# Advanced Topics in Multi-label Learning


Weiwei Liu

Faculty of Engineering and Information Technology

University of Technology Sydney

A thesis submitted for the degree of

*Doctor of Philosophy*

July, 2017

# Certificate of Original Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Student: Weiwei Liu
Date: 24/07/2017

I would like to dedicate this thesis to my loving grandparents and wife.

# Acknowledgements

I would like to express my deepest gratitude to my supervisor Prof. Ivor W.Tsang for his patient and valuable guidance. I really appreciate that Prof. Ivor W.Tsang provided me an opportunity to do research under his supervision, which means the change point to me and my life. I knew very little about machine learning and research when I follow Prof. Ivor W.Tsang. It was Prof. Ivor W.Tsang who taught me how to do research, how to find interesting ideas, how to develop fancy models and algorithms, how to write technical papers and how to become an independent researcher all from scratch. He gave me too much patience, and was very willing to teach everything he knows to me. I remember that Prof. Ivor W.Tsang even helped me to fix the bugs for my first research project, and rewrite the technical papers. Without his illuminating instructions, insightful inspiration, consistent encouragement, and expert guidance, I would not have published papers on the leading journals or conferences in my research field. Therefore, I feel very lucky to be supervised by Prof. Ivor W.Tsang.

I am also greatly indebted to the Centre for Quantum Computation & Intelligent Systems (QCIS) directed by Prof. Chengqi Zhang. QCIS has supported me to attend many prestigious international conferences, such as AAAI and NIPS. I really thank QCIS for the support. I also want to express my gratitude to all the students in QCIS.

Last but not the least, I also want to express my deepest gratitude to my wife, Xiuwen Gong. She has accompanied me for six years poor life. She never complains, and always gives me too much encouragement and patience. She is very smart. I like to talk with my wife about my problems, and she always gives me inspirations. During these years, we met with many problems. But, I still feel happiness. Without her support and patience, I can not make any achievements, and also can not live a happy life. I feel extremely grateful for my wife's consistently supporting, encouraging and caring for me all of my life!

# Abstract

Multi-label learning, in which each instance can belong to multiple labels simultaneously, has significantly attracted the attention of researchers as a result of its wide range of applications, which range from document classification and automatic image annotation to video annotation.

Many multi-label learning models have been developed to capture label dependency. Amongst them, the classifier chain (CC) model is one of the most popular methods due to its simplicity and promising experimental results. However, CC suffers from three important problems: Does the label order affect the performance of CC? Is there any globally optimal classifier chain which can achieve the optimal prediction performance for CC? If yes, how can the globally optimal classifier chain be found? It is non-trivial to answer these problems. Another important branch of methods for capturing label dependency is encoding-decoding paradigm. Based on structural SVMs, maximum margin output coding (MMOC) has become one of the most representative encoding-decoding methods and shown promising results for multi-label classification. Unfortunately, MMOC suffers from two major limitations: 1) Inconsistent performance: D. McAllester has already proved that structural SVMs fail to converge on the optimal decoder even with infinite training data. 2) Prohibitive computational cost: the training of MMOC involves a complex quadratic programming (QP) problem over the combinatorial space, and its computational cost on the data sets with many labels is prohibitive. Therefore, it is non-trivial to break the bottlenecks of MMOC, and develop efficient and consistent algorithms for solving multi-label learning tasks. The prediction of most multi-label learning methods either scales linearly with the number of labels or involves an expensive decoding process, which usually requires solving a combinatorial optimization. Such approaches become unacceptable when tackling thousands of labels, and are impractical for real-world applications, such as document annotation. It is imperative to design an efficient, yet accurate multi-label learning algorithm with the minimum number of predictions. This thesis systematically studies how to efficiently solve aforementioned issues with provable guarantee.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

During the past decade, multi-label classification has become a popular machine learning paradigm as its modern wide applications, such as protein function classification, music categorization and semantic scene classification. In this chapter, we briefly introduce the background of multi-label learning, related advanced topics, thesis contributions, and finally present the organization of the entire thesis.

## 1.1 Background

Traditional supervised learning is one of the most important machine learning paradigms, where each example is represented by a feature vector and associated with a single label from a set of disjoint labels $\mathcal{L}$ ($\mathcal{L} > 1$). If $\mathcal{L} = 2$, the learning problem is called a binary classification problem. If $\mathcal{L} > 2$, then it is called a multiclass classification problem.

Because real-world objects may be very complicate and have multiple semantic meanings simultaneously, there are many learning problems where the above settings do not fit well. For example, an image as shown in Figure 1.1 may have *cloud*, *tree* and *sky* tags Boutell et al. [2004]; Huang et al. [2016]; a document can be associated with a range of topics, such as *Sports*, *Finance* and *Education* Schapire and Singer [2000a]; a gene belongs to the functions of *protein synthesis*, *metabolism* and *transcription* Barutcuoglu et al. [2006].

To deal with the multiple semantic settings that each instance may involve, one can assign a set of proper labels, which represents multiple semantic meanings, to the instances. Then, multi-label learning paradigm Tsoumakas et al. [2009, 2010] emerges based on aforementioned real applications. In contrast to traditional supervised learning, each instance can belong to a set of labels instead of a single label simultaneously in multi-label learning. The learning task is to learn a function which is able to predict the proper label sets for unseen instances.

