Aggregating Independent and Dependent Models to Learn Multi-label Classifiers*

Elena Montañés, José Ramón Quevedo, and Juan José del Coz

Artificial Intelligence Center, University of Oviedo at Gijón (Spain)
{elena,quevedo,juanjo}@aic.uniovi.es

Abstract. The aim of multi-label classification is to automatically obtain models able to tag objects with the labels that better describe them. Despite it could seem like any other classification task, it is widely known that exploiting the presence of certain correlations between labels helps to improve the classification performance. In other words, object descriptions are usually not enough to induce good models, also label information must be taken into account. This paper presents an aggregated approach that combines two groups of classifiers, one assuming independence between labels, and the other considering fully conditional dependence among them. The framework proposed here can be applied not only for multi-label classification, but also in multi-label ranking tasks. Experiments carried out over several datasets endorse the superiority of our approach with regard to other methods in terms of some evaluation measures, keeping competitiveness in terms of others.

1 Introduction

In multi-label classification the goal is to induce a hypothesis to assign a set of labels for each instance rather than a single class, as happens in multi-class classification. This kind of tasks arises in many practical domains; nowadays almost all media contents (text documents, songs, movies or videos) are tagged with several labels to briefly inform users about their actual content. Another well-known example in the research community is the keywords attached to a paper; useful to indicate the *relevant* topics of the paper.

At first sight, one could think that multi-label classification can be easily solved applying or adapting state-of-the-art (binary or multi-class) classification algorithms. In fact, many of the first approaches proposed were aimed to extend these methods to handle multi-label data, including decision trees [2], instance-based algorithms [22], Neural Networks [21], Support Vector Machines [6], Naive Bayes [12], Conditional Random Fields [9] and boosting [16]. However, in order to obtain good performance results, it is not enough to adapt a good learning approach, otherwise it is also necessary to design specific methods that exploit somehow the particularities of multi-label data, as most of the cited works do.

* This research has been partially supported by Spanish Ministerio de Ciencia e Innovación (MICINN) grant TIN2008-06247.

D. Gunopulos et al. (Eds.): ECML PKDD 2011, Part II, LNAI 6912, pp. 484–500, 2011.

Mainly, multi-label learning presents two challenging problems. The first one bears on the computational complexity of the algorithms. If the number of labels is high, then a very complex approach is not practical for business use, so the scalability is a key issue in this field. The second dare is related with the own nature of this task and the multi-label data. Not only the number of classes is higher than in multi-class classification, but also each example belongs to an indeterminate number of labels, and more important, labels present some relationships between them. In other words, object descriptions are usually not enough to induce correct models, also the information among labels must be taken into account. The dimensionality of the label space, together with the possibility of incomplete labeling data obtained from different sources, make this goal even more difficult to achieve.

Despite the first issue is important to make algorithms applicable in large domains, from a learning perspective probably the hottest topic in multi-label community is to design new methods able to detect and exploit dependences among labels. In fact, several methods are being proposed in that direction. Roughly speaking, we can categorize them according to two major properties: i) the size of the subset of labels whose dependences are searched for; and ii) depending on the type of correlations they try to find. Looking at the first property, we have those methods that only consider pairwise relations between labels [6,7,13,16,21], and, secondly, approaches that take into account correlations among the labels in bigger subsets [14,15,19], including those that consider the influence of all other labels in the prediction of one particular label [1,10]. On the other hand, concerning the type on dependences they seek to capture [3], there are some methods designed to detect conditional label dependence (refered to the dependence of the labels given a specific instance), for example [3,9,15,18]; and unconditional dependence (a global type of dependence independent of any concrete observation), see [1,10,21].

The main proposal of this paper is grounded on the following idea. Thinking again about the keywords example, all of us know that, for a particular paper, there are some clearly relevant and irrelevant keywords. But there are also a few keywords than can be considered relevant or irrelevant, depending on the authors' opinion. Different authors with different criteria could select some different keywords for the same paper. Our assumption is that labels that are clearly relevant or clearly irrelevant can be predicted using only the description of the object. However, considering the relationships with other labels is also necessary to correctly assign the somehow-relevant labels. This is the reason why this paper presents a decomposition approach based on aggregating two groups of classifiers: the first one is learned assuming label independence, and the second one is built considering a complete label dependence. These dependent classifiers are designed to capture conditional dependence, taking into account the influence of all other labels. Another interesting property of the approach presented here is that it can be easily adapted to cope with different multi-label learning tasks, not only classification, but also ranking.

The rest of the paper is organized as follows. Next section describes a formal framework for multi-label classification and reviews previous approaches related with this work. Section 3 exposes the proposals of this paper, based on the idea of combining two groups of binary models. Finally, experimental results are reported in Section 4 and some conclusions are drawn in Section 5.

2 Notation and Related Work

2.1 Formal Framework for Multi-label Classification

Let $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_m\}$ be a finite and non-empty set of labels, and let $\mathcal{X}$ be an input space. We consider a multi-label classification task given by a training set $S = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)\}$, whose instances were independently and randomly obtained from an unknown probability distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ on $\mathcal{X} \times \mathcal{Y}$, in which the output space $\mathcal{Y}$ is the power set of $\mathcal{L}$, in symbols $\mathcal{P}(\mathcal{L})$. In order to make the notation easier to read, we define $\boldsymbol{y}_i$ as a binary vector, $\boldsymbol{y}_i = \{y_1, y_2, \ldots, y_m\}$, in which each component $y_j = 1$ indicates the presence of label ℓ_j in the set of relevant labels of $\boldsymbol{x}_i$. Using this convention, the output space can be also defined as $\mathcal{Y} = \{0, 1\}^m$.

The goal of a multi-label classification is to induce a hypothesis $\boldsymbol{h} : \mathcal{X} \longrightarrow \mathcal{Y}$ from S, that correctly predicts the subset of labels from $\mathcal{L}$ for a new unlabeled instance $\boldsymbol{x}$. Without any loss of generality, this hypothesis can be seen as a combination of a collection of sub-hypotheses, $\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots h_m(\boldsymbol{x}))$, one per label, in which each h_j takes the form of

$$h_j : \mathcal{X} \longrightarrow \{0, 1\}, \tag{1}$$

and it is able to predict if the label ℓ_j must be attached to the instance $\boldsymbol{x}$ or not. Sometimes, the goal is not to return the relevant labels, but to obtain the posterior probability of each label given $\boldsymbol{x}$. In that case, $h_j : \mathcal{X} \longrightarrow [0, 1]$. Notice that these probabilistic classifiers can also be useful to rank the labels according to their posteriors. This paper only discusses some multi-label classifiers of the form of Equation 1, despite many of them can have a probabilistic/ranking version, including ours.

