's Razor—learning when and how to prune. In *Proceedings of the 10th European Conference on Machine Learning*. Springer, 119–124.

H. Bensusan and C. Giraud-Carrier. 2000. Casa batlo is in passeig de gracia or landmarking the expertise space. In *Proceedings of the ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*. 29–47.

E. Bernadó-Mansilla and T. K. Ho. 2005. Domain of competence of XCS classifier system in complexity measurement space. *IEEE Transactions on Evolutionary Computation* 9, 1 (2005), 82–104.

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

P. Brazdil, J. Gama, and B. Henery. 1994. Characterizing the applicability of classification algorithms using meta-level learning. In *Proceedings of European Conference on Machine Learning*. Springer, 83–102.

P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta. 2009. *Meta-Learning: Applications to Data Mining*. Springer.

P. Brazdil and C. Soares. 2000. A comparison of ranking methods for classification algorithm selection. In *Proceedings of 11th European Conference on Machine Learning.* 63–75.

P. B. Brazdil, C. Soares, and J. P. Da Costa. 2003. Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. *Machine Learning* 50, 3 (2003), 251–277.

K. Brinker and E. Hüllermeier. 2007. Case-based multilabel ranking. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, Vol. 707.

C. E. Brodley. 1993. Addressing the selective superiority problem: Automatic algorithm/model class selection. In *Proceedings of the Tenth International Conference on Machine Learning*. Citeseer, 17–24.

W. Chen, J. Yan, B. Zhang, Z. Chen, and Q. Yang. 2007. Document transformation for multi-label feature selection in text categorization. In *Proceedings of the 7th IEEE International Conference on Data Mining*. IEEE, 451–456.

W. Cheng and E. Hüllermeier. 2009. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning* 76, 2 (2009), 211–225.

A. Clare and R. King. 2001. Knowledge discovery in multi-label phenotype data. *Principles of Data Mining and Knowledge Discovery* (2001), 42–53.

W. J. Conover and R. L. Iman. 1981. Rank transformations as a bridge between parametric and nonparametric statistics. *American Statistician* (1981), 124–129.

W. J. Conover and R. L. Iman. 1982. Analysis of covariance using the rank transformation. *Biometrics* (1982), 715–724.

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

M. C. P. De Souto, R. B. C. Prudencio, R. G. F. Soares, D. S. A. de Araujo, I. G. Costa, T. B. Ludermir, and A. Schliep. 2008. Ranking and selecting clustering algorithms using a meta-learning approach. In *IEEE International Joint Conference on Neural Networks*. IEEE, 3729–3735.

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

R. P. W. Duin, E. Pekalska, and D. M. J. Tax. 2004. The characterization of classification problems by classifier disagreements. In *Proceedings of the 17th International Conference on Pattern Recognition*, Vol. 1. IEEE, 140–143.

C. W. Dunnett. 1955. A multiple comparison procedure for comparing several treatments with a control. *Journal of the American Statistical Association* 50, 272 (1955), 1096–1121.

D. A. Elizondo, R. Birkenhead, M. Gamez, N. Garcia, and E. Alfaro. 2009. Estimation of classification complexity. In *Proceedings of International Joint Conference on Neural Networks*. IEEE, 764–770.

R. Engels and C. Theusinger. 1998. Using a data metric for preprocessing advice for data mining applications. In *Proceedings of the European Conference on Artificial Intelligence*. 430–434.

A. Esuli, T. Fagni, and F. Sebastiani. 2008. Boosting multi-label hierarchical text categorization. *Information Retrieval* 11, 4 (2008), 287–313.

M. Friedman. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association* 32, 200 (1937), 675–701.

J. Fürnkranz and E. Hüllermeier. 2010. *Preference Learning*. Springer-Verlag, New York.

J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, and K. Brinker. 2008. Multilabel classification via calibrated label ranking. *Machine Learning* 73, 2 (2008), 133–153.

J. Gama and P. Brazdil. 2000. Cascade generalization. *Machine Learning* 41, 3 (2000), 315–343.

S. Godbole and S. Sarawagi. 2004. Discriminative methods for multi-labeled classification. *Advances in Knowledge Discovery and Data Mining* (2004), 22–30.

H. Guo. 2003. *Algorithm selection for sorting and probabilistic inference: a machine learning-based approach*. PhD dissertation, Kansas State University.