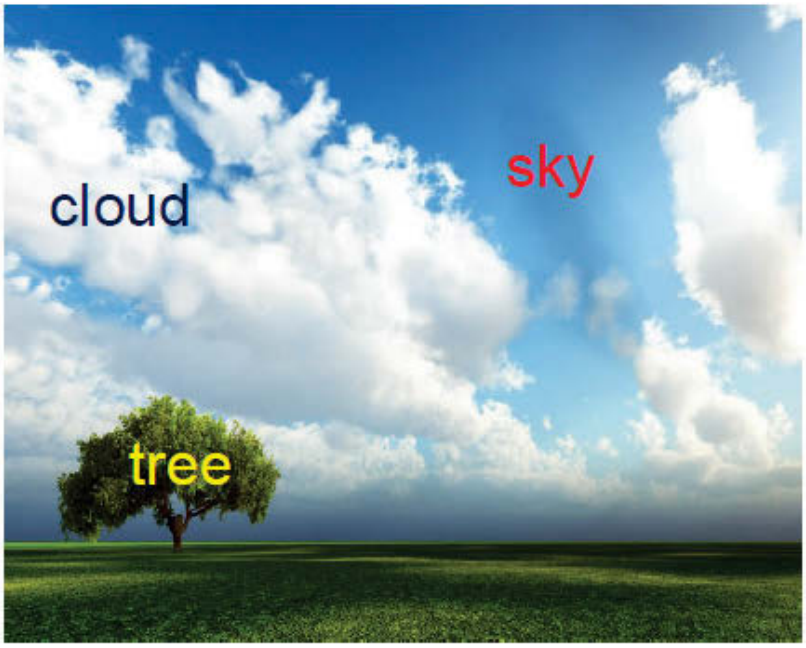


Figure 1.1: The illustration of image annotation, where an image may have *cloud*, *tree* and *sky* tags.

Multi-label learning has been applied many modern wide applications. We show the following applications.

With the development of considerable videos on the Internet (e.g., Youtube, Flickr and Facebook) [Liu et al., 2015a,b], efficient and effective indexing and searching these video corpus becomes more and more important for the research and industry community. Currently, semantic-level video annotation (i.e., the semantic video concept detection) has been an important research topic in the multimedia research community, which aims to tag videos with a set of concepts of interest, including scenes (e.g., garden, sky, tree), objects (e.g., animals, people, airplane, car), events (e.g., election, ceremony) and certain named entities (e.g., university, person, home). Recently, much effort have been made on annotating video concepts in a generic fashion. For example, Naphade et al. [2006] develop a large scale concept ontology for generic video annotation and Snoek et al. [2006] build an ontology of 101 concepts from News video as well. In contrast to the generic video annotation algorithms, Qi et al. [2008] focus on a multi-label video annotation setting, where a video can belong to multiple labels at the same time. Qi et al. [2008] attempt to capture the correlations between different labels to improve the annotation performance on generic

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Outdoor | T | T | T | T | T | T | T |
| Face | T | T | T | T | T | T | T |
| Person | T | T | T | T | T | T | T |
| People-Marching | F | F | F | T | T | F | F |
| Road | T | T | T | T | T | T | T |
| Walking_running | T | T | T | T | T | T | T |

Figure 1.2: Some multi-labeled examples from TRECVID data set. T and F represent the positive and negative labels for corresponding concepts respectively.

video concepts. In many real-world video corpus, the videos are multi-labeled. For instance, most of the videos in the popular TRECVID data set Snoek et al. [2006] are annotated by more than one label from a set of 39 different concepts. Figure 1.2 illustrates that the videos belong to multiple labels. For example, a video can be classified as “outdoor”, “face” and “road” at the same time.

Recently, the development of green computing and energy efficient 5G applications has become one of the most important topics in communications Wunder et al. [2014]. Under this field, advanced high performance algorithms for mobile applications have attracted the attention of researchers Han et al. [2015, 2017]. Recommendation systems are widely used to predict the “rating” or “preference” that a user would give to an item. A good recommendation system with high performance is able to attract users to the service for 5G applications. Guo et al. [2016] focus on high performance multi-label classification methods and their applications for medical recommendations in the domain of 5G communication.

Implementing the improvement of situational awareness play a vital role for making decision in urban emergency management. Recently, bystanders and the city infrastructure provide most information for urban emergency management. However, there is no standard way of collecting information from bystanders, and the city infrastructure is built from cost intensive sensors. Both bystanders and the city infrastructure have their limitations and disadvantages. Therefore,it is imperative to utilize other information sources for obtaining incident information to improve situational awareness. With the fast development of social media, microblogs and twitter have shown as valuable information source during incidents, such as real-time detection of earthquakes Sakaki et al. [2010], tracking of diseases Signorini et al. [2011], as well as the detection of fires and floods Vieweg et al. [2010]. These applications have already demonstrated the value of microblogs and twitter in the course of crisis mitigation. Schulz et al. [2016] study small-scale incident reporting

behavior with microblogs, and employ multi-label classification of tweets to evaluate the rapid prototyping capabilities and usefulness of the framework.