In order to measure the performance of multi-label classifiers, several metrics have been proposed. A unified presentation of existing evaluation measures for multi-label classification can be found in [18], including their categorization into example-based, label-based and ranking-based measures. Here we will consider only example-based evaluation metrics for three reasons: i) this paper focuses on multi-label classification rather than multi-label ranking, moreover, some of the state-of-the-art methods employed to compare do not produce a ranking of labels, ii) we are interested in studying whether the different approaches capture or not the dependencies among labels at example-level, and iii) some of the example-based metrics used were originally proposed in [10], that also introduces a stacking-based method for multi-label learning, which is probably the most similar approach to ours. The following evaluation measures have been taken from the Information Retrieval field:

– **Jaccard index** computes the percentage of relevant labels predicted in the subset formed by the union of returned and relevant labels[1],

$$Jaccard(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{\sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} [\![y_i = 1 \text{ or } h_i(\boldsymbol{x}) = 1]\!]}. \quad (2)$$

– **Precision** determines the fraction of relevant labels in the predicted labels,

$$Precision(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{\sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} [\![h_i(\boldsymbol{x}) = 1]\!]}. \quad (3)$$

– **Recall** is the proportion of relevant labels of the example correctly predicted,

$$Recall(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{\sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} [\![y_i = 1]\!]}. \quad (4)$$

– $\boldsymbol{F_1}$ is the evenly weighted harmonic mean of Precision and Recall,

$$F_1(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{2 \sum_{i=1}^{m} [\![y_i = 1 \text{ and } h_i(\boldsymbol{x}) = 1]\!]}{\sum_{i=1}^{m} ([\![y_i = 1]\!] + [\![h_i(\boldsymbol{x}) = 1]\!])}. \quad (5)$$

The evaluation metrics presented above are biased towards those methods that correctly predict the relevant labels. That is one of the reasons to select them, because they allow us to study if relevant labels are detected or not, specially in those situations where some kind of correlation occurs. Finally, the performance in multi-label classification can be reported in terms of other two measures:

– **Hamming loss**, which is defined as the proportion of labels whose relevance is incorrectly predicted:

$$Hamming(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{1}{m} \sum_{i=1}^{m} [\![y_i \neq h_i(\boldsymbol{x})]\!]. \quad (6)$$

– **0/1 loss**, looks if predicted and relevant label subsets are equal or not.

$$Zero - One(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = [\![\boldsymbol{y} \neq \boldsymbol{h}(\boldsymbol{x})]\!]. \quad (7)$$

2.2 Some Approaches for Multi-label Classification

The most employed baseline method for multi-label classification is the Binary Relevance (BR) algorithm. BR decomposes the learning of $\boldsymbol{h}$ into a set of binary classification tasks, one per label, where each single model h_j is learned independently of the rest, using only the information of that particular label and ignoring all other labels. Despite its simplicity, BR algorithm presents several advantages: i) any binary learning method can be taken as base learner, ii) it has linear complexity with respect to the number of labels and iii) it can be easily parallelized. The main drawback of BR is that it does not take into account

[1] The expression $[\![p]\!]$ evaluates to 1 if the predicate p is true, and to 0 otherwise.

any label dependences and may fail to predict some label combinations if such correlations are present. However, using a state-of-the-art base learner, for instance SVM, with a proper tuning parameters process, BR usually obtains quite good results in benchmark datasets of the literature. Moreover, for some particular evaluation metrics, as Hamming loss (Eq. 6), BR offers a very competitive performance. This behavior can be explained studying BR from a probabilistic point of view. As in most classification learning process, each binary model h_j is able to estimate $\mathbf{P}(y_j|\boldsymbol{x})$. This is the reason why BR is well-suited for every loss function whose risk minimizer can be expressed in terms of marginal distributions of labels, for instance Hamming loss. On the other hand, the fact that BR does not take label dependence into account obviously affects its performance for evaluation measures like 0/1 loss. Hence, it is necessary to estimate the joint label probability distributions to obtain predictions that minimize this sort of metrics. A formal probabilistic analysis of multi-label classification, studying the connection between risk minimization and loss functions can be found in [3,4].

Godbole and Sharawagi present one approach [10] to overcome the label-independence problem of BR. They apply the stacked generalization learning paradigm [20], also known simply as stacking, in the context of multi-label classification. In the learning phase, their method builds a stack of two groups of classifiers. The first one is formed by the same binary classifiers yielded by BR method, in symbols, $\boldsymbol{h}^1(\boldsymbol{x}) = (h_1^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x}))$. In a second level, also called meta-level, another group of binary models (one for each label again) is learned, but these classifiers consider an augmented feature space that includes the binary outputs of all models of the first level, $\boldsymbol{h}^2(\boldsymbol{x}, \boldsymbol{y}') = (h_1^2(\boldsymbol{x}, \boldsymbol{y}'), \ldots, h_m^2(\boldsymbol{x}, \boldsymbol{y}'))$, where $\boldsymbol{y}' = \boldsymbol{h}^1(\boldsymbol{x})$. The idea is to learn the relationships between labels in the meta-level step. In the testing phase, the final predictions are the outputs of the meta-level classifiers, $\boldsymbol{h}^2(\boldsymbol{x})$, using the outputs of $\boldsymbol{h}^1(\boldsymbol{x})$ exclusively to obtain the values of the augmented feature space.

Some variants of the stacking approach have been proposed, mainly focused on reducing the augmented feature space removing some label dimensions. The idea is to ignore the information of those labels that are no related with the label j of the model h_j^2 being learned. For instance, in [17] the authors propose to calculate the chi-coefficient between each pair of labels, (j, k), based on an initial single pass over the training set. The method prunes the information of each label k with a correlation below a threshold to induce the meta-model h_j^2. This approach improves the computational efficiency, without any significant loss in predictive performance, even some gains are obtained for some data sets.

In any case, the meta-level classifiers of the stacking approach estimates $\mathbf{P}(y_j|\boldsymbol{x}, \boldsymbol{y}')$, where $\boldsymbol{y}'$ is in turn an estimation that depends only on $\boldsymbol{x}$. This chain of estimations can explain why perhaps $\boldsymbol{y}'$ does not contain enough information to infer the dependence of label y_j with respect to other labels.

Read et al. describe [15] a learning algorithm called Classifier Chain (CC), that can model label correlations while maintaining a computational complexity of the same order as that of BR. As its name denotes, CC involves m binary classifiers linked along a chain, where each classifier deals with the binary relevance problem

associated with one label. In the training phase, the feature space of each classifier in the chain is extended with the actual label information of all previous links. For instance, if the chain follows the order of the set of labels, then the functional form of each classifier h_j will be:

$$h_j : \mathcal{X} \times \{0,1\}^{j-1} \longrightarrow \{0,1\}, \tag{8}$$

in which the actual label data of the previous labels in the chain, $y_1, \ldots, y_{j-1}$, are used to build h_j. Notice that all binary models can be learned in parallel. However, in the testing phase, the classifiers are applied following the chain order, using the binary outputs of the previous models as additional input information. In symbols, $\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}, h_1(\boldsymbol{x})), h_3(\boldsymbol{x}, h_1(\boldsymbol{x}), h_2(\boldsymbol{x}, h_1(\boldsymbol{x}))), \ldots)$. Obviously, the label order in the chain affects the performance obtained. Although heuristics can be used to select a promising chain order, the authors solve the issue by an Ensemble of Classifier Chains (ECC). This approach ensembles different random chain orderings and a different sample of the training data for learning each CC model.

