TWO STAGE CLASSIFIER CHAIN ARCHITECTURE FOR EFFICIENT PAIR-WISE MULTI-LABEL LEARNING

Gjorgji Madjarov, Dejan Gjorgjevikj

FEEIT, Ss. Cyril and Methodius University, Skopje, Macedonia

ABSTRACT

A common approach for solving multi-label learning problems using problem-transformation methods and dichotomizing classifiers is the pair-wise decomposition strategy. One of the problems with this approach is the need for querying a quadratic number of binary classifiers for making a prediction that can be quite time consuming, especially in learning problems with large number of labels. To tackle this problem we propose a Two Stage Classifier Chain Architecture (TSCCA) for efficient pair-wise multi-label learning. Six different real-world datasets were used to evaluate the performance of the TSCCA. The performance of the architecture was compared with six methods for multi-label learning and the results suggest that the TSCCA outperforms the concurrent algorithms in terms of predictive accuracy. In terms of testing speed TSCCA shows better performance comparing to the pair-wise methods for multi-label learning.

***Index Terms*—** Multi-label, two stage, learning, classification

1. INTRODUCTION

The traditional problem of single-label classification is concerned with learning from examples, each associated with a single label λ_i from a finite set of disjoint labels $L = \{\lambda_1, \lambda_2, ..., \lambda_Q\}, Q > 1$. For $Q > 2$, the learning problem is referred to as a *multi-class classification*. On the other hand, the task of learning a mapping from an example $x \in X$ (X denotes the domain of examples) to a set of labels $Y \subseteq L$ is referred to as a *multi-label classification*. Thus, in contrast to multi-class classification, alternatives are not assumed to be mutually exclusive such that multiple labels may be associated with a single example i.e., each example can be a member of more than one class. The set of labels Y are called relevant, while the set $L\backslash Y$ represents irrelevant labels for a given example.

Besides the concept of multi-label classification, the multi-label learning introduces the concept of *multi-label ranking* [1], which is understood as learning a model that the query example x associates both with a (label) ranking of the complete label set and a bipartite partition of this set into relevant and irrelevant labels.

The issue of learning from multi-label data has recently attracted significant attention from many researchers. They are motivated from an increasing number of new applications, such as semantic annotation of images and video (news clips, movies clips), functional genomics (gene and protein function), music categorization into emotions, text classification (news articles, web pages, patents, emails, bookmarks, ...), directed marketing and others.

In recent years, many different approaches have been developed to solve the multi-label learning problems. Tsoumakas and Katakis [2] summarize them into two main categories: a) algorithm adaptation methods, and b) problem transformation methods. Algorithm adaptation methods extend specific learning algorithms to handle multi-label data directly. Examples include lazy learning [3] [4], neural networks [5], boosting [6], etc. Problem transformation methods, on the other hand, transform the multi-label learning problem into one or more single-label classification problems. A common approach for problem transformation is to use class binarization methods, i.e. decomposition of the problem into several binary sub-problems that can then be solved using a binary base classifier. The simplest strategy in the multi-label setting is the one-against-all strategy also referred to as the Binary Relevance method (BR) [2]. A method closely related to the BR method is the Classifier Chain method (CC) proposed by Read et al. [7]. Brinker et al. [1] propose a conceptually new technique for extending the common pair-wise learning approach to the multi-label scenario named Calibrated Label Ranking (CLR). The key idea of calibrated label ranking is to introduce an artificial (calibration) label λ_0, which represents the split-point between relevant and irrelevant labels. The calibration label λ_0 is assumed to be preferred over all irrelevant labels, but all relevant labels are preferred over it. At prediction time (when majority voting strategy is usually used), one will get a ranking over $Q + 1$ labels (the Q original labels plus the calibration label). Besides the majority voting that is usually used strategy in the prediction phase of the CLR algorithm, Mencia et al. [8] propose another more effective voting algorithm named Quick Weighted Voting algorithm for multi-label classification(QWeightedML). In our previous work [9] we proposed two stage voting strategy that signifcantly improves the testing times of the CLR and QWeightedML methods.