Effective and consistent segmentation of brain white matter bundles at neonatal stage plays a vital role in detecting white matter abnormalities and understanding brain development for the prediction of psychiatric disorders. Because the complexity of white matter anatomy and the spatial resolution of diffusion-weighted MR imaging, multiple fiber bundles can pass through one voxel. Ratnarajah and Qiu [2014] aim to to assign one or multiple anatomical labels of white matter bundles to each voxel to reflect complex white matter anatomy of the neonatal brain. To achieve this goal, Ratnarajah and Qiu [2014] explore the supervised multi-label learning algorithm in Riemannian diffusion tensor spaces, which considers diffusion tensors lying on the Log-Euclidean Riemannian manifold of symmetric positive definite (SPD) matrices and their corresponding vector space as feature space. Ratnarajah and Qiu [2014] demonstrate that they are able to automatically learn the number of white matter bundles at a location and provide anatomical annotation of the neonatal white matter. Moreover, Ratnarajah and Qiu [2014] also develop the binary mask for individual white matter bundles to facilitate tract-based statistical analysis in clinical studies. Finally, Ratnarajah and Qiu [2014] apply their method to automatically segment 13 white matter bundles of the neonatal brain and examine the segmentation accuracy against semi-manual labels derived from tractography.

We summarize the applications in Table 1.1.

A comprehensive review of multi-label learning can be found in Zhang and Zhou [2014] and references therein.

## 1.2 Advanced Topics

Many multi-label learning models [Chen and Lin, 2012b; Dembczynski et al., 2010; Gong et al., 2017; Guo and Gu, 2011; Huang and Zhou, 2012; Kang et al., 2006; Liu and Tsang, 2015a; Read et al., 2009; Zhang and Schneider, 2012] have been developed to capture label dependency. Amongst them, classifier chain (CC) Read et al. [2009] and maximum margin output coding (MMOC) Zhang and Schneider [2012] are two most popular methods due to their simplicity and promising experimental results Read et al. [2009]; Zhang and Schneider [2012]. Unfortunately, both CC and MMOC suffer from some major limitations. Moreover, the prediction of aforementioned multi-label learning methods either scales linearly with the number of labels or involves an expensive decoding process, which usually requires solving a combinatorial optimization. Such approaches become unacceptable when tackling thousands of labels, and are impractical for real-world applications, such as document annotation.

This thesis systematically investigates the following three advanced topics in multi-label learning:

Table 1.1: The applications of multi-label learning.

| Reference | Applications |
|---|---|
| Boutell et al. [2004] | automatic image annotation |
| Zhang and Zhou [2006] | document classification |
| Barutcuoglu et al. [2006] | gene function prediction |
| Qi et al. [2008] | automatic video annotation |
| Guo et al. [2016] | mobile medical recommendations |
| Bucak et al. [2010] | visual object recognition |
| Schulz et al. [2016] | social network analysis |
| Venkatesan et al. [2016] | high-speed streaming data |
| Shao et al. [2015] | quantitative structure-activity relationship models |
| Ciarelli et al. [2014] | web page categorization |
| Ratnarajah and Qiu [2014] | neonatal brains |
| Wan et al. [2014] | protein subcellular localization |
| He et al. [2012] | visual mobile robot navigation |
| Neagu et al. [2005] | visual arts data mining |
| Grady and Funka-Lea [2004] | image segmentation |

### 1.2.1 Underlying Problems Behind CC

CC suffers from three important problems: Does the label order affect the performance of CC? Apparently yes, because different classifier chains involve different classifiers trained on different training sets. Thus, to reduce the influence of the label order, Read et al. [2009] proposed the ensembled classifier chain (ECC) to average the multi-label predictions of CC over a set of random chain ordering. Since the performance of CC is sensitive to the choice of label order, there is another important question:Is there any globally optimal classifier chain which can achieve the optimal prediction performance for CC? If yes, how can the globally optimal classifier chain be found? It is non-trivial to answer these problems.

### 1.2.2 Major Limitations of MMOC

Based on structural SVMs McAllester [2006]; Tsochantaridis et al. [2005], MMOC has become one of the most representative encoding-decoding methods and shown promising results for multi-label classification. Unfortunately, MMOC suffers from two major limitations: 1) Inconsistent performance: McAllester [2006] has already proved that structural SVMs fail to converge on the optimal decoder even with infinite

training data. 2) Prohibitive computational cost: the training of MMOC involves a complex quadratic programming (QP) problem over the combinatorial space, and its computational cost on the data sets with many labels is prohibitive. Therefore, it is non-trivial to break the bottlenecks of MMOC, and develop efficient and consistent algorithms for solving multi-label learning tasks.

### 1.2.3 Scalability of Prediction

One central challenging issue for practical multi-label learning is the scalability of prediction. Suppose one needs to predict the presence or absence of $q$ labels for $m$ input instances, simple approaches, like Read et al. [2009]; Tsoumakas et al. [2010], scaling the number of predictions to be linear with the number of labels, take $\mathcal{O}(q \times m)$ time for the prediction. However, in real-world applications, like image annotation Deng et al. [2009], $m$ and $q$ can be very large and the cost of those exhaustive approaches quickly becomes prohibitive. We cannot get the results of these exhaustive approaches on the large-scale data sets within one week.