A. Halabi Echeverry, D. Richards, and A. Bilgin. 2012. Identifying characteristics of seaports for environmental benchmarks based on meta-learning. *Knowledge Management and Acquisition for Intelligent Systems* (2012), 350–363.

R. J. Henery. 1994. *Methods for comparison*. Ellis Horwood, Upper Saddle River, NJ, USA, 107–124. http://dl.acm.org/citation.cfm?id=212782.212789

M. Hilario and A. Kalousis. 2001. Fusion of meta-knowledge and meta-data for case-based model selection. *Principles of Data Mining and Knowledge Discovery* (2001), 180–191.

T. K. Ho. 2000. Complexity of classification problems and comparative advantages of combined classifiers. *Multiple Classifier Systems* (2000), 97–106.

T. K. Ho and M. Basu. 2002. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, 3 (2002), 289–300.

G. Hommel. 1988. A stagewise rejective multiple test procedure based on a modified Bonferroni test. *Biometrika* 75, 2 (1988), 383–386.

E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker. 2008. Label ranking by learning pairwise preferences. *Artificial Intelligence* 172, 16–17 (2008), 1897–1916.

A. K. Jain, R. P. W. Duin, and J. Mao. 2000. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 1 (2000), 4–37.

A. Kalousis. 2002. *Algorithm selection via meta-learning*. PhD dissertation, University of Geneve.

A. Kalousis, J. Gama, and M. Hilario. 2004. On data and algorithms: Understanding inductive performance. *Machine Learning* 54, 3 (2004), 275–312.

A. Kalousis and M. Hilario. 2000. Model selection via meta-learning: A comparative study. In *Proceedings of the 12th IEEE International Conference on Tools with Artificial Intelligence*. IEEE, 406–413.

A. Kalousis and T. Theoharis. 1999. NOEMON: Design, implementation and performance results of an intelligent assistant for classifier selection. *Intelligent Data Analysis* 3, 5 (1999), 319–337.

I. Katakis, G. Tsoumakas, and I. Vlahavas. 2008. Multilabel text classification for automated tag suggestion. In *Procedings of the ECML/PKDD-08 Workshop on Discovery Challenge*.

R. D. King, C. Feng, and A. Sutherland. 1995. Statlog: Comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence an International Journal* 9, 3 (1995), 289–333.

C. Köpf, C. Taylor, and J. Keller. 2000. Meta-analysis: From data characterisation for meta-learning to meta-regression. In *Proceedings of the PKDD-00 Workshop on Data Mining, Decision Support, Meta-Learning and ILP*. Citeseer.

J. W. Lee and C. Giraud-Carrier. 2013. Automatic selection of classification learning algorithms for data mining practitioners. *Intelligent Data Analysis* 17, 4 (2013), 665–678.

G. Lindner and R. Studer. 1999. AST: Support for algorithm selection with a CBR approach. *Principles of Data Mining and Knowledge Discovery* (1999), 418–423.

E. L. Mencía and J. Furnkranz. 2008. Pairwise learning of multilabel classifications with perceptrons. In *Proceedings of IEEE International Joint Conference on Neural Networks*. IEEE, 2899–2906.

D. Michie, D. J. Spiegelhalter, and C. C. Taylor. 1994. Machine learning, neural and statistical classification. *Ellis Horwood Series in Artificial Intelligence* (1994).

G. Nakhaeizadeh and A. Schnabl. 1997. Development of multi-criteria metrics for evaluation of data mining algorithms. In *Proceedings of 3rd International Conference on Knowledge Discovery and Data Mining*. 37–42.

Y. Peng, P. Flach, C. Soares, and P. Brazdil. 2002. Improved dataset characterisation for meta-learning. In *Discovery Science*. Springer, 193–208.

B. Pfahringer, H. Bensusan, and C. Giraud-Carrier. 2000. Meta-learning by landmarking various learning algorithms. In *Proceedings of the 17th International Conference on Machine Learning*. Morgan Kaufmann, 743–750.

J. Pizarro, E. Guerrero, and P. L. Galindo. 2002. Multiple comparison procedures applied to model selection. *Neurocomputing* 48 (2002), 155–173.