In this paper, we propose a novel architecture for efficient pair-wise multi-label learning, named Two Stage Classifier

Chain Architecture (TSCCA) and its modification Two Stage Pruned Classifier Chain Architecture (TSPCCA). The performances of these two architectures are evaluated on a selection of multi-label datasets that vary in terms of problem domain, number of labels and label cardinality. The obtained results demonstrate that our approaches outperform the competing methods in terms of predictive accuracy.

Section 2 introduces the Two Stage Classifier Chain Architecture and its modification. Section 3 presents the computational complexity in prediction phase of TSCCA. The experimental results, that compare the performance of the proposed approaches with the other competing methods are presented in Section 4. Section 5 gives the conclusions.

2. TWO STAGE CLASSIFIER CHAIN ARCHITECTURE (TSCCA)

In this paper, we propose a novel Two Stage Classifier Chain Architecture (TSCCA) for efficient pair-wise multi-label learning that is related to the CLR algorithm [10]. The main idea of this architecture is to reduce the number of classifiers that are needed to be consulted in the prediction phase of the CLR algorithm and increase the predictive accuracy.

The conventional pair-wise approach learns a model M_{ij} for all combinations of labels λ_i and λ_j, $1 \leq i < j \leq Q$. This means that, for a given training set $S = \{(x_1, Y_1), (x_2, Y_2), ..., (x_p, Y_p)\}$ ($x_i \in X, Y_i \subseteq L$, where X denotes the domain of examples and $L = \{\lambda_1, \lambda_2, ..., \lambda_Q\}$ is a finite set of labels), each model M_{ij} is trained with the examples (x_r, Y'_r) ($0 < r \leq p$) where Y'_r is defined as:

$$Y'_r = \begin{cases} +1, & \text{if } \lambda_i \in Y_r \text{ and } \lambda_j \notin Y_r \quad \text{(1a)} \\ -1, & \text{if } \lambda_j \in Y_r \text{ and } \lambda_i \notin Y_r \quad \text{(1b)} \end{cases}$$

This transformation of the dataset is known as the two-label transformation. The main disadvantage of this approach is that in the prediction phase a quadratic number of base classifiers (models) have to be consulted for each test example.

As a result of introducing the artificial calibration label λ_0 in the calibrated label ranking algorithm [10], the number of the base classifiers is increased by Q i.e., an additional set of Q binary preference models M_{0k} ($1 \leq k \leq Q$) is learned. The models M_{0k} that are learned by a pair-wise approach to calibrated ranking, and the models M_k that are learned by conventional binary relevance are equivalent. Using the same notation as in the case of the pair-wise models (M_{ij}), each model M_{0k} is trained with the examples (x_r, Y'_r) ($r \in 1...p$) where Y'_r is defined as:

$$Y'_r = \begin{cases} +1, & \text{if } \lambda_k \notin Y_r \quad \text{(2a)} \\ -1, & \text{if } \lambda_k \in Y_r \quad \text{(2b)} \end{cases}$$

This transformation of the datasets is addressed as single-label transformation.

In the standard voting algorithm for CLR, each test example needs to consult all the models M_{0k} ($1 \leq k \leq Q$) and

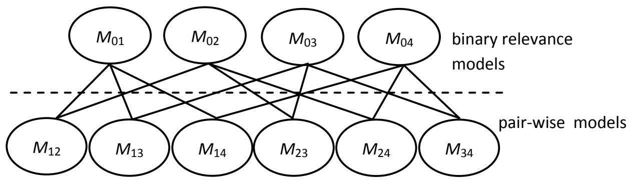

Fig. 1. Two Stage Classifier Chain Architecture

M_{ij} ($1 \leq i < j \leq Q$) in order to rank the labels by their order of preference. As a result of the increased number of models, the CLR method leads to more accurate prediction, but also leads to slower testing and higher computational complexity, especially when the number of the labels in the problem is large.