## 1.3 Thesis Contributions

The main contributions of this thesis can be summarized in following three parts.

### 1.3.1 Underlying Problems Behind CC

*Contribution:*

- This thesis first generalizes the CC model over a random label order. Then, we present a theoretical analysis of the generalization error for the proposed generalized model.
- Based on our results, we propose a dynamic programming based classifier chain (CC-DP) algorithm to search the globally optimal label order for CC and a greedy classifier chain (CC-Greedy) algorithm to find a locally optimal CC. Furthermore, we propose Tree-DP and Tree-Greedy algorithms to further speed up CC-DP and CC-Greedy, respectively.

*Outcome:*

- This contribution was published in NIPS-2015 Liu and Tsang [2015b].
- The extension of this work is currently under the review by JMLR.

### 1.3.2 Major Limitations of MMOC

*Contribution:*

- To avoid the inconsistent performance of MMOC, we present a novel large margin metric learning paradigm for multi-label classification. Our theoretical analysis shows that our proposed model converges to the optimal solutions, and also reduces the generalization error for multi-label classification.
- To incorporate the feature and label correlations, we project both the input and output to the same embedding space, in which the input and output can be compared. A large margin formulation with $k$ nearest neighbor constraints is proposed to learn the embedding space. Lastly, we transform the formulation to metric learning Kulis [2013]; Yang and Jin [2006] problems.
- After transformation, our optimization problem is reduced to a semidefinite programming problem. To handle many outputs, an accelerated proximal gradient (APG) method Beck and Teboulle [2009]; Toh and Yun [2009] is adapted to solve the resultant optimization problem.
- To avoid the expensive decoding step, we select $k$ nearest neighbors from the training set for each testing instance in the embedding space and make a rapid prediction based on the labels of those $k$ nearest neighbors.

*Outcome:*

- This contribution was published in AAAI-2015 Liu and Tsang [2015a].
- The extension of this work is currently under the review by TPAMI.

### 1.3.3 Scalability of Prediction

*Contribution:*

- We observe three important phenomena of real-world multi-label data sets.
- To reduce the number of predictions, based on the properties of multi-label data sets, we employ the coding tree model to deal with both the transformed multi-class classification task and multi-label classification problems in bottom-up and top-down manners, respectively.
- We provide a theoretical analysis of the average number of predictions for the both situations and the testing time complexity analysis.

*Outcome:*

- This contribution is currently accepted by JMLR-2017 Liu and Tsang [2017].

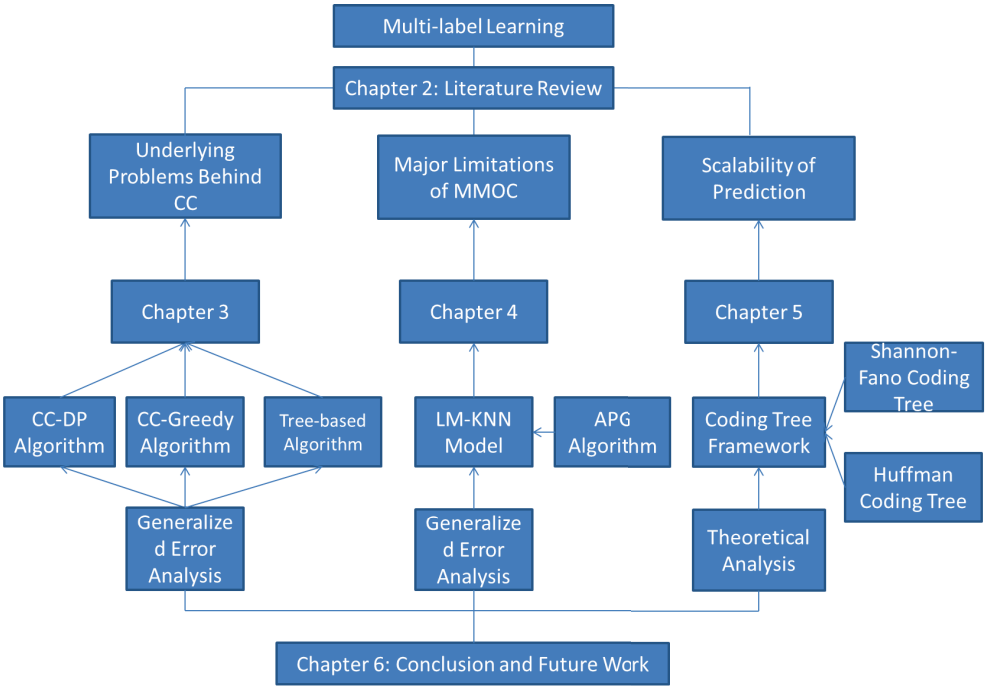


Figure 1.3: The organization of this thesis.

## 1.4 Thesis Outline

This thesis systematically studies the underlying problems behind CC and MMOC and provides the solutions with provable guarantee. Then, we focus on designing efficient, yet accurate multi-label learning algorithms with the minimum number of predictions. This thesis is organized as follows:

Chapter 2 introduces the related work.

Chapter 3 studies the underlying problems behind CC and provides the answers with provable guarantee.