Dembczyński et al. present in [3] a probabilistic framework for multi-label classification and propose the Probabilistic Classifier Chains (PCC) and its ensemble version (EPCC). They generalize and outperform their counterpart methods, CC and ECC, but increasing their testing complexity. One of the interesting contributions of the paper is that it offers a probabilistic interpretation of CC. Given an instance $\boldsymbol{x}$ it is possible to compute the conditional probability of each label combination $\boldsymbol{y} \in \mathcal{Y}$, applying the product rule of probability:

$$\mathbf{P}(\boldsymbol{y}|\boldsymbol{x}) = \mathbf{P}(y_1|\boldsymbol{x}) \prod_{j=2}^{m} \mathbf{P}(y_j|\boldsymbol{x}, y_1, \ldots, y_{j-1}), \tag{9}$$

whose probabilities can be obtained from the classifiers of the chain (Eq. 8) when a probabilistic learner is used. The difference between PCC and CC is that the former estimates the entire joint distribution of labels, whereas the later takes sequential decisions and, in that sense, it offers a deterministic approximation of PCC. PCC produces much better estimates but at the cost of higher complexity, limiting its applicability to data sets with a small number of labels.

There are other methods, less related with our approach, that try to find interdependences between labels. RAkEL(RAndom k-labELsets), presented by Tsoumakas and Vlahavas [19], iteratively constructs an ensemble of Label Powerset (LP) classifiers. LP algorithm considers each unique subset of labels that exists in a multi-label training set as one of the classes of a new multi-class classification task. At each iteration, RAkEL randomly selects a k-labelset $\boldsymbol{Y}_i$ from $\mathcal{L}$ without replacement. Then, it learns a LP classifier of the form $\mathcal{X} \rightarrow \mathcal{P}(\boldsymbol{Y}_i)$. A simple voting process determines the final classification set. Also, Cheng and Hüllermeier propose IBLR (Instance-Based Learning by Logistic Regression) [1]. IBLR unifies instance-based learning and logistic regression, comprising both methods as special cases. Considering only the label dependence problem, the main idea is to extend the description of each example $\boldsymbol{x}$ by additional features that express the presence or absent of each label in the neighborhood of $\boldsymbol{x}$.

3 Aggregating Independent and Dependent Classifiers

The main proposal of this paper is to build a multi-label classifier that combines the two main options to tackle multi-label learning. On one hand, there are methods based on the assumption of label independence, that is, they only use object descriptions in order to predict the labels attached to the object, as BR. On the other hand, there are plenty of algorithms that induce models considering some kind of label dependence, that is, they also employ the information about other labels, like those described in the previous section. Formally, the latter approach can encapsulate the former, in the sense that a label-dependent model can also capture the cases that an independent model predicts well. However, when the interdependences among labels are complex, learning reliable dependent models becomes more difficult: richest hypothesis spaces must be used, increasing the risk of overfitting, and more labeled data is needed, which will not be available in some cases.

In our opinion, the two approaches are not exclusive, but complementary. Hence, aggregating them may produce a more robust multi-label classifier. This is the reason to propose Aggregating Independent and Dependent classifiers (AID), a decomposition method that combines two groups of models. The first one is learned assuming label independence and it will be formed by the same classifiers yielded by BR method or by the first-level models of the stacking-based approach [10], $\boldsymbol{h}^1(\boldsymbol{x}) = (h_1^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x}))$. The second group of binary classifiers, $\boldsymbol{h}^2(\boldsymbol{x}, \boldsymbol{y}) = (h_1^2(\boldsymbol{x}, y_2, \ldots, y_m), \ldots, h_m^2(\boldsymbol{x}, y_1, \ldots, y_{m-1}))$, is built considering the information of all other labels. Thus, each h_j^2 is defined as:

$$h_j^2 : \mathcal{X} \times \{0,1\}^{m-1} \longrightarrow \{0,1\}. \tag{10}$$

These classifiers try to detect fully conditional label dependence. Notice that all models can be learned in parallel, because they are induced using only training data. However, in the testing phase, the classifiers of the first group, $\boldsymbol{h}^1$, are applied first and their binary outputs form the label features of models $\boldsymbol{h}^2$. The final prediction is calculated aggregating both groups of responses:

$$\begin{aligned} \boldsymbol{h}(\boldsymbol{x}) = \oplus(& (h_1^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x})), \\ & (h_1^2(\boldsymbol{x}, h_2^1(\boldsymbol{x}), \ldots, h_m^1(\boldsymbol{x})), \ldots, h_m^2(\boldsymbol{x}, h_1^1(\boldsymbol{x}), \ldots, h_{m-1}^1(\boldsymbol{x}))) \ ), \end{aligned} \tag{11}$$

in which $\oplus$ can be selected by practitioners, depending on the target loss function and on the specific learning task. A natural choice for multi-label classification is the $or()$ function, being $max()$ for ranking whenever $\boldsymbol{h}^1$ and $\boldsymbol{h}^2$ provide probabilities. From a probabilistic point of view, AID method merges two different estimations $\mathbf{P}(y_j|\boldsymbol{x})$ and $\mathbf{P}(y_j|\boldsymbol{x}, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_m)$ for label y_j. In this sense AID takes into account the conditional dependence between labels.

Let us highlight that our method is quite general and can be adapted and improved in several directions. We will cite three of them: i) the aggregate function $\oplus$ can be selected (or even learned) to optimize a specific target loss function, ii) for a given query, the dependent models $\boldsymbol{h}^2$ can be iteratively applied until not

new labels are assigned, as it is pointed out in [10], and iii) some dimensions of the label data can be removed in order to make the learning of models $\boldsymbol{h}^2$ easier, for instance, applying methods like [17]. Due to the lack of space, the study of all these issues are beyond the scope of this work.

3.1 Comparison with Related Approaches

In order to better understand the properties of our method, it is interesting to analyze the differences with respect to the most related approaches [3,10,15] described above. We support the idea that AID solves two drawbacks of the stacking-based approach [10]. Firstly, the independent models learned in the first level are only employed to obtain the label-related features for training and testing the meta-level classifiers. In other words, they are not used as predictors, when it is well-known that BR classifiers by themselves can obtain a relatively good performance. The outputs of independent classifiers, once they are calculated, can be additionally employed to decide the predicted labels. Secondly, and even more important, maybe some information about the dependence among labels is missing when learning the meta-level classifiers. Instead of using the actual labels of each example to augment the feature space, as AID classifier does, the stacking method employs first-level classifiers predictions. Although it is absolutely formal from a learning perspective–the data source is the same in both training and the testing phases for meta-level classifiers–, if we think about the trueness of the training data, the actual labels are less noisy and contain the true correlation information among the labels. For this reason the estimation of $\mathbf{P}(y_j|\boldsymbol{x}, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_m)$ is expected to be more accurate than $\mathbf{P}(y_j|\boldsymbol{x}, \boldsymbol{y}')$. Next section experimentally analyzes these two issues.

Comparing our method with Classifier Chain [15], including its probabilistic variant (PCC) [3], we can found pros and cons for both. One of the best properties of CC approaches, specially PCC, is that they are supported by probability theory. PCC is able to estimate the entire joint distribution of the labels and can select the most appropriate label combination for a particular loss function, whereas CC and our approach offer only greedy approximations. But, on the other hand, CC variants assume an in-chain dependence between labels when sometimes their interdependences are much more complex. Moreover, some probability estimations of the chain can be poor when the correlated labels of a label are not placed before in the chain, which most likely happens at the first links of the chain. This is the reason for ensembling several CC or PCC classifiers. Our approach considers for every label all others, so the correlated labels are always taken into account, although detecting their dependences can be more difficult.

Analyzing the training computational complexity, all the approaches incur in the same linear complexity with respect to the number of labels, but they differ in the number of models, CC/PCC (m), stacking method and AID ($2m$), and ECC/EPCC (Nm, where N is the number of ensembles). The testing complexity of AID, the stacking method and CC/ECC is linear again, differing in the number of evaluations requiered. PCC/EPCC evaluate an exponential order times the models limiting their applicability to domains with few labels.