The Two Stage Classifier Chain Architecture is organized in two layers (Fig. 1). The first layer has Q binary relevance models M_{0k}, while the second layer has $Q * (Q-1)/2$ pair-wise models M_{ij}. Each model M_{0k} from the first layer is connected to $Q-1$ models M_{ij} from the second layer, where $k = i$ or $k = j$ ($1 \leq i \leq Q-1, i+1 \leq j \leq Q$). It is responsible for learning and predicting the probability association of label λ_k. On the other hand, each model of the second layer M_{ij} is connected to exactly two binary relevance models from the first layer (M_{0i} and M_{0j}) and is responsible for learning and predicting the probability associations of label λ_i and label λ_j ($p(\lambda_i) = 1 - p(\lambda_j)$).

The training phase of TSCCA starts with learning the binary relevance models M_{0k} (the models from the first layer). Each model is trained with the corresponding examples of a given training dataset. After learning the models of the first layer of the architecture, the feature space of each training example involved in the learning process of the second layer is extended with the probability predictions of all binary relevance models M_{0k}. That means that the feature space of each link which connects the binary relevance models and the pair-wise models is extended with the probability predictions of all models of the first layer. The training procedure of the first and the second layer is outlined on Fig. 2. Recall the notation for a training example (x, Y), where $Y \subseteq L$ ($L = \{\lambda_1, \lambda_2, ..., \lambda_Q\}$), and x is an instance feature vector.

In the prediction phase, each model M_{0k} tries to determine the relevant labels for the corresponding test example. Each model M_{0k} gives the probability (the output value of model M_{0k} is converted to probability) that the test example is associated with the label λ_k. If that probability is appropriately small (under some predetermined threshold), we can conclude that the artificial calibration label λ_0 is preferred over the label λ_k, i.e., the label λ_k belongs to the set of irrelevant labels. In that case, we conclude that, the pair-wise models of the second layer M_{ij} where $i = k$ or $j = k$, need not be consulted for the corresponding test example, because the binary relevance model M_{0k} from the first layer has suggested that the label λ_k belongs to the set of irrelevant labels.

```
TRAINING FIRST LAYER - TSCCA
($S = \{(x_1, Y_1), ..., (x_p, Y_p)\}$)
 1: for $k \in 1, ..., Q$ do
 2:     $S_{0k} = SingleLabelTransformation(S, \lambda_k)$
 3:     $M_{0k} = TrainingModel(S_{0k})$
 4: $x' \leftarrow x$
 5: $S' \leftarrow \{\}$
 6: for $(x, Y) \in S$ do
 7:     for $k \in 1, ..., Q$ do
 8:         $probability(M_{0k}) = classify(M_{0k}(x))$
 9:         $x' \leftarrow x' \cup (probability(M_{0k}))$
10:     $S' \leftarrow S' \cup (x', Y)$

TRAINING SECOND LAYER - TSCCA
($S' = \{(x'_1, Y_1), ..., (x'_p, Y_p)\}$)
 1: for $i \in 1, ..., Q-1$ do
 2:     for $j \in i+1, ..., Q$ do
 3:         $S'_{ij} = TwoLabelTransformation(S', \lambda_i, \lambda_j)$
 4:         $M_{ij} = TrainingModel(S'_{ij})$
```

```
CLASSIFY($x$) - TSCCA
 1: $x' \leftarrow \{x\}$
 2: for $k \in 1, ..., Q$ do
 3:     $probability(M_{0k}) = classify(M_{0k}(x))$
 4:     if $probability(M_{0k}) > 0.5$ then
 5:         votes[$\lambda_k$]++
 6:     else
 7:         votes[$\lambda_0$]++
 8:     $x' \leftarrow x' \cup (probability(M_{0k}))$
 9: for $i \in 1, ..., Q-1$ do
10:     for $j \in i+1, ..., Q$ do
11:         if $probability(M_{0i}) > t$ & $probability(M_{0j}) > t$ then
12:             $probability(M_{ij}) = classify(M_{ij}(x'))$
13:             if $probability(M_{ij}) > 0.5$ then
14:                 votes[$\lambda_j$]++
15:             else
16:                 votes[$\lambda_i$]++
17:         else
18:             if $probability(M_{0i}) > t$ then votes[$\lambda_i$]++
19:             if $probability(M_{0j}) > t$ then votes[$\lambda_j$]++
20: $order$(votes[ ])
```

Fig. 2. Training and testing procedures of TSCCA

For each label λ_k that belongs to the set of irrelevant labels, the number of pair-wise models that should be consulted decreases for $Q - 1$.

