Multi-Label Lazy Associative Classification*

Adriano Veloso[1], Wagner Meira Jr.[1], Marcos Gonçalves[1], and Mohammed Zaki[2]

[1] Computer Science Department, Universidade Federal de Minas Gerais, Brazil
{adrianov,meira,mgoncalv}@dcc.ufmg.br
[2] Computer Science Department, Rensselaer Polytechnic Institute, USA
zaki@cs.rpi.edu

Abstract. Most current work on classification has been focused on learning from a set of instances that are associated with a single label (i.e., single-label classification). However, many applications, such as gene functional prediction and text categorization, may allow the instances to be associated with multiple labels simultaneously. Multi-label classification is a generalization of single-label classification, and its generality makes it much more difficult to solve.
Despite its importance, research on multi-label classification is still lacking. Common approaches simply learn independent binary classifiers for each label, and do not exploit dependencies among labels. Also, several small disjuncts may appear due to the possibly large number of label combinations, and neglecting these small disjuncts may degrade classification accuracy. In this paper we propose a multi-label lazy associative classifier, which progressively exploits dependencies among labels. Further, since in our lazy strategy the classification model is induced on an instance-based fashion, the proposed approach can provide a better coverage of small disjuncts. Gains of up to 24% are observed when the proposed approach is compared against the state-of-the-art multi-label classifiers.

1 Introduction

The classification problem is to build a model, which, based on external observations, assigns an instance to one or more labels. A set of examples is given as the training set, from which the model is built. A typical assumption in classification is that labels are mutually exclusive, so that an instance can be mapped to only one label. However, due to ambiguity or multiplicity, it is quite natural that most of the applications violate this assumption, allowing instances to be mapped to multiple labels simultaneously. For example, a movie being mapped to *action* or *adventure*, or a song being classified as *rock* or *ballad*, could all lead to violations of the single-label assumption.

Multi-label classification consists in learning a model from instances that may be associated with multiple labels, that is, labels are not assumed to be mutually exclusive. Most of the proposed approaches [7,1,3] for multi-label classification employ heuristics, such as learning independent classifiers for each label, and employing ranking and thresholding schemes for classification. Although simple, these heuristics do not deal with important issues such as *small disjuncts* and *correlated labels*.

* This research was sponsored by UOL (www.uol.com.br) through its UOL Bolsa Pesquisa program, process number 20060519184000a.

In essence, small disjuncts are rules covering a small number of examples, and thus they are often neglected. The problem is that, although a single small disjunct covers only few examples, many of them, collectively, may cover a substantial fraction of all examples, and simply eliminating them may degrade classification accuracy [4]. Small disjuncts pose significant problems in single-label classification, and in multi-label classification these problems are worsened, because the search space for disjuncts increases due to the possibly large number of label combinations. Also, it is often the case that there are strong dependencies among labels, and such dependencies, when properly explored, may provide improved accuracy in multi-label classification.

In this paper we propose an approach which deals with small disjuncts while exploring dependencies among labels. To address the problem with small disjuncts, we adopt a lazy associative classification approach. Instead of building a single set of *class association rules* (CARs) that is good on average for all predictions, the proposed lazy approach delays the inductive process until a test instance is given for classification, therefore taking advantage of better qualitative evidence coming from the test instance, and generating CARs on a demand-driven basis. Small disjuncts are better covered, due to the highly specific bias associated with this approach. We address the label correlation issue by defining *multi-label class association rules* (MCARs), a variation of CARs that allows the presence of multiple labels in the antecedent of the rule. The search space for MCARs is huge and to avoid an exhaustive enumeration. which would be necessary to find the best label combination, we employ a novel heuristic called *progressive label focusing*, which makes feasible the exploration of associations among labels.

The proposed approach was evaluated using two different applications: text categorization and gene functional prediction. It consistently achieves better performance than the state-of-the-art multi-label classifiers, showing gains up to 24%.

2 Related Work

Typical approaches for multi-label classification are based on training an independent binary classifier for each label. These independent classifiers are used to assign a probability of membership to each label, and then an instance is classified into the labels that rank above a given threshold. Examples of this approach include ADTBOOST.MH [2] (decision trees that can directly handle multi-label problems), a multi-label generalization of SVMs [3], and a a multi-label lazy learning based on the kNN approach [7]. In [6] an approach based on independent associative classifiers was proposed. However, this approach was only evaluated in single-label problems, and thus, the performance of multi-label associative classifiers for multi-label problems is still unknown.