Table 1. Properties of the data sets used in the experiments

Data set	Attributes	Examples	Labels	Cardinality
bibtex	1836	7395	159	2.40
emotions	72	593	6	1.87
enron	1001	1702	53	3.38
genbase	1185	662	27	1.25
image	135	2000	5	1.24
mediamill	120	5000	101	4.27
medical	1449	978	45	1.25
reuters	243	7119	7	1.24
scene	294	2407	6	1.07
slashdot	1079	3782	22	1.18
yeast	103	2417	14	4.24

4 Experiments

This section reports the results of the experiments performed to evaluate the proposed multi-label classification method. The aim of the experiments was twofold. Firstly, a deep comparative study between aggregating-based and stacking-based methods was performed. The idea was to experimentally analyze the different properties of both kind of strategies. Secondly, our aggregating approach was compared with some other state-of-the-art methods for multi-label classification, most of them aimed to detect correlations among labels. The experiments were performed over several multi-label data sets whose main properties are shown in Table 1. As it can be seen, they are quite different among them in the number of attributes, examples, labels and cardinality (number of labels per example).

We tested two groups of multi-label classification algorithms. In the first place, AID classifier (using *or*() as the aggregate function) and stacking-based approach, denoted as STA, were compared. We also included a couple of variants of both to study some of their properties. Then, our aggregating approach was compared with BR, MLkNN [22], RAkEL [19], IBLR [1] and ECC [15], in the version described in [3], named as ECC*. Among chain-based methods, we selected ECC* because it offers the best trade-off between performance and complexity [3]. CC performs worse and PCC/EPCC have a higher computational complexity.

The binary base learner employed to obtain single classifiers for each label was the *logistic regression* of [11]. The regularization parameter C was established for each binary model performing a grid search over the values $C \in \{10^p \mid p \in [-3, \ldots, 3]\}$ optimizing the accuracy estimated by means of a balanced 2-fold cross validation repeated 5 times. Such parameter settings have only sense for BR, ECC* and AID, and kept equal for all of these methods. That is, their models are exactly the same when their respective input spaces are equal. The parameters taken for the rest of the state-of-the-art methods were the default ones suggested by their authors.

The evaluation measures applied were those discussed in Section 2.1. As they are defined on a per instance basis, the value for a test set is the average over all instances. The scores reported, displayed as percentages for all measures, were estimated by means of a 10-fold cross-validation. The ranks of each data sets are indicated in brackets. In case of ties, average ranks are shown. The average ranks over all data sets are computed and shown at the last row of each table. Following the recommendations of [5] and [8], a two-step comparison for each of the considered measures was performed. The first step is a Friedman test that rejects the null hypothesis that states that not all learners perform equally. The second step is a *post-hoc* pairwise comparison. We performed a Bergmann-Hommel's test using the software provided in [8]. This comparison is preferred to that of Nemenyi [5], because it is a less conservative procedure able to detect certain obvious differences that Nemenyi's test may not obtain. In any case, the significant differences found with both tests are almost equal.

4.1 AID Classifier vs. Stacking Approach

The goal of this experiment was to compare AID classifier and STA method and also to gain some insights about their principles. This was the motivation to include two variants of both. Table 2 and Table 3 show the scores of all them; BR was added as the baseline reference. Our main intention was to obtain answers to the following questions: i) Which method performs better, AID or STA? ii) Which information is more appropriate to detect the correlations among labels, the actual label data or the predictions of independent models? iii) In the testing phase, which data is preferable to augment the feature space of dependent models, binary or probabilistic outputs? And, finally, iv) is preferable to aggregate or to stack the predictions of independent and dependent models?

The first question is the easiest to answer because it only involves two of the algorithms. AID classifier performs better than STA in all the measures except in Hamming loss. The differences are significant in the cases of Recall, F_1 and Jaccard index (see Table 6). In fact, our proposed method ranks first in five out the six performance measures, while STA is not in the top-three for any of them. Comparing both with the baseline method (BR), AID obtains better scores in all metrics (significant differences in Recall, F_1 and Jaccard index) except in Hamming loss, in which BR performs significantly better. In the comparison between STA and BR, STA wins in four out of the six (except Hamming loss and Precision), but the differences are quite small and none of them are significant.

In order to answer the second question we included two new methods. The idea was to feed AID classifier and STA with the training information that uses the other one to learn the dependent models, $\boldsymbol{h}^2$. Following this idea, AID$^{\boldsymbol{y}'}$ classifier uses the predicted labels given by independent models $\boldsymbol{h}^1$, and STA$^{\boldsymbol{y}}$ employs the true label data. Comparing now each algorithm with its counterpart (AID vs. AID$^{\boldsymbol{y}'}$, STA vs. STA$^{\boldsymbol{y}}$), we find that in both cases the algorithm that uses the actual label information (AID or STA$^{\boldsymbol{y}}$) improves the performance of its counterpart in all measures, but Hamming loss. The differences are significant in the case of F_1 and Jaccard index for AID vs. AID$^{\boldsymbol{y}'}$. Interestingly, now AID$^{\boldsymbol{y}'}$

Table 2. Aggregated vs. stacking-based approaches: Precision, Recall, F_1 and Jaccard index