In order to decide which labels belong to the set of irrelevant labels i.e. which pair-wise models M_{ij} from the second layer do not have to be consulted, a threshold t ($0 \leq t \leq 1$) is introduced. The value of the threshold t can be determined by cross-validation or by some other method.

As previously described, each test example consults all binary relevance models M_{0k} first and then its feature space is extended with the prediction probabilities of all M_{0k} models. After that, the prediction probability of each model M_{0k} ($1 \leq k \leq Q$) is compared to the threshold t.

- If the prediction probability is above the threshold, the test example is forwarded to all the models M_{ij} of the second layer of the architecture that are associated to the model M_{0k}.
- If the prediction probability is under the threshold, the test example is not forwarded to any model of the second layer of the architecture.

From the viewpoint of the pair-wise models M_{ij}, if we consider the prediction probabilities of the binary relevance models M_{0i} and M_{0j} of the first layer, three distinct cases in the voting process can appear:

1. The prediction probabilities of both binary relevance models M_{0i} and M_{0j} that are connected to the pair-wise model M_{ij} are above the threshold t.
2. The prediction probability of only one of the binary relevance models (M_{0i} or M_{0j}) is above the threshold t.
3. The prediction probabilities of the binary relevance models M_{0i} and M_{0j} are both under the threshold t.

In the first case, the model M_{ij} is consulted and its prediction is decoded into a vote for one of the labels λ_i or λ_j. In the second case, M_{ij} is not consulted and its vote goes directly to the label whose binary relevance model prediction probability is above the threshold t. In the third case M_{ij} is not consulted and it does not vote at all. The votes of all M_{0k} models and M_{ij} models (where at least one prediction probability of the models M_{0i} and M_{0j} is above the threshold t) are then aggregated to obtain the final prediction by majority voting. The classification process is outlined on Fig. 2.

By increasing the value of the threshold, the number of pair-wise models that should be consulted decreases. For $t = 1$ no example is forwarded to the second layer of the architecture and the decision is made by the classifiers of the first layer. On the other hand, for $t = 0$, for each test example all pair-wise models of the second layer are consulted.

2.1. Two Stage Pruned Classifier Chain Architecture - TSPCCA

In this subsection we propose a modification of the TSCCA in which the feature space of each example, involved in the learning and testing process of the model M_{ij} of the second layer, is extended not by all, but only with the probability predictions of the binary relevance models M_{0i} and M_{0j}, i.e., the models that are directly connect to the model M_{ij}. These predictions are the most relevant for the model M_{ij} because the model M_{ij} tries to distinguish the label λ_i from the label λ_j. This architecture will be called the Two Stage Pruned Classifier Chain Architecture (TSPCCA).

Table 1. Dataset description and the values of the parameters t, a_{brmf} and r

	domain	$\#tr.e.$	$\#t.e.$	$\#f.$	$\#l.$	l_c	a_{brmf}	t	r
emotions[11]	music	391	202	72	6	1.87	3.118	0.25	0.220
scene [12]	image	1211	1159	294	6	1.07	3.337	0.02	0.260
yeast [13]	biology	1500	917	103	14	4.24	7.928	0.15	0.302
tmc2007 [14]	text	21519	7077	49060	22	2.16	4.93	0.1	0.042
bibtex [15]	text	4880	2515	1836	159	2.40	15.1	0.02	0.008
corel5k [16]	image	4500	500	499	374	3.52	64.01	0.01	0.029

3. COMPUTATIONAL COMPLEXITY

In the following, the term computational complexity is strictly used in the sense of a computational complexity in the prediction phase. The computational complexity of TSCCA significantly differs from the computational complexity of CLR. The computational complexity of CLR (O_{CLR}) can be defined as a sum of the computational complexity of the binary relevance models (O_{BR}) and the pair-wise models (O_P):

$$O_{CLR} = O_{BR} + O_P \tag{3}$$

On the other hand, computational complexity of TSCCA can be defined as a sum of computational complexity of the models located in the first layer of the architecture (O_{FL}) and computational complexity of the models located in the second layer of the architecture (O_{SL}):

$$O_{TSCCA} = O_{FL} + O_{SL} \tag{4}$$

The computational complexity of the first layer of the TSCCA and the computational complexity of the binary relevance models of the CLR method are equal ($O_{BR} = O_{FL}$). In both methods the models are the same and each test example must consult all of these models in order to predict the class the example belongs to.