Chapter 4 addresses the major limitations of MMOC.

Chapter 5 introduces the coding tree framework for multi-label prediction.

Chapter 6 concludes this thesis and also presents the possible future works.

The organization of this thesis is shown in Figure 1.3.

## 1.5 Publications

1. **Weiwei Liu**, and Ivor W.Tsang, Sparse Perceptron Decision Tree for Millions of Dimensions, *AAAI Conference on Artificial Intelligence (AAAI)*, 2016: 1881-1887.

2. **Weiwei Liu**, and Ivor W.Tsang, On the Optimality of Classifier Chain for Multi-label Classification, *Advances in Neural Information Processing Systems (NIPS)*, 2015: 712-720.

3. **Weiwei Liu**, and Ivor W.Tsang, Large Margin Metric Learning for Multi-Label Prediction, *AAAI Conference on Artificial Intelligence (AAAI)*, 2015: 2800-2806.

4. **Weiwei Liu**, and Ivor W. Tsang, Making Decision Trees Feasible in Ultrahigh Feature and Label Dimensions, *Journal of Machine Learning Research (JMLR)*, Accept.

5. **Weiwei Liu**, Ivor W. Tsang, and Klaus-Robert Müller, An Easy-to-hard Learning Paradigm for Multiple Classes and Multiple Labels, *Journal of Machine Learning Research (JMLR)*, under review.

# Chapter 2

# Literature Review

In multi-label learning, we shall receive a feature vector $\mathbf{x}_i \in \mathbb{R}^d$ and a corresponding label set $\mathcal{Y}_i$, where $\mathcal{Y}_i \subseteq \{\lambda_1, \lambda_2, \cdots, \lambda_q\}$. For simplicity, $\mathbf{y}_i \in \{0,1\}^q$ is used to represent the label set $\mathcal{Y}_i$, where $\mathbf{y}_i(j) = 1$ if and only if $\lambda_j \in \mathcal{Y}_i$, for any integer $j : 1 \leq j \leq q$. Given the training data set $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$, the goal of multi-label learning is to learn a multi-label classifier $f : \mathbb{R}^d \to \{0,1\}^q$ that accurately predicts the label vector for any unseen instances. Figure 2.1 shows an example of multi-label samples with five labels.

Tsoumakas et al. [2010] group existing multi-label classification methods into two major categories: *algorithm adaptation* (AA) or *problem transformation* (PT). This chapter first introduces a broad branch of methods, which belong to AA and PT categories. Then, we focus on the review of the advanced topics involved in this thesis. Figure 2.16 summarizes the categorization of representative AA and PT algorithms reviewed in this Chapter.

## 2.1 Problem Transformation

This category of algorithms aim to transform multi-label learning problems into some other well-established learning settings. Some works, such as Boutell et al. [2004] and Chen et al. [2007], have transformed a multi-label data set into a single-label data set with the same set of labels, and then trained a single-label classifier for the transformed data set. According to Tsoumakas et al. [2010], the *copy* transformation replaces each multi-label sample $\mathbf{x}_i, \mathcal{Y}_i$ with $|\mathcal{Y}_i|$ ($|\mathcal{Y}_i|$ means the cardinality of $\mathcal{Y}_i$ ) samples : $(\mathbf{x}_i, \lambda_j)$, if $\lambda_j \in \mathcal{Y}_i$. *copy-weight* is a variation of *copy* transformation. It assigns a weight of $\frac{1}{|\mathcal{Y}_i|}$ to each of the reduced samples. The *select-max* transformation chooses one label from $\mathcal{Y}_i$ with the most frequency among all samples to replace $\mathcal{Y}_i$. The *select-min* transformation chooses one label from $\mathcal{Y}_i$ with the least frequency among all samples to replace $\mathcal{Y}_i$. The *select-random* transformation randomly chooses one label from $\mathcal{Y}_i$

| Sample | Feature | Label Set |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | $\{\lambda_1, \lambda_2, \lambda_5\}$ |
| 2 | $\boldsymbol{x}_2$ | $\{\lambda_3, \lambda_5\}$ |
| 3 | $\boldsymbol{x}_3$ | $\{\lambda_2, \lambda_3, \lambda_5\}$ |
| 4 | $\boldsymbol{x}_4$ | $\{\lambda_5\}$ |
| 5 | $\boldsymbol{x}_5$ | $\{\lambda_1, \lambda_4\}$ |
| 6 | $\boldsymbol{x}_6$ | $\{\lambda_4\}$ |
| 7 | $\boldsymbol{x}_7$ | $\{\lambda_1, \lambda_3\}$ |
| 8 | $\boldsymbol{x}_8$ | $\{\lambda_1, \lambda_3, \lambda_4, \lambda_5\}$ |

Figure 2.1: An example of multi-label samples with five labels.

to replace $\mathcal{Y}_i$. The *ignore* transformation simply removes every multi-labelled sample. Figures 2.2, 2.3, 2.4, 2.5, 2.6 and 2.7 demonstrate the various transformations on the data set of Figure 2.1.