The main problem with the binary approach is that it does not consider correlation among labels. The direct multi-label approach explores this correlation by considering a combination of labels as a new, separate label [1]. For instance, a multi-label problem with 10 labels will be transformed to a single-label problem composed of potentially 1,024 labels. The problem now is that a relatively small number of examples may be associated with those new labels, specially if the combination contains many labels. While these approaches are able to capture dependencies among labels, the poor coverage of small disjuncts may degrade overall accuracy.

3 Single-Label Associative Classification

A typical associative classifier, suitable for single-label classification problems, is described in this section. An associative classification model is composed of *class association rules* (CARs), which are defined in the following.

DEFINITION 1. [CLASS ASSOCIATION RULES] CARs are rules of the form $\mathcal{X} \xrightarrow{\sigma,\theta} c_i$, where the set $\mathcal{X}$ is allowed to contain only features (i.e., $\mathcal{X} \subseteq \mathcal{I}$, where $\mathcal{I}$ is the set of all possible features), and c_i is one of the n labels (i.e., $c_i \in \mathcal{C}$, where $\mathcal{C}$ is the set of all possible labels). A valid CAR has support (σ) and confidence (θ) greater than or equal to the corresponding thresholds, σ_{min} and θ_{min}.

Common approaches for associative classification employ a slightly modified algorithm for mining valid CARs directly from the training data. When a sufficient number of valid CARs are found, the model (denoted as $\mathcal{M}$) is finally completed, and it is used to predict the label of the test instances. Due to class overlapping, and since labels are mutually exclusive, CARs may perform contradictory predictions. For example, let $\mathcal{T}$ be a test instance, and let $\mathcal{X}$ and $\mathcal{Y}$ be two subsets of $\mathcal{T}$. Also suppose that the valid CARs $\mathcal{X} \rightarrow c_i$ and $\mathcal{Y} \rightarrow c_j$ (with $i \neq j$) are in $\mathcal{M}$. These CARs are contradictory, since they predict different labels for the same test instance, $\mathcal{T}$. To address this problem, the rule-set $\mathcal{M}$ is interpreted as a poll, in which CAR $\mathcal{X} \xrightarrow{\sigma,\theta} c_i \in \mathcal{M}$ is a vote of weight $\sigma \times \theta$ given by $\mathcal{X}$ for label c_i (note that other criteria for weighting the votes can be used). Weighted votes for each label are then summed, and the score of label c_i is given by the real-valued function s showed in Equation 1. In the end, the label associated with the highest score is finally predicted.

$$s(c_i) = \sum_{\mathcal{X} \xrightarrow{\sigma,\theta} c_i \in \mathcal{M}} \sigma \times \theta \tag{1}$$

Consider the set of instances shown in Table 1, used as a running example in this paper. Each instance corresponds to a movie, and to each movie is assigned a set of labels (but for this example, which refers to single-label classification, only the first label will be considered). If we set σ_{min} to 0.20 and θ_{min} to 0.66, then the model $\mathcal{M}$ will be composed of the following CARs:

1. actor=T. Hanks $\xrightarrow{0.30,0.75}$ label=Drama
2. ~~actor=L. DiCaprio $\xrightarrow{0.20,0.67}$ label=Drama~~
3. actor=M. Damon $\xrightarrow{0.20,0.67}$ label=Crime

Now, suppose we want to classify instance 11. In this case, only first and third CARs are applicable, since feature *actor=L. DiCaprio* is not present in instance 11 (thus, second CAR is crossed out). According to Equation 1, s(Drama)=0.225 and s(Crime)=0.134, and thus *Drama* will be predicted.

4 Multi-Label Lazy Associative Classification

In this section we extend the basic classifier described in the previous section, allowing it to predict multiple labels. We also propose an approach for exploring correlated labels, while dealing with small disjuncts, improving the classification model.

4.1 Independent Classifiers

A heuristic employed for multi-label classification is to build an independent classifier for each label. This extension is natural, and it is based on assigning a probability of membership to each label, $f(c_i)$. The probabilities are computed using the proportion of scores associated with each label normalized by the highest score (i.e., max):

$$f(c_i) = \frac{s(c_i)}{max}. \quad (2)$$

Once all probabilities are computed, labels are inserted into a ranking $\mathcal{L}$=$\{l_1, ..., l_n\}$, so that $f(l_1) \geq f(l_2) \geq \ldots \geq f(l_n)$. Those labels that rank above a threshold δ_{min} (i.e., $l_k | f(l_k) \geq \delta_{min}$) are assigned to the test instance. To illustrate this process, consider again the example shown in Table 1, but now each movie has multiple labels, which are all considered when mining the CARs. If we set σ_{min} to 0.20 and θ_{min} to 0.66, then $\mathcal{M}$ will be composed of the following CARs:

1. actor=M. Damon $\xrightarrow{0.30,1.00}$ label=Action
2. ~~actor=L. DiCaprio $\xrightarrow{0.30,1.00}$ label=Crime~~
3. ~~actor=T. Hanks $\xrightarrow{0.30,0.75}$ label=Drama~~
4. actor=M. Damon $\xrightarrow{0.20,0.67}$ label=Crime
5. ~~actor=L. DiCaprio $\xrightarrow{0.20,0.67}$ label=Drama~~

Now, suppose we want to classify instance 12 and δ_{min} is set to 0.66. Following Equation 2, f(Action)=1.00 and f(Crime)=0.45, and therefore label *Action* is predicted. Note that, although there is a strong association between feature *actor=B. Pitt* and label *Romance*, CAR *actor=B. Pitt→Romance* is considered a small disjunct, and is neglected by the classifier, even being important to classify instance 12. We refer to this classifier as IEAC (*independent eager associative classifier*).

	Id	Label	Title	Actors
Training Set	1	Comedy/Romance	Forrest Gump	T. Hanks
	2	Drama/Romance	The Terminal	T. Hanks
	3	Drama/Crime	Catch Me If You Can	T. Hanks and L. DiCaprio
	4	Drama/Crime	The Da Vinci Code	T. Hanks
	5	Drama/Crime	Blood Diamond	L. DiCaprio
	6	Crime/Action	The Departed	L. DiCaprio and M. Damon
	7	Crime/Action	The Bourne Identity	M. Damon
	8	Action/Romance	Syriana	M. Damon
	9	Romance	Troy	B. Pitt
	10	Drama/Crime	Confidence	E. Burns
Test Set	11	? [Drama/Action]	Saving Private Rian	T. Hanks, M. Damon and E. Burns
	12	? [Action/Romance]	Ocean's Twelve	B. Pitt and M. Damon
	13	? [Crime/Drama]	The Green Mile	T. Hanks

Table 1. Training and Test Instances.

Like most of the eager classifiers, IEAC does not perform well on complex spaces. This is because it generates CARs before the test instance is even known, and the difficulty in this case is in anticipating all the different directions in which it should attempt to generalize its training examples. In order to perform more general predictions, common approaches usually prefer to generalize more frequent disjuncts. This can reduce the performance in complex spaces, where small disjuncts may be important to classify specific instances. Lazy classifiers, on the other hand, generalize the examples exactly as needed to cover a specific test instance.

In lazy associative classification, whenever a test instance is being considered, that instance is used as a filter to remove irrelevant features and examples from the training data. This process automatically reduces the size of the training data, since irrelevant examples are not considered. As a result, disjuncts that are not frequent in the original training data, may become frequent in the filtered training data, providing a better coverage of small disjuncts. To illustrate this process, suppose we want to classify instance 12. As shown in Table 2 only four examples are relevant to this instance. If we set σ_{min} to 0.20 and θ_{min} to 0.66, then $\mathcal{M}$ will be composed of the following CARs:

1. actor=M. Damon $\xrightarrow{0.75,1.00}$ label=Action
2. actor=M. Damon $\xrightarrow{0.50,0.67}$ label=Crime
3. actor=B. Pitt $\xrightarrow{0.25,1.00}$ label=Romance

According to Equation 2, f(Action)=1.00, f(Crime)=0.45 and f(Romance)=0.33, and for $\delta_{min} = 0.66$, label combination *Action/Romance* is predicted. We refer to this classifier as ILAC (*independent lazy associative classifier*).

4.2 Correlated Classifiers

Labels in multi-label problems are often correlated, and as we will see in our experiments, this correlation can be helpful for improving classification performance. In this section we describe CLAC (*correlated lazy associative classifier*), which, unlike IEAC and ILAC, explicitly explores interactions among labels. The classification model is composed of *multi-label class association rules* (MCARs), which are defined next.

DEFINITION 2. [MULTI-LABEL CLASS ASSOCIATION RULES] MCARs are a special type of association rules of the form $\mathcal{X} \cup \mathcal{F} \xrightarrow{\sigma,\theta} c_i$, where $\mathcal{F} \subseteq (\mathcal{C} - c_i)$. A valid MCAR has σ and θ greater than or equal to the corresponding thresholds, σ_{min} and θ_{min}.

Id	Label	Actors
6	Crime/Action	M. Damon
7	Crime/Action	M. Damon
8	Action/Romance	M. Damon
9	Romance	B. Pitt

Table 2. Filtering according to Instance 12.

Id	Label	Actors
1	Comedy/Romance	T. Hanks
2	Drama/Romance	T. Hanks
3	Drama/Crime	T. Hanks
4	Drama/Crime	T. Hanks

Table 3. Filtering according to Instance 13.