The main difference in computational complexity between CLR and TSCCA is in the computational complexity of the pair-wise models of CLR and the second layer of TSCCA. As noted in the previous section, if the threshold is set to zero ($t = 0$), in TSCCA, all models of the second layer are consulted and we have $O_{SL} \approx O_P$ (the inequality is appearing as a result of the extended feature vector that adds some complexity to the classifiers themselves). If the threshold is set to one ($t = 1$), no models of the second layer will be consulted, so O_{SL} will be 0 and $O_{TSCCA} = O_{FL} = O_{BR}$. For threshold values $0 < t < 1$, $O_{SL} = r * O_P$ where r is a reduction parameter specific for each multi-label dataset ($0 < r < 1$). The reduction parameter r is related to label cardinality (l_c) [2], i.e., the average number of relevant labels per example in a given multi-label dataset. For a real world problem the reduction parameter r can be determined as:

$$r = \frac{a_{brmf} * (a_{brmf} - 1)}{Q * (Q - 1)} \tag{5}$$

where a_{brmf} is the average number of binary relevance models located in the first layer of TSCCA that give a probability that is above the threshold t in the prediction process. For an ideal case (prediction accuracy of 100% by the binary relevance models) a_{brmf} is getting equal to the label cardinality l_c ($a_{brmf} = l_c$).

4. EXPERIMENTS

The performances of the proposed methods were measured with two different multi-label evaluation metrics (Hamming Loss and Average Precision) proposed Schapire et al. [6] on the problems of recognition of text, video, images and protein function. The performance of TSCCA and TSPCCA are compared with the CLR method with majority voting strategy for pairwise multi-label classification [10], QWeightedML algorithm [8], Multi-label k-NN (ML-kNN) [3], Classifier Chain method (CC) [7] and the Two Stage Voting Method (TSVM) [9]. The training and the testing of TSCCA and TSPCCA were performed using a custom developed application that uses the MULAN [1] library for the machine learning framework Weka [17]. The other comparing methods are also implemented in MULAN.

Six different multi-label classification problems were addressed by each of the mentioned classifying methods. The recognition performance was recorded for every method. The complete description of the datasets: domain, number of training ($\#tr.e.$) and test ($\#t.e.$) examples, number of features ($\#f.$), the total number of labels ($\#l.$) and the label cardinality (l_c) are shown in Table 1.

The LIBSVM library [18] utilizing the SVM's with radial basis kernel were used for solving the partial binary classification problems. The kernel parameter $gamma$ and the penalty C for the datasets were determined by 5-fold cross validation using only the samples of the training sets.

Table 1 also shows the values of the threshold t for each dataset separately, for which the presented results of TSVM, TSCCA and TSPCCA are obtained. The value of the threshold t for each dataset was determined by 5-fold cross validation using only the samples of the training set. This was done in order to achieve optimal performance (trade off) in terms of the two evaluation metrics and the computational efficiency. The values 0.005 to 0.01 with step 0.001, 0.01 to 0.1 with step