Binary relevance (BR) Tsoumakas et al. [2010] is one of the most popular problem transformation methods, which learns $q$ binary classifiers independently for each different label in $\{\lambda_1, \lambda_2, \cdots, \lambda_q\}$. BR transforms the original data set into $q$ data sets that contain all examples of the original data set. The instances in the $j$-th data set are positively labeled if the label set of these instances contain $\lambda_j$ and negatively labeled otherwise.

Given a new testing instance, BR uses $q$ classifiers to predict all the labels. Based on the data set of Figure 2.1, Figures 2.8, 2.9, 2.10, 2.11 and 2.12 show the five data sets that are constructed by BR.

Label Powerset (LP) Tsoumakas et al. [2010] is an effective problem transformation methods, which reduces a multi-label classification problem into a multiclass prediction problem by treating each distinct label set as one of the classes for a transformed multi-class learning task. Figure 2.13 shows the transformed data set of Figure 2.1 using LP.

Given a new testing instance, LP exploits the multiclass classifier to output the most probable class, which is actually a set of labels. Read [2008] proposes an extended LP to output a probability distribution over all classes. Based on the data set of Figure 2.1, Figure 2.14 shows an example of a probability distribution over all classes produced by LP. We can calculate the sum of the probabilities for each label to obtain a ranking among the labels.

Based on LP, Read [2008] also proposes the pruned problem transformation (PPT) method to remove label sets that appear less than a pre-defined threshold and replaces their information by introducing disjoint subsets of these label sets that do exist more times than the threshold.

Tsoumakas and Vlahavas [2007a] present the random $k$-labelsets (RKL) method to

combine ensemble learning Zhou [2012] with LP for multi-label learning. Random $k$-labelsets ensembles a number of LP classifiers, where each LP classifier is trained using a different small random subset of the set of label, to make the prediction. Random $k$-labelsets is able to capture the label correlations.

Ranking by pairwise comparison (RPC) Hüllermeier et al. [2008] reduces the multi-label data set into $\frac{q(q-1)}{2}$ binary label data sets, where each binary data set contains the samples that belong to at least one of the two corresponding labels, but not both. Then, RPC trains a binary classifier for each reduced data set. For the testing, RPC makes use of all the classifiers for the prediction of a new testing instance, and then obtains a ranking for each label by voting. Motivated by RPC, Loza Mencía and Fürnkranz [2008a] develop a multi-label pairwise perceptron (MLPP) algorithm to use perceptrons for the binary classification tasks.

Calibrated label ranking (CLR) Loza Mencía and Fürnkranz [2008a] employs the techniques of pairwise comparison to conduct the ranking between labels, and transform the multi-label learning problems into the label ranking problems. Specifically, CLR extends RPC by introducing an additional virtual label, which acts as a natural breaking point of the ranking into relevant and irrelevant sets of labels. Assume that each instance which belongs to a given label is considered as positive for this label and negative for the virtual label, while each instance which does not belong to a given label is considered negative for this label and positive for the virtual label. Then, CLR learns binary models to classify the virtual label and each of the other labels.

Zhang and Zhou [2007a] hypothesize that if the inherent ambiguity can be explicitly expressed in the input space appropriately, the problem of multi-label learning can be solved more effectively. Then, this work proposes a novel multi-label learning approach, INSDIF, to verify this hypothesis. INSDIF aims to perform instance differentiation that transforms an example into a bag of instances each of which reflects the example's relationship with one of the possible classes. Specifically, INSDIF builds a prototype vector for each label, by averaging all instances of the training set that belong to this label. After that, every instance is transformed to a bag of $q$ instances, each of which equal to the difference between the initial instance and one of the prototype vectors. In this way, INSDIF directly addresses the inherent ambiguity of each example in the input space. Furthermore, a two level classification strategy is used to learn from the transformed data set.

Classifier chain is one of the most important PT methods, which will be introduced in Chapter 2.3.

## 2.2 Algorithm Adaptation

ML-$k$NN Zhang and Zhou [2007b] is one of the most popular AA methods. The main idea of ML-$k$NN is to adapt $k$ nearest neighbor techniques to deal with multi-

label learning problems, where maximum a posteriori (MAP) rule is utilized to make prediction by reasoning with the Euclidean distance between the $k$ nearest neighbors and the testing instance, and the labeling information embodied in the neighbors.

The C4.5 algorithm is also adapted in Clare and King [2001] for dealing with multi-label learning problems. In specific, Clare and King [2001] calculate the modified entropy and assign multiple labels in the leave node of the tree.

Based on AdaBoost, Schapire and Singer [2000b] present AdaBoost.MH and AdaBoost.MR for multi-label classification. AdaBoost.MH aims to minimize Hamming loss, while AdaBoost.MR is designed to find a hypothesis which places the correct labels at the top of the ranking.

Motivated by the production of multi-label models that can be easily explained and understood by humans, Comité et al. [2003] combine AdaBoost.MH and propose an algorithm for producing alternating decision trees (ADT) for multi-label classification.

Based on independent word-based representation, known as Bag-of-Words (BOW) representation Dumais et al. [1998], Ueda and Saito [2002] propose two types of probabilistic generative models for multi-label learning, called parametric mixture models (PMM1, PMM2), where PMM2 is a more flexible version of PMM1. PMMs assume that multi-labeled text has a mixture of characteristic words appearing in single-labeled text that belong to each category of the multi-categories. Based on this assumption, Ueda and Saito [2002] develop generative models with a good feature: the objective function of PMM1 is convex. Moreover, they also present efficient learning and prediction algorithms for PMMs. Streich and Buhmann [2008] present a deconvolution approach to estimate the individual contribution of each label to a given item.