Precision	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	48.19(3)	48.70(2)	47.17(7)	48.01(4.5)	48.01(4.5)	**49.08**(1)	47.39 (6)
Emotions	56.36(6)	**62.30**(1)	62.09(2)	57.99(4)	56.58(5)	59.03(3)	54.31 (7)
Enron	**69.99**(1)	66.10(3)	65.51(4)	65.05(6)	65.18(5)	66.37(2)	61.69 (7)
Genbase	99.52(4)	99.52(4)	**99.60**(1.5)	99.40(6.5)	99.40(6.5)	99.52(4)	**99.60** (1.5)
Image	44.23(7)	**53.97**(1)	53.88(2)	44.49(6)	44.54(5)	48.18(3)	46.17 (4)
Mediamill	**78.81**(1)	70.91(3)	70.11(5)	44.06(6)	43.49(7)	70.16(4)	71.67 (2)
Medical	78.94(6)	**82.50**(1)	81.33(2)	79.47(4.5)	79.47(4.5)	80.88(3)	78.31 (7)
Reuters	85.79(7)	**87.53**(1)	87.50(2)	85.88(6)	85.89(5)	87.48(3)	86.39 (4)
Scene	61.46(7)	67.81(3)	66.14(6)	66.88(5)	67.13(4)	**71.89**(1)	69.05 (2)
Slashdot	46.06(6)	**53.32**(1)	53.20(2)	47.91(3.5)	47.91(3.5)	46.98(5)	23.92 (7)
Yeast	**71.13**(1)	66.50(7)	66.80(6)	70.82(2)	70.81(3)	67.83(5)	68.51 (4)
Avg. rank	(4.45)	(**2.45**)	(3.59)	(4.91)	(4.82)	(3.09)	(4.68)
Recall	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	33.80(7)	**36.86**(1)	34.82(5)	35.50(3)	35.49(4)	36.47(2)	34.06 (6)
Emotions	48.16(6)	**68.03**(1)	65.84(2)	51.02(4)	49.81(5)	53.44(3)	47.00 (7)
Enron	50.50(6)	**59.89**(1)	57.48(2)	56.78(3)	56.68(4)	54.86(5)	41.78 (7)
Genbase	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07**(4)	**99.07** (4)
Image	43.32(6)	**55.96**(1)	53.68(2)	43.69(4)	43.54(5)	47.74(3)	43.07 (7)
Mediamill	52.33(7)	59.25(3)	54.34(5)	**62.03**(1)	60.02(2)	58.02(4)	52.41 (6)
Medical	78.34(6)	**83.86**(1)	81.02(4)	83.09(2.5)	83.09(2.5)	80.57(5)	75.09 (7)
Reuters	84.90(7)	**91.90**(1)	90.29(2)	85.01(5)	84.93(6)	87.90(3)	85.10 (4)
Scene	62.87(7)	**87.55**(1)	82.38(2)	68.50(5)	68.17(6)	77.10(3)	70.81 (4)
Slashdot	44.21(6)	**71.42**(1)	70.37(2)	45.96(3.5)	45.96(3.5)	45.05(5)	21.68 (7)
Yeast	58.86(6)	**66.42**(1)	60.93(3)	59.43(4)	59.38(5)	62.50(2)	55.83 (7)
Avg. rank	(6.18)	(**1.45**)	(3.00)	(3.55)	(4.27)	(3.55)	(6.00)
$\boldsymbol{F_1}$	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	37.02(7)	**39.22**(1)	37.54(5)	37.93(3)	37.92(4)	39.03(2)	37.04 (6)
Emotions	49.20(6)	**62.05**(1)	60.87(2)	51.54(4)	50.33(5)	53.31(3)	48.03 (7)
Enron	55.66(6)	**59.97**(1)	58.20(2)	57.78(3.5)	57.78(3.5)	56.73(5)	47.32 (7)
Genbase	99.18(4)	99.18(4)	**99.21**(1.5)	99.10(6.5)	99.10(6.5)	99.18(4)	**99.21** (1.5)
Image	42.12(7)	**52.69**(1)	51.68(2)	42.42(5)	42.38(6)	46.18(3)	43.10 (4)
Mediamill	59.17(3)	**60.39**(1)	57.23(4)	48.00(6)	47.06(7)	59.67(2)	56.67 (5)
Medical	77.33(6)	**81.66**(1)	79.82(2)	79.45(3.5)	79.45(3.5)	79.35(5)	75.54 (7)
Reuters	84.13(7)	**87.74**(1)	87.05(2)	84.23(5)	84.21(6)	86.43(3)	84.65 (4)
Scene	61.25(7)	71.38(2)	68.46(4)	66.68(6)	66.75(5)	**72.85**(1)	68.78 (3)
Slashdot	44.33(6)	**55.97**(1)	55.47(2)	46.05(3.5)	46.05(3.5)	45.18(5)	22.36 (7)
Yeast	61.68(5)	**63.28**(1)	60.98(6)	61.89(3)	61.86(4)	62.42(2)	58.44 (7)
Avg. rank	(5.82)	(**1.36**)	(2.95)	(4.45)	(4.91)	(3.18)	(5.32)
Jaccard	BR	AID	STA^{y}	AID$^{y'}$	STA	AID^{p}	STA^{p}
Bibtex	31.50(7)	**33.59**(1)	32.32(3)	32.16(4)	32.15(5)	33.28(2)	31.70 (6)
Emotions	42.27(6)	**52.89**(1)	51.76(2)	44.52(4)	43.46(5)	46.12(3)	41.68 (7)
Enron	44.69(5)	**48.50**(1)	47.09(2)	45.95(3.5)	45.95(3.5)	44.23(6)	35.81 (7)
Genbase	98.94(4)	98.94(4)	**98.97**(1.5)	98.82(6.5)	98.82(6.5)	98.94(4)	**98.97** (1.5)
Image	38.60(7)	**47.88**(1)	47.32(2)	38.85(6)	38.87(5)	42.43(3)	39.95 (4)
Mediamill	46.70(2)	**48.13**(1)	45.42(4)	34.70(6)	34.09(7)	46.00(3)	42.47 (5)
Medical	74.51(6)	**78.40**(1)	76.95(2)	75.43(4.5)	75.43(4.5)	76.37(3)	73.13 (7)
Reuters	81.67(7)	**84.37**(1)	84.00(2)	81.77(5)	81.76(6)	83.80(3)	82.36 (4)
Scene	59.41(7)	65.90(3)	64.00(6)	64.64(5)	64.88(4)	**69.74**(1)	66.67 (2)
Slashdot	42.71(6)	**49.98**(1)	49.70(2)	44.29(3.5)	44.29(3.5)	43.50(5)	21.55 (7)
Yeast	50.71(5)	**52.40**(1)	49.68(6)	50.97(3)	50.93(4)	51.22(2)	46.77 (7)
Avg. rank	(5.64)	(**1.45**)	(2.95)	(4.64)	(4.91)	(3.18)	(5.23)

Table 3. Aggregated vs. stacking-based approaches: Hamming loss and 0/1 loss

Hamming	BR	AID	STA$^{\boldsymbol{y}}$	AID$^{\boldsymbol{y}'}$	STA	AID$^{\boldsymbol{p}}$	STA$^{\boldsymbol{p}}$
Bibtex	**1.21**(1.5)	1.22(4)	**1.21**(1.5)	1.26(6.5)	1.26(6.5)	1.22(4)	1.22 (4)
Emotions	22.03(4)	23.04(6)	23.15(7)	21.75(2)	21.83(3)	**21.47**(1)	22.14 (5)
Enron	**4.46**(1)	4.82(3)	4.88(6)	4.84(5)	4.83(4)	4.69(2)	4.95 (7)
Genbase	0.08(4)	0.08(4)	**0.07**(1.5)	0.09(6.5)	0.09(6.5)	0.08(4)	**0.07** (1.5)
Image	20.25(3.5)	21.90(7)	21.61(6)	20.33(5)	20.23(2)	20.25(3.5)	**20.04** (1)
Mediamill	**2.76**(1)	3.03(3)	3.10(5)	5.48(6)	5.50(7)	2.99(2)	3.09 (4)
Medical	0.99(4)	**0.95**(1)	0.98(3)	1.12(6.5)	1.12(6.5)	0.97(2)	1.02 (5)
Reuters	4.58(3)	5.81(7)	5.77(6)	4.59(4)	4.60(5)	**4.44**(1)	4.51 (2)
Scene	**9.83**(1)	18.66(7)	18.31(6)	9.99(3)	9.85(2)	10.28(5)	10.07 (4)
Slashdot	**3.73**(1)	8.81(7)	8.80(6)	3.86(3.5)	3.86(3.5)	3.75(2)	4.49 (5)
Yeast	**19.81**(1)	21.29(5)	21.56(7)	19.85(2.5)	19.85(2.5)	20.66(4)	21.41 (6)
Avg. rank	(**2.27**)	(4.91)	(5.00)	(4.59)	(4.41)	(2.77)	(4.05)
0/1 loss	BR	AID	STA$^{\boldsymbol{y}}$	AID$^{\boldsymbol{y}'}$	STA	AID$^{\boldsymbol{p}}$	STA$^{\boldsymbol{p}}$
Bibtex	82.83(5)	**81.49**(1)	81.54(2)	82.99(6.5)	82.99(6.5)	81.96(3)	82.30 (4)
Emotions	79.42(7)	**74.19**(1)	74.70(2)	77.40(4)	77.91(6)	76.05(3)	77.73 (5)
Enron	86.90(3)	85.37(2)	**85.25**(1)	88.13(5)	88.07(4)	92.54(6)	95.95 (7)
Genbase	**1.81**(3)	**1.81**(3)	**1.81**(3)	2.11(6.5)	2.11(6.5)	**1.81**(3)	**1.81** (3)
Image	71.50(7)	65.75(2)	**65.20**(1)	71.40(6)	71.25(5)	68.35(3)	69.20 (4)
Mediamill	90.36(3)	**88.10**(1)	88.14(2)	97.96(7)	97.90(6)	94.48(4)	96.54 (5)
Medical	33.94(4)	**31.18**(1)	31.49(2)	36.61(6.5)	36.61(6.5)	32.62(3)	34.05 (5)
Reuters	25.69(7)	25.16(4)	24.60(3)	25.59(6)	25.54(5)	**24.10**(1)	24.50 (2)
Scene	46.03(6)	46.07(7)	44.99(5)	41.38(4)	40.63(3)	**39.26**(1)	39.38 (2)
Slashdot	61.95(4)	63.20(6)	62.85(5)	**60.82**(1.5)	**60.82**(1.5)	61.34(3)	80.83 (7)
Yeast	84.53(5)	**83.08**(1)	84.07(4)	83.95(2.5)	83.95(2.5)	85.85(6)	90.03 (7)
Avg. rank	(4.91)	(**2.64**)	(2.73)	(5.05)	(4.77)	(3.27)	(4.64)