[1]http://mulan.sourceforge.net/

Table 2. The evaluation of each method for every dataset

Eval. m.	**Algorithm**	**emotions**	**scene**	**yeast**	**tmc2007**	**bibtex**	**corel5k**
Hamming Loss	ML-kNN	0.293	0.098	0.198	0.058	0.014	**0.011**
	BR	0.271	0.117	0.205	**0.017**	**0.012**	0.017
	CC	0.267	0.114	0.196	0.038	**0.012**	0.017
	CLR	0.257	0.096	0.191	0.018	**0.012**	0.012
	QWeightedML	0.262	0.095	0.191	0.026	**0.012**	0.012
	TSVM	0.259	0.095	0.191	0.022	**0.012**	0.012
	TSPCCA	**0.255**	**0.094**	**0.190**	0.018	**0.012**	0.012
	TSCCA	0.256	**0.094**	**0.190**	0.018	**0.012**	**0.011**
Average Precision	ML-kNN	0.693	0.851	0.758	0.844	0.348	0.265
	BR	0.707	0.818	0.754	0.907	0.575	0.303
	CC	0.713	0.815	0.757	0.916	0.576	0.298
	CLR	0.721	0.860	0.768	**0.963**	0.578	0.352
	QWeightedML	0.679	0.840	0.700	0.923	0.497	0.310
	TSVM	0.724	0.859	0.764	0.934	0.578	0.341
	TSPCCA	**0.732**	0.866	**0.771**	0.962	0.580	0.354
	TSCCA	0.726	**0.867**	**0.771**	**0.963**	**0.585**	**0.355**
Testing time (s)	ML-kNN	0.25	13.92	5.16	230	77	46
	BR	1.02	24.12	25.01	950	1370	35
	CC	1.06	25.05	25.12	988	1410	37
	CLR	2.56	66.15	104.34	6106	83358	2020
	QWeightedML	1.67	40.32	60.39	2534	4710	119
	TSVM	1.34	34.27	54.65	1135	1564	67
	TSPCCA	1.35	35.25	54.72	1143	1593	233
	TSCCA	1.40	36.68	58.42	1737	1742	328

0.01 and 0.1 to 1.0 with step 0.05 were considered for t.

In all classification problems the classifiers were trained using all available training samples and were evaluated by recognizing all test samples from the corresponding dataset. Table 2 gives the performance of each method on each of the datasets measured in terms of the two performance metrics and testing speed. The first column of the table lists the evaluation metrics, the second lists the compared methods, while the remaining columns show the performance of each method for every dataset. The best results per dataset in terms of the two performance metrics are shown in boldface. The testing times of each method are measured in seconds.

Table 2 show that among the seven tested approaches in terms of Hamming Loss, TSCCA and TSPCCA offer better predictive performance for the emotions, scene and the yeast datasets and similar results for the other three datasets compared to the competing methods. In terms of Average Precision the situation is more clear. For all six dataset, TSCCA and TSPCCA show slightly better predictive performance than CLR, while compared to the other methods, TSCCA and TSPCCA are 1% to 5% better than CC and BR, 2% to 20% better than ML-kNN and up to 8% better than QWeightedML. The results also show that for the six treated classification problems TSCCA and TSPCCA are 2 to 50 times faster (at prediction time) than CLR for all datasets. The proposed methods are 10% to 220% faster than the QWeightedML method for all datasets, except for the corel5k dataset where TSPCCA and TSCCA showed 2 and 2.5 times slower testing time, respectively. The testing time of the BR method is actually the same as the time spent in the testing process for the models located in the first layer of the proposed methods. The testing time of the CC method is slightly longer than the testing time of the BR method as a result of the increase in the feature vector size in each binary relevance model (classifier) of the classifier chain. Compared to TSVM, proposed methods show better predictive performance and slightly slower testing times.

The testing time of the ML-kNN method is shorter than the testing time of the problem transformation methods. The intent of including algorithm adaptation method, as ML-kNN, in the experiments was to show that the proposed methods, that belong to the group of problem transformation methods, can achieve comparable or even better predictive accuracy. However, their performance and computational complexity depend strongly on the type of the base classifier that is used for solving the partial classification problems.

TSCCA and TSPCCA show similar predictive performance for the six multi-label classification problems. Statistically, TSCCA shows slightly better results than TSPCCA only for Average Precision. In terms of testing time, TSPCCA is significantly better than TSCCA.

The values of the reduction parameter (r) obtained by equation 5 are shown in Table 1. It is interesting to be noticed that for smaller values of the reduction parameter r the testing times of TSCCA and TSPCCA approach to the testing time of the binary relevance models (the models of the first layer of the architecture).