R. Prudêncio, M. de Souto, and T. Ludermir. 2011a. Selecting machine learning algorithms using the ranking meta-learning approach. *Meta-Learning in Computational Intelligence* (2011), 225–243.

R. Prudêncio, C. Soares, and T. Ludermir. 2011b. Combining meta-learning and active selection of datasetoids for algorithm selection. *Hybrid Artificial Intelligent Systems* (2011), 164–171.

R. B. C. Prudencio, C. Soares, and T. B. Ludermir. 2011. Uncertainty sampling methods for selecting datasets in active meta-learning. In *The 2011 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1082–1089.

J. Read, B. Pfahringer, G. Holmes, and E. Frank. 2009. Classifier chains for multi-label classification. *Machine Learning and Knowledge Discovery in Databases* (2009), 254–269.

J. R. Rice. 1976. The algorithm selection problem. *Advances in Computers* 15 (1976), 65–118.

S. L. Salzberg. 1997. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery* 1, 3 (1997), 317–328.

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

K. Sechidis, G. Tsoumakas, and I. Vlahavas. 2011. On the stratification of multi-label data. *Machine Learning and Knowledge Discovery in Databases* (2011), 145–158.

K. A. Smith-Miles. 2008. Cross-disciplinary perspectives on meta-learning for algorithm selection. *Computing Surveys* 41, 1 (2008), 1–25.

C. Soares. 2009. UCI++: Improved support for algorithm selection using datasetoids. *Advances in Knowledge Discovery and Data Mining* (2009), 499–506.

R. Soares, T. Ludermir, and F. De Carvalho. 2009. An analysis of meta-learning techniques for ranking clustering algorithms applied to artificial data. *Artificial Neural Networks* (2009), 131–140.

S. Y. Sohn. 1999. Meta analysis of classification algorithms for pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21, 11 (1999), 1137–1144.

Q. B. Song, G. T. Wang, and C. Wang. 2012. Automatic recommendation of classification algorithm based on data set characteristics. *Pattern Recognition* 45, 7 (2012), 2672–2689.

E. Spyromitros, G. Tsoumakas, and I. Vlahavas. 2008. An empirical study of lazy multilabel classification algorithms. *Artificial Intelligence: Theories, Models and Applications* (2008), 401–406.

E. Spyromitros Xioufis, G. Tsoumakas, and I. Vlahavas. 2011. Multi-label learning approaches for music instrument recognition. *Foundations of Intelligent Systems* (2011), 734–743.

F. A. Thabtah, P. Cowling, and Y. Peng. 2004. MMAC: A new multi-class, multi-label associative classification approach. In *Proceedings of the 4th IEEE International Conference on Data Mining*. IEEE Computer Society, 217–224.

L. E. Toothaker. 1993. *Multiple Comparison Procedures*. Vol. 89. Sage.

G. Tsoumakas and I. Katakis. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining* 3, 3 (2007), 1–13.

G. Tsoumakas, I. Katakis, and I. Vlahavas. 2010. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*. 667–685.

G. Tsoumakas and I. Vlahavas. 2007. Random k-labelsets: An ensemble method for multilabel classification. In *Proceedings of the 18th European Conference on Machine Learning*. Springer, 406–417.

A. Veloso, W. Meira, M. Gonçalves, and M. Zaki. 2007. Multi-label lazy associative classification. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'07)*. Springer, 605–612.

R. Vilalta and Y. Drissi. 2002. A perspective view and survey of meta-learning. *Artificial Intelligence Review* 18, 2 (2002), 77–95.

D. H. Wolpert. 2001. The supervised learning no-free-lunch theorems. In *Proceedings of 6th Online World Conference on Soft Computing in Industrial Applications*. Citeseer, 25–42.

J. Yang and B. Jiu. 2006. Algorithm selection: A quantitative approach. *Algorithmic Trading II: Precision, Control, Execution. Institutional Investor, Inc.* 26–34.

M. Zhang and Z. H. Zhou. 2007a. Multi-label learning by instance differentiation. In *Proceedings of the 22th National Conference on Artificial Intelligence*, Vol. 22. AAAI Press 2007, 669–674.

M. L. Zhang. 2009. ML-RBF: RBF neural networks for multi-label learning. *Neural Processing Letters* 29, 2 (2009), 61–74.

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

Received June 2013; revised March 2014; accepted March 2014