is only better than BR in Recall, but STA$^{\boldsymbol{y}}$ is significantly better than BR in Recall, F_1 and Jaccard index, and worse in Hamming loss. The results concerning Hamming loss are quite intriguing, because both approaches seem able to improve when true labels are used, but the results in terms of Hamming loss are worse. Despite this fact, we do think that these results confirm the idea that using the actual label data is better to capture the correlations among labels.

Given that the base learner employed is a logistic regressor that provides posterior probabilities, we included two other methods (AID$^{\boldsymbol{p}}$, STA$^{\boldsymbol{p}}$) that consist of taking the probabilities yielded by the independent models rather than their binary outputs in the testing phase, remaining unchanged the original training phase for each approach. Comparing their results with the original versions, we have that AID performs better than AID$^{\boldsymbol{p}}$ in all measures, except in Hamming loss in which the performance of AID$^{\boldsymbol{p}}$ ranks second behind BR. AID$^{\boldsymbol{p}}$ is significantly better than BR in Recall, F_1 and Jaccard index, and it is very competitive in Hamming loss. In the case of STA$^{\boldsymbol{p}}$ we do not observe any improvement, otherwise the results are pretty the same of STA. It seems that using the posteriors helps to reduce the Hamming loss, but it is worse for the other measures.

Finally, we want to compare the aggregating and stacking strategies. Despite the stacking method improves when true labels are used (STA$^{\boldsymbol{y}}$), the results are still worse than those of AID classifier. This different performance must come from the aggregating idea, given that the only difference between both algorithms is the way they decide the final predictions. The fact that AID prevails

Table 4. AID vs. state-of-the-art methods: Precision, Recall, F_1 and Jaccard index

Precision	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	48.19(3)	26.60(7)	28.92(6)	47.08(5)	47.69(4)	48.70(2)	**49.08** (1)
Emotions	56.36(4)	52.42(7)	**67.54**(1)	52.79(6)	56.26(5)	62.30(2)	59.03 (3)
Enron	**69.99**(1)	54.90(6)	52.75(7)	56.05(5)	65.19(4)	66.10(3)	66.37 (2)
Genbase	99.52(3.5)	97.70(7)	98.90(6)	**99.57**(1)	99.52(3.5)	99.52(3.5)	99.52 (3.5)
Image	44.23(7)	44.33(6)	48.52(2)	46.66(4)	45.99(5)	**53.97**(1)	48.18 (3)
Mediamill	78.81(2)	76.93(4)	73.52(5)	**80.40**(1)	77.87(3)	70.91(6)	70.16 (7)
Medical	78.94(5)	62.43(7)	63.40(6)	80.79(4)	81.04(2)	**82.50**(1)	80.88 (3)
Reuters	85.79(5)	82.23(6)	70.71(7)	**89.62**(1)	86.41(4)	87.53(2)	87.48 (3)
Scene	61.46(7)	69.71(3)	71.40(2)	69.69(4)	67.45(6)	67.81(5)	**71.89** (1)
Slashdot	46.06(4)	6.15(7)	8.09(6)	50.91(2)	42.79(5)	**53.32**(1)	46.98 (3)
Yeast	71.13(3)	**72.92**(1)	71.75(2)	68.62(5)	70.58(4)	66.50(7)	67.83 (6)
Avg. rank	(4.05)	(5.55)	(4.55)	(3.45)	(4.14)	(**3.05**)	(3.23)
Recall	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	33.80(4)	14.06(7)	21.50(6)	**41.97**(1)	33.78(5)	36.86(2)	36.47 (3)
Emotions	48.16(6)	37.73(7)	64.54(2)	57.19(3)	48.92(5)	**68.03**(1)	53.44 (4)
Enron	50.50(4)	37.04(7)	38.04(6)	54.07(3)	45.79(5)	**59.89**(1)	54.86 (2)
Genbase	99.07(4.5)	94.96(7)	99.14(2)	**99.57**(1)	99.07(4.5)	99.07(4.5)	99.07 (4.5)
Image	43.32(6)	39.11(7)	43.68(5)	49.73(2)	44.10(4)	**55.96**(1)	47.74 (3)
Mediamill	52.33(5)	53.78(4)	56.69(3)	49.57(7)	51.25(6)	**59.25**(1)	58.02 (2)
Medical	78.34(5)	59.01(7)	65.05(6)	81.00(2)	79.01(4)	**83.86**(1)	80.57 (3)
Reuters	84.90(5)	81.09(6)	69.45(7)	89.63(2)	85.19(4)	**91.90**(1)	87.90 (3)
Scene	62.87(7)	68.73(5)	69.75(3)	69.52(4)	66.43(6)	**87.55**(1)	77.10 (2)
Slashdot	44.21(4)	5.69(7)	7.67(6)	53.18(2)	39.93(5)	**71.42**(1)	45.05 (3)
Yeast	58.86(6)	56.89(7)	60.41(4)	61.84(3)	59.40(5)	**66.42**(1)	62.50 (2)
Avg. rank	(5.14)	(6.45)	(4.55)	(2.73)	(4.86)	(**1.41**)	(2.86)
$\boldsymbol{F_1}$	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	37.02(4)	16.98(7)	22.38(6)	**41.28**(1)	36.86(5)	39.22(2)	39.03 (3)
Emotions	49.20(6)	41.60(7)	**62.97**(1)	51.95(4)	49.81(5)	62.05(2)	53.31 (3)
Enron	55.66(3)	41.82(6)	41.52(7)	52.43(4)	51.14(5)	**59.97**(1)	56.73 (2)
Genbase	99.18(3.5)	95.81(7)	98.78(6)	**99.50**(1)	99.18(3.5)	99.18(3.5)	99.18 (3.5)
Image	42.12(6)	40.63(7)	44.91(4)	46.32(2)	43.46(5)	**52.69**(1)	46.18 (3)
Mediamill	59.17(5)	59.55(4)	60.17(2)	57.72(7)	58.15(6)	**60.39**(1)	59.67 (3)
Medical	77.33(5)	59.41(7)	62.19(6)	79.65(2)	78.83(4)	**81.66**(1)	79.35 (3)
Reuters	84.13(5)	80.50(6)	69.10(7)	**88.58**(1)	84.69(4)	87.74(2)	86.43 (3)
Scene	61.25(7)	68.49(5)	69.97(3)	68.84(4)	66.31(6)	71.38(2)	**72.85** (1)
Slashdot	44.33(4)	5.84(7)	7.73(6)	50.49(2)	40.66(5)	**55.97**(1)	45.18 (3)
Yeast	61.68(6)	60.97(7)	62.85(2)	62.48(3)	61.80(5)	**63.28**(1)	62.42 (4)
Avg. rank	(4.95)	(6.36)	(4.55)	(2.82)	(4.86)	(**1.59**)	(2.86)
Jaccard	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID^p
Bibtex	31.50(4)	13.61(7)	18.09(6)	**34.28**(1)	31.46(5)	33.59(2)	33.28 (3)
Emotions	42.27(6)	34.09(7)	**55.08**(1)	42.98(5)	43.24(4)	52.89(2)	46.12 (3)
Enron	44.69(2)	31.83(7)	31.99(6)	41.30(4)	39.68(5)	**48.50**(1)	44.23 (3)
Genbase	98.94(3.5)	94.86(7)	98.25(6)	**99.29**(1)	98.94(3.5)	98.94(3.5)	98.94 (3.5)
Image	38.60(6)	38.45(7)	42.46(2)	42.15(4)	40.15(5)	**47.88**(1)	42.43 (3)
Mediamill	46.70(4)	48.11(3)	**48.82**(1)	45.10(6)	44.69(7)	48.13(2)	46.00 (5)
Medical	74.51(5)	56.76(7)	58.19(6)	76.88(2)	76.33(4)	**78.40**(1)	76.37 (3)
Reuters	81.67(5)	78.11(6)	67.10(7)	**86.38**(1)	82.41(4)	84.37(2)	83.80 (3)
Scene	59.41(7)	67.03(4)	68.77(2)	67.28(3)	65.05(6)	65.90(5)	**69.74** (1)
Slashdot	42.71(4)	5.68(7)	7.42(6)	47.18(2)	39.32(5)	**49.98**(1)	43.50 (3)
Yeast	50.71(6)	50.50(7)	**52.65**(1)	51.75(3)	50.96(5)	52.40(2)	51.22 (4)
Avg. rank	(4.77)	(6.27)	(4.00)	(2.91)	(4.86)	(**2.05**)	(3.14)