Id	Label	Actors
2	Drama/Romance	T. Hanks
3	Drama/Crime	T. Hanks
4	Drama/Crime	T. Hanks

Table 4. Filtering according to Instance 13 and Label *Drama*.

The model is built iteratively, following a greedy heuristic called *progressive label focusing*, which tries to find the best label combination by making locally best choices. In the first iteration, $\mathcal{F} = \emptyset$, and a set of MCARs ($\mathcal{M}_1$) of the form $\mathcal{X} \rightarrow c_i$ is generated. Based on $\mathcal{M}_1$, label l_1 is assigned to the test instance. In the second iteration, l_1 is treated as a new feature and thus $\mathcal{F} = \{l_1\}$. A set of MCARs of the form $\mathcal{X} \cup \{l_1\} \rightarrow c_i$ ($\mathcal{M}_2$) is generated, and $\mathcal{M}_2$ is then used to assign label l_2 to the test instance. This process iterates until no more MCARs are generated. The basic idea is to progressively narrow the search space for MCARs as labels are being assigned to the test instance.

Consider again the example in Table 1, and suppose that we want to classify instance 13. The first step is to filter the training data according to the features in instance 13. The filtered training data is shown in Table 3, and if we set σ_{min} to 0.20 and θ_{min} to 0.66, then the corresponding model (i.e., $\mathcal{M}_1$) is composed of the following MCAR:

1. actor=T. Hanks $\xrightarrow{0.75,0.75}$ label=Drama

Label *Drama* is assigned to instance 13, and now this label is considered a new feature. The training data is filtered again, as shown in Table 4. The corresponding model (i.e., $\mathcal{M}_2$) is composed of the following MCAR:

1. actor=T. Hanks $\wedge$ label=Drama $\xrightarrow{0.67,0.67}$ label=Crime

Thus, label *Crime* is also assigned to instance 13, and since no more MCARs can be generated, the process stops. In summary, labels *Romance* and *Crime* are equaly related to feature *actor=T. Hanks* (see Table 3). Therefore it may be difficult to distinguish these two labels based solely on this feature. However, if we are confident that a movie starred by *T. Hanks* should be classified as *Drama*, then it is more likely that this movie should be classified as *Crime*, rather than *Romance* (as seen in Table 4).

5 Experimental Evaluation

Three datasets were used in our experiments. The first dataset, which is called ACM-DL (first level), was extracted from the first level of the ACM Computing Classification System (http://portal.acm.org/dl.cfm/), comprising a set of 81,251 documents labeled using the 11 first level categories of ACM. The second dataset, ACM-DL (second level) contains the same set of documents of ACM-DL (first level), but these documents are labeled using the 81 second level categories. In both datasets, each document is described by its title and abstract, citations, and authorship, resulting in a huge and sparse feature space. The third dataset, YEAST [5], is composed of a set of 2,417 genes. Each gene is described by the concatenation of micro-array expression data and phylogenetic profile, and is associated with a set of functional classes. There are 14 possible class labels, and the average number of labels for each gene is 4.24.

Figure 1 shows the number of instances associated with each label combination size for each dataset. The YEAST dataset presents very large combinations of labels (combinations of 11 labels). Figure 2 shows the association of each pair of labels for each dataset (an association level of 0.8 between labels $\mathcal{A}$ and $\mathcal{B}$, means that 80% of the instances that belong to $\mathcal{A}$, also belong to $\mathcal{B}$).

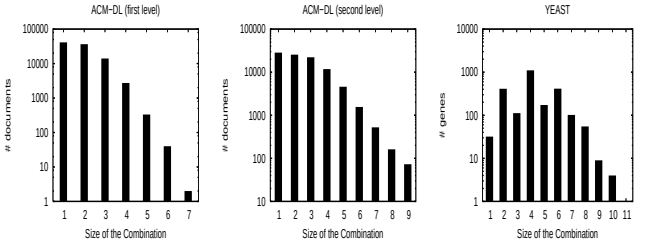

Fig. 1. Number of Instances Associated with each Combination Size.

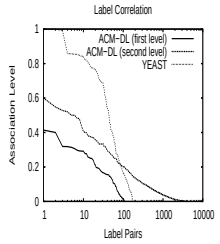

Fig. 2. Association of each Pair of Labels.

The experiments were performed on a Linux-based PC with a INTEL PENTIUM III 1.0 GHz processor and 1.0 GB RAM. In all experiments with the aforementioned datasets, we used 10-fold cross-validation and the final results of each experiment represent the average of the ten runs. We used three evaluation criteria that were proposed in [5]: Hamming Loss (h), Ranking Loss (r) and One-Error (o). All the results to be presented were found statistically significant based on a t-test at 5% significance level.