Ghamrawi and McCallum [2005] explore multi-label conditional random field (CRF) classification models that directly parameterize label co-occurrences in multi-label classification. In specific, collective multi-label classifier (CML) aims to capture co-occurrence patterns among labels, while collective multi-label with features classifier (CMLF) learns parameters for feature-label-label triples-capturing the impact that an individual feature has on the co-occurrence probability of a pair of labels.

Zhang and Zhou [2006] propose the first neural network algorithm, named back-propagation for multi-label learning (BP-MLL), for multi-label learning. Derived from the popular back-propagation algorithm Rumelhart et al. [1986], BP-MLL introduces a novel error function to capture the characteristics of multi-label learning, that is, the labels belonging to an instance should be ranked higher than those not belonging to that instance.

Rank-SVM Elisseeff and Weston [2001] adapt the maximum margin strategy to deal with multi-label data, where a set of linear classifiers are optimized to minimize the empirical ranking loss and enabled to handle nonlinear cases with kernel tricks Liu et al. [2015c]. Rank-SVM is also able to provide a way of controlling the time complexity while having a small empirical error.

| Sample | Feature | Label Set |
|---|---|---|
| 1a | $\boldsymbol{x}_1$ | $\{\lambda_1\}$ |
| 1b | $\boldsymbol{x}_1$ | $\{\lambda_2\}$ |
| 1c | $\boldsymbol{x}_1$ | $\{\lambda_5\}$ |
| 2a | $\boldsymbol{x}_2$ | $\{\lambda_3\}$ |
| 2b | $\boldsymbol{x}_2$ | $\{\lambda_5\}$ |
| 3a | $\boldsymbol{x}_3$ | $\{\lambda_2\}$ |
| 3b | $\boldsymbol{x}_3$ | $\{\lambda_3\}$ |
| 3c | $\boldsymbol{x}_3$ | $\{\lambda_3\}$ |
| 4 | $\boldsymbol{x}_4$ | $\{\lambda_5\}$ |
| 5a | $\boldsymbol{x}_5$ | $\{\lambda_1\}$ |
| 5b | $\boldsymbol{x}_5$ | $\{\lambda_4\}$ |
| 6 | $\boldsymbol{x}_6$ | $\{\lambda_4\}$ |
| 7a | $\boldsymbol{x}_7$ | $\{\lambda_1\}$ |
| 7b | $\boldsymbol{x}_7$ | $\{\lambda_3\}$ |
| 8a | $\boldsymbol{x}_8$ | $\{\lambda_1\}$ |
| 8b | $\boldsymbol{x}_8$ | $\{\lambda_3\}$ |
| 8c | $\boldsymbol{x}_8$ | $\{\lambda_4\}$ |
| 8d | $\boldsymbol{x}_8$ | $\{\lambda_5\}$ |

Figure 2.2: The illustration of *copy* transformation.

Maximum margin output coding is one of the most important AA methods, which will be introduced in Chapter 2.4.

## 2.3 Classifier Chain

To capture label dependency, various approaches attempt to exploit the different orders (first-order, second-order and high-order) of label correlations Zhang and Zhang [2010]. For example, Kang et al. [2006] explicitly exploit high-order correlation between labels, but this involves an optimization problem with an exponential number of constraints. All these methods assume that the correlations are shared by all instances, thus Huang and Zhou [2012] try to exploit label correlations in the data locally and measure the similarity between instances in the label space rather than in the feature space. However, the label space is usually sparse when there are many labels, making it impossible to obtain accurate similarity between instances through the measurement in the label space.

Some other works also try to provide a probabilistic interpretation for label correlations. For example, Guo and Gu Guo and Gu [2011] model the label correlations

| Sample | Feature | Label Set | Weight |
|---|---|---|---|
| 1a | $\boldsymbol{x}_1$ | $\{\lambda_1\}$ | 0.33 |
| 1b | $\boldsymbol{x}_1$ | $\{\lambda_2\}$ | 0.33 |
| 1c | $\boldsymbol{x}_1$ | $\{\lambda_5\}$ | 0.33 |
| 2a | $\boldsymbol{x}_2$ | $\{\lambda_3\}$ | 0.50 |
| 2b | $\boldsymbol{x}_2$ | $\{\lambda_5\}$ | 0.50 |
| 3a | $\boldsymbol{x}_3$ | $\{\lambda_2\}$ | 0.33 |
| 3b | $\boldsymbol{x}_3$ | $\{\lambda_3\}$ | 0.33 |
| 3c | $\boldsymbol{x}_3$ | $\{\lambda_3\}$ | 0.33 |
| 4 | $\boldsymbol{x}_4$ | $\{\lambda_5\}$ | 1.00 |
| 5a | $\boldsymbol{x}_5$ | $\{\lambda_1\}$ | 0.50 |
| 5b | $\boldsymbol{x}_5$ | $\{\lambda_4\}$ | 0.50 |
| 6 | $\boldsymbol{x}_6$ | $\{\lambda_4\}$ | 1.00 |
| 7a | $\boldsymbol{x}_7$ | $\{\lambda_1\}$ | 0.50 |
| 7b | $\boldsymbol{x}_7$ | $\{\lambda_3\}$ | 0.50 |
| 8a | $\boldsymbol{x}_8$ | $\{\lambda_1\}$ | 0.25 |
| 8b | $\boldsymbol{x}_8$ | $\{\lambda_3\}$ | 0.25 |
| 8c | $\boldsymbol{x}_8$ | $\{\lambda_4\}$ | 0.25 |
| 8d | $\boldsymbol{x}_8$ | $\{\lambda_5\}$ | 0.25 |