Table 5. AID vs. state-of-the-art methods: Hamming loss and 0/1 loss

Hamming	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID$^{\boldsymbol{p}}$
Bibtex	**1.21**(1.5)	1.36(5)	1.60(7)	1.49(6)	**1.21**(1.5)	1.22(3.5)	1.22 (3.5)
Emotions	22.03(4)	26.21(6)	**18.72**(1)	28.01(7)	21.72(3)	23.04(5)	21.47 (2)
Enron	**4.46**(1)	5.22(5)	5.60(6)	5.85(7)	4.72(3)	4.82(4)	4.69 (2)
Genbase	0.08(3.5)	0.45(7)	0.19(6)	**0.06**(1)	0.08(3.5)	0.08(3.5)	0.08 (3.5)
Image	20.25(4.5)	19.28(2)	**18.75**(1)	24.40(7)	19.80(3)	21.90(6)	20.25 (4.5)
Mediamill	2.76(2)	**2.70**(1)	2.82(4)	2.81(3)	2.86(5)	3.03(7)	2.99 (6)
Medical	0.99(5)	1.56(6)	1.90(7)	**0.95**(2)	**0.95**(2)	**0.95**(2)	0.97 (4)
Reuters	4.58(4)	6.03(6)	8.32(7)	**3.85**(1)	4.42(2)	5.81(5)	4.44 (3)
Scene	9.83(4)	8.66(2)	**8.38**(1)	9.90(5)	9.06(3)	18.66(7)	10.28 (6)
Slashdot	**3.73**(1)	5.18(6)	5.17(5)	4.65(4)	3.78(3)	8.81(7)	3.75 (2)
Yeast	19.81(3)	19.43(2)	**19.18**(1)	20.30(5)	19.98(4)	21.29(7)	20.66 (6)
Avg. rank	(3.05)	(4.36)	(4.18)	(4.36)	(**3.00**)	(5.18)	(3.86)
0/1 loss	BR	MLkNN	IBLR	RAkEL	ECC*	AID	AID$^{\boldsymbol{p}}$
Bibtex	82.83(4)	94.06(7)	91.64(6)	83.56(5)	82.61(3)	**81.49**(1)	81.96 (2)
Emotions	79.42(5)	87.00(7)	**68.97**(1)	83.45(6)	77.05(4)	74.19(2)	76.05 (3)
Enron	86.90(2)	94.89(7)	92.30(4)	88.49(3)	93.07(6)	**85.37**(1)	92.54 (5)
Genbase	1.81(3.5)	8.16(7)	4.08(6)	**1.51**(1)	1.81(3.5)	1.81(3.5)	1.81 (3.5)
Image	71.50(7)	67.95(3)	**64.70**(1)	69.70(6)	69.40(5)	65.75(2)	68.35 (4)
Mediamill	90.36(4)	86.24(2)	**85.86**(1)	91.14(5)	93.80(6)	88.10(3)	94.48 (7)
Medical	33.94(5)	51.12(6)	52.98(7)	31.39(3)	31.19(2)	**31.18**(1)	32.62 (4)
Reuters	25.69(5)	29.04(6)	38.88(7)	**20.24**(1)	24.43(3)	25.16(4)	24.10 (2)
Scene	46.03(6)	37.35(2.5)	**34.82**(1)	37.35(2.5)	38.72(4)	46.07(7)	39.26 (5)
Slashdot	61.95(2)	94.76(7)	93.47(6)	62.11(3)	64.57(5)	63.20(4)	**61.34** (1)
Yeast	84.53(6)	82.29(2)	**79.19**(1)	83.08(3.5)	83.58(5)	83.08(3.5)	85.85 (7)
Avg. rank	(4.50)	(5.14)	(3.73)	(3.55)	(4.23)	(**2.91**)	(3.95)

in all measures, although the differences are sometimes small (Hamming and 0/1 loss), suggests that the aggregation idea helps to increase the performance. We observe even bigger differences when comparing the scores of BR and AID. These facts corroborate our original assumption of the existence of two kind of labels: some labels are explained by the mere description of the examples, whereas others have been assigned as a consequence of other labels. The differences in the performance may provide an estimation about the proportion of existing labels of each kind and the degree of overlapping.

4.2 AID Classifier vs. State-of-the-Art Methods

The second group of experiments was designed to compare our approach AID with other well-known multi-label classifiers: BR, MLkNN, RAkEL, IBLR and ECC*. We also included AID$^{\boldsymbol{p}}$ because of its good performance in all measures, specially in terms of Hamming loss. Table 4 and Table 5 show all the scores.