The proposed classifiers, IEAC, ILAC and CLAC are compared against boosting-style classifiers BOOSTEXTER [5] and ADTBOOST.MH [2], and the multi-label kernel method RANK-SVM [3]. We believe that these approaches are representative of some of the most effective multi-label methods available. For BOOSTEXTER and ADTBOOST.MH, the number of boosting rounds was set to 500 and 50, respectively. For RANK-SVM, polynomial kernels of degree 10 were used. For IEAC, ILAC and CLAC, σ_{min}, θ_{min} and δ_{min} were set to 0.01, 0.90 and 0.25, respectively.

Best results (including statistical ties) on each criterion are shown in bold face. Table 5 shows results obtained using the YEAST dataset, which is considered complex, with strong dependencies among labels. CLAC provide gains of 24% in terms of one-error, considering BOOSTEXTER as the baseline. The reason is that the simple decision function used by BOOSTEXTER is not suitable for this complex dataset. Also, the classification models employed by ILAC and CLAC are able to explore many more associations than the model induced by ADTBOOST.MH. CLAC performs much better than RANK-SVM since CLAC is able to explore dependencies between labels.

In the next set of experiments we compare IEAC, ILAC and CLAC, against RANK-SVM using the ACM-DL dataset (first and second levels). As can be seen, CLAC and ILAC are always superior than their eager counterpart, IEAC. RANK-SVM and ILAC shown competitive performance, and CLAC is the best performer in the ACM-DL datasets. To verify if the association between labels was properly explored by CLAC, we checked if the explicitly correlated categories shown in the ACM Computing Classification System (http://www.acm.org/class/1998/overview.html) were indeed used. We verified that some of these explicitly correlated categories often appear together in the predicted label combination (i.e., *Files* and *Database Management*, or *Simulation/Modeling* and *Probability/Statistics*). We further verified that some of the associated labels appear more frequently in the predictions performed by CLAC than was observed in the predictions of the other classifiers.

Evaluation Criterion	Classifier					
	BOOSTEXTER	ADTBOOST.MH	RANK-SVM	IEAC	ILAC	CLAC
h	0.220	0.207	0.196	0.203	0.191	**0.179**
r	0.186	–	0.163	0.178	0.164	**0.150**
o	0.278	0.244	0.217	0.232	**0.213**	**0.213**

Table 5. Results for Different Classifiers using the YEAST Dataset.

Evaluation Criterion	First Level				Second Level			
	Classifier							
	RANK-SVM	IEAC	ILAC	CLAC	RANK-SVM	IEAC	ILAC	CLAC
h	0.225	0.295	0.222	**0.187**	0.327	0.419	0.319	**0.285**
r	0.194	0.276	0.216	**0.179**	0.299	0.378	0.294	**0.273**
o	**0.244**	0.304	**0.238**	**0.238**	0.348	0.427	**0.331**	**0.331**

Table 6. Results for Different Classifiers using the ACM-DL Datasets.

6 Conclusions and Future Work

In this paper we propose a novel associative classification approach for multi-label classification. The model is induced in an instance-based fashion, in which the test instance is used as a filter to remove irrelevant features from the training data. Then a specific model is induced for each test instance, providing a much better coverage of small disjuncts. Also, the proposed approach properly explores the correlation among labels by employing a greedy heuristic called progressive label focusing, which allows the presence of multiple labels in the antecedent of the rule. Experimental results underscore the benefits of covering small disjuncts (i.e., lazy model induction) and exploring correlated labels (i.e., progressive label focusing). As future work, we intend to further explore correlated labels by also allowing the presence of multiple labels in the consequent of the rule.

References

1. M. Boutell, J. Luo, X. Shen, and C. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.
2. F. Comité, R. Gilleron, and M. Tommasi. Learning multi-label alternating decision trees from texts and data. In *Proc. of the Intl. Conf. on Machine Learning and Data Mining in Pattern Recognition*, pages 35–49, 2003.
3. A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems 14*, pages 681–687, 2001.
4. R. Holte, L. Acker, and B. Porter. Concept learning and the problem of small disjuncts. In *Proc. of the Intl. Joint Conf. on Artificial Intelligence*, pages 813–818, 1989.
5. R. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2-3):135–168, 2000.
6. F. Thabtah, P. Cowling, and Y. Peng. Multiple labels associative classification. *Knowledge and Information Systems*, 9(1):109–129, 2006.
7. M. Zhang and Z. Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.