5. CONCLUSION

A two stage classifier chain architecture (TSCCA) for efficient pair-wise multi-label learning and its pruned modification (TSPCCA) were presented. The performances of these architectures were compared with the CLR method with the majority voting strategy, QWeightedML, classifier chains, binary relvance, multi-label kNN and the two stage voting method on six different real-world datasets. The results showed that the TSCCA and its modification TSPCCA outperform CLR in terms of predictive accuracy. TSCCA and TSPCCA also show significantly better predictive performances than the other compared methods. In terms of testing speed TSCCA and TSPCCA show similar testing times and were 2 to 50 times faster than CLR and up to 2.5 times faster than the QWeightedML method. Comparing to the CC method and the BR method, TSCCA and TSPCCA show better predictive performance, while their testing times are always bigger as a result of the testing time of the models of the second layer of the architectures.

6. REFERENCES

[1] K. Brinker, J. Furnkranz, and E. Hullermeie, "A unified model for multilabel classification and ranking," in *Proc. of the 17th European conference on artificial intelligence, Riva Del Garda, Italy*, 2006, pp. 489–493.

[2] G. Tsoumakas and I. Katakis, "Multi label classification: An overview," *International Journal of Data Warehousing and Mining*, , no. 3, 2007.

[3] M. L. Zhang and Z. H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern Recognition*, , no. 40, pp. 2038–2048, 2007.

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

[5] M. L. Zhang and Z. H. Zhou, "Multi-label neural networks with applications to functional genomics and text categorization," *IEEE Transactions on Knowledge and Data Engineering*, , no. 18, pp. 1338–1351, 2006.

[6] R. E. Schapire and Y. Singer, "Boostexter: a boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2/3, pp. 135–168, 2000.

[7] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in *Proc of the ECML/PKDD, Bled, Slovenia*, 2009, pp. 254–269.

[8] E. Loza Mencia, S. H. Park, and J. Furnkranz, "Efficient voting prediction for pairwise multi-label classification," *Neurocomputing*, vol. 73, pp. 1164–1176, 2010.

[9] Gj. Madjarov, D. Gjorgjevikj, and T. Delev, "Efficient Two Stage Voting Architecture for Pairwise Multi-label Classification," in *AI 2010: Advances in Artificial Intelligence*, Jiuyong Li, Ed., vol. 6464 of *LNCS*, chapter 17, pp. 164–173. Berlin, Heidelberg, 2011.

[10] J. Furnkranz, E. Hullermeier, E. Loza Mencia, and K. Brinker, "Multi-label classification via calibrated label ranking," *Machine Learning*, vol. 73, no. 2, pp. 133–153, 2008.

[11] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas, "Multilabel classification of music into emotions," in *Proc. of International Conference on Music Information Retrieval, Philadelphia, PA, USA*, 2008, pp. 320–330.

[12] M.R. Boutell, J. Luo, X. Shen, and C.M. Brown, "Learning multi-labelscene classiffication," *Pattern Recognition*, vol. 9, no. 37, pp. 1757–1771, 2004.

[13] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," *Advances in Neural Information Processing Systems*, vol. 14, 2001.

[14] A. Srivastava and B. Zane-Ulman, "Discovering recurring anomalies in text reports regarding complex space systems," in *Proc. of the IEEE Aerospace Conference*. 2005, pp. 55–63, Morgan Kaufmann.

[15] I. Katakis, G. Tsoumakas, and I. Vlahavas, "Multilabel text classification for automated tag suggestion," in *Proc. of the ECML/PKDD 2008 Discovery Challenge, Antwerp, Belgium*, 2008.

[16] P. Duygulu, K. Barnard, N. de Freitas, and D. Forsyth, "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," in *Proc. of the 7th European Conference on Computer Vision*, 2002, pp. 97–112.

[17] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten, "The weka data mining software: An update, software available at http://www.cs.waikato.ac.nz/ml/weka/," *SIGKDD Explorations*, vol. 11, no. 1, 2009.

[18] C.C. Chang and C.J. Lin, "Libsvm: a library for support vector machines, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm/," 2001.