As it can be seen, the three top positions in the ranking are occupied by AID RAkEL and AID$^{\boldsymbol{p}}$ for all measures, except for Hamming loss, for which ECC*, BR and also AID$^{\boldsymbol{p}}$ win the others. It is remarkable than MLkNN is always placed in the last position for all measures. The Bergmann-Hommel's test (see Table 6) shows that AID is significantly better than BR, MLkNN, IBLR and ECC* in Recall, F_1 and Jaccard index, and there are not significant differences in Precision, Hamming and 0/1 losses. Also, AID$^{\boldsymbol{p}}$ presents significant differences

Table 6. Pairs of methods with significant differences according to Bergmann-Hommel's test. The number 90% or 95% indicates the significant level.

		Recall	F_1	Jaccard	Hamming			Recall	F_1	Jaccard
AID	$\succ$ BR	95%	95%	95%	-90%	AID	$\succ$ BR	95%	95%	95%
AID	$\succ$ STA	95%	95%	95%		AID	$\succ$ MLkNN	95%	95%	95%
AID	$\succ$ STA$^{\boldsymbol{p}}$	95%	95%	95%		AID	$\succ$ IBLR	95%	95%	
AID	$\succ$ AID$^{\boldsymbol{y}'}$		95%	95%		AID	$\succ$ ECC*	95%	95%	95%
AID$^{\boldsymbol{p}}$	$\succ$ BR	95%	95%	90%		AID$^{\boldsymbol{p}}$	$\succ$ MLkNN	95%	95%	95%
AID$^{\boldsymbol{p}}$	$\succ$ STA$^{\boldsymbol{p}}$	90%				AID$^{\boldsymbol{p}}$	$\succ$ BR	90%		
AID$^{\boldsymbol{y}'}$	$\succ$ BR	95%				RAkEL	$\succ$ MLkNN	95%	95%	95%
AID$^{\boldsymbol{y}'}$	$\succ$ STA$^{\boldsymbol{p}}$	90%				RAkEL	$\succ$ BR	90%		
STA$^{\boldsymbol{y}}$	$\succ$ BR	95%	95%	90%	-90%					
STA$^{\boldsymbol{y}}$	$\succ$ STA$^{\boldsymbol{p}}$	95%								

in those measures but only with regard to BR and MLkNN. RAkEL is also quite competitive, since it is significantly better than BR in Recall and than MLkNN in Recall, F_1 and Jaccard index.

From the results obtained, one can extract that AID$^{\boldsymbol{p}}$ keeps a steady behavior over all performance measures, whereas AID improves in Precision, Recall, F_1, Jaccard index and 0/1 loss and ECC* does it in Hamming loss. Both seem to be opposing methods with regard to the behavior of the measures, whereas RAkEL seems to be placed in between. Hence, if there is not a clear target measure to optimize, then AID$^{\boldsymbol{p}}$ may be a good choice, followed by RAkEL. Conversely, if the goal is to maximize Hamming loss, then ECC*, or even BR must be chosen. Finally, for maximizing the rest of measures AID is more promising.

5 Conclusions

This paper proposes a multi-label learning approach, called AID, whose main property is to aggregate independent and dependent models. Under the assumption that, for a given instance, certain labels can be predicted using only the description of the object, but others require to consider their dependences with respect to other labels, the idea is to combine classifiers aimed to learn each of these kind of relationships. In this work, we study the properties of our approach in the context of multi-label classification, but our framework is flexible enough to be adapted to other learning tasks, specially to multi-label ranking, and it can also be extended in some directions.

Several experiments over a benchmark multi-label data sets were carried out using different learning approaches. None of the methods outperforms the others for all measures considered, but AID classifier exhibits a very competitive performance, specially in terms of Recall, F_1 and Jaccard index, with regard to other state-of-the-art algorithms previously proposed in the literature. Besides, the computational complexity of AID is linear with respect to the number of labels.

References

1. Cheng, W., Hüllermeier, E.: Combining instance-based learning and logistic regression for multilabel classification. Machine Learning 76(2-3), 211–225 (2009)
2. Clare, A., King, R.D.: Knowledge discovery in multi-label phenotype data. In: European Conf. on Data Mining and Knowledge Discovery, pp. 42–53 (2001)
3. Dembczynski, K., Cheng, W., Hüllermeier, E.: Bayes Optimal Multilabel Classification via Probabilistic Classifier Chains. In: ICML, pp. 279–286 (2010)
4. Dembczyński, K., Waegeman, W., Cheng, W., Hüllermeier, E.: Regret analysis for performance metrics in multi-label classification: The case of hamming and subset zero-one loss. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS, vol. 6321, pp. 280–295. Springer, Heidelberg (2010)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
6. Elisseeff, A., Weston, J.: A Kernel Method for Multi-Labelled Classification. In: ACM Conf. on Research and Develop. in Infor. Retrieval, pp. 274–281 (2005)
7. Fürnkranz, J., Hüllermeier, E., Loza Mencía, E., Brinker, K.: Multilabel classification via calibrated label ranking. Machine Learning 73, 133–153 (2008)
8. García, S., Herrera, F.: An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. Journal of Machine Learning Research 9, 2677–2694 (2008)
9. Ghamrawi, N., McCallum, A.: Collective multi-label classification. In: ACM Int. Conf. on Information and Knowledge Management, pp. 195–200. ACM, New York (2005)
10. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Pacific-Asia Conf. on Know. Disc. and Data Mining, pp. 22–30 (2004)
11. Lin, C.-J., Weng, R.C., Keerthi, S.S.: Trust region Newton method for logistic regression. Journal of Machine Learning Research 9(apr), 627–650 (2008)
12. McCallum, A.K.: Multi-label text classification with a mixture model trained by em. In: AAAI 1999 Workshop on Text Learning (1999)
13. Qi, G.J., Hua, X.S., Rui, Y., Tang, J., Mei, T., Zhang, H.J.: Correlative multi-label video annotation. In: Proceedings of the International conference on Multimedia, pp. 17–26. ACM, New York (2007)
14. Read, J., Pfahringer, B., Holmes, G.: Multi-label classification using ensembles of pruned sets. In: IEEE Int. Conf. on Data Mining, pp. 995–1000. IEEE, Los Alamitos (2008)
15. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5782, pp. 254–269. Springer, Heidelberg (2009)
16. Schapire, R.E., Singer, Y.: Boostexter: A boosting-based system for text categorization. Machine Learning, 135–168 (2000)
17. Tsoumakas, G., Dimou, A., Spyromitros, E., Mezaris, V., Kompatsiaris, I., Vlahavas, I.: Correlation-based pruning of stacked binary relevance models for multi-label learning. In: Workshop on Learning from Multi-Label Data, Bled, Slovenia, pp. 101–116 (2009)
18. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. In: Data Mining and Knowledge Discovery Handbook, pp. 667–685 (2010)
19. Tsoumakas, G., Vlahavas, I.P.: Random k-Labelsets: An Ensemble Method for Multilabel Classification. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007)

20. Wolpert, D.H.: Stacked generalization. Neural Networks 5, 214–259 (1992)
21. Zhang, M.-L., Zhou, Z.-H.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans. on Knowl. and Data Eng. 18, 1338–1351 (2006)
22. Zhang, M.-L., Zhou, Z.-H.: Ml-knn: A lazy learning approach to multi-label learning. Pattern Recognition 40(7), 2038–2048 (2007)