Figure 2.3: The illustration of *copy-weight* transformation.

using a conditional dependency network; PCC Dembczynski et al. [2010] exploits a high-order Markov Chain model to capture the correlations between the labels and provide an accurate probabilistic interpretation of classifier chain (CC). Other works Huang and Zhou [2012]; Kang et al. [2006]; Read et al. [2009] focus on modeling the label correlations in a deterministic way, and CC is one of the most popular methods among them.

Similar to BR, the CC model Read et al. [2009] trains $q$ binary classifiers $h_j$ ($j \in \{1, \cdots, q\}$). Classifiers are linked along a chain where each classifier $h_j$ deals with the binary classification problem for label $\lambda_j$. The augmented vector $\{\mathbf{x}_t, \mathbf{y}_t(1), \cdots, \mathbf{y}_t(j)\}_{t=1}^n$ is used as the input for training classifier $h_{j+1}$. Given a new testing instance $\mathbf{x}$, classifier $h_1$ in the chain is responsible for predicting the value of $\mathbf{y}(1)$ using input $\mathbf{x}$. Then, $h_2$ predicts the value of $\mathbf{y}(2)$ taking $\mathbf{x}$ plus the predicted value of $\mathbf{y}(1)$ as an input. Following in this way, $h_{j+1}$ predicts $\mathbf{y}(j+1)$ using the predicted value of $\mathbf{y}(1), \cdots, \mathbf{y}(j)$ as additional input information. CC passes label information between classifiers, allowing CC to exploit the label dependence and thus overcome the label independence problem of BR. Essentially, it builds a deterministic high-order Markov Chain model to capture the label correlations.

| Sample | Feature | Label Set |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | $\{\lambda_5\}$ |
| 2 | $\boldsymbol{x}_2$ | $\{\lambda_5\}$ |
| 3 | $\boldsymbol{x}_3$ | $\{\lambda_5\}$ |
| 4 | $\boldsymbol{x}_4$ | $\{\lambda_5\}$ |
| 5 | $\boldsymbol{x}_5$ | $\{\lambda_1\}$ |
| 6 | $\boldsymbol{x}_6$ | $\{\lambda_4\}$ |
| 7 | $\boldsymbol{x}_7$ | $\{\lambda_3\}$ |
| 8 | $\boldsymbol{x}_8$ | $\{\lambda_5\}$ |

Figure 2.4: The illustration of *select-max* transformation.

| Sample | Feature | Label Set |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | $\{\lambda_2\}$ |
| 2 | $\boldsymbol{x}_2$ | $\{\lambda_3\}$ |
| 3 | $\boldsymbol{x}_3$ | $\{\lambda_2\}$ |
| 4 | $\boldsymbol{x}_4$ | $\{\lambda_5\}$ |
| 5 | $\boldsymbol{x}_5$ | $\{\lambda_4\}$ |
| 6 | $\boldsymbol{x}_6$ | $\{\lambda_4\}$ |
| 7 | $\boldsymbol{x}_7$ | $\{\lambda_1\}$ |
| 8 | $\boldsymbol{x}_8$ | $\{\lambda_4\}$ |

Figure 2.5: The illustration of *select-min* transformation.

Different classifier chains involve different classifiers learned on different training sets and thus the order of the chain itself clearly affects the prediction performance. To solve the issue of selecting a chain order for CC, Read et al. [2009] proposed the extension of CC, called ensembled classifier chain (ECC), to average the multi-label predictions of CC over a set of random chain ordering. ECC first randomly reorders the labels $\{\lambda_1, \lambda_2, \cdots, \lambda_q\}$ many times. Then, CC is applied to the reordered labels for each time and the performance of CC is averaged over those times to obtain the final prediction performance.

However, CC suffers from three important problems: Does the label order affect the performance of CC? Apparently yes, because ECC is proposed to reduce the influence of the label order. Is there any globally optimal classifier chain which can achieve the optimal prediction performance for CC? If yes, how can the globally optimal classifier chain be found? This thesis will answer these problems.

| Sample | Feature | Label Set |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | $\{\lambda_2\}$ |
| 2 | $\boldsymbol{x}_2$ | $\{\lambda_5\}$ |
| 3 | $\boldsymbol{x}_3$ | $\{\lambda_3\}$ |
| 4 | $\boldsymbol{x}_4$ | $\{\lambda_5\}$ |
| 5 | $\boldsymbol{x}_5$ | $\{\lambda_1\}$ |
| 6 | $\boldsymbol{x}_6$ | $\{\lambda_4\}$ |
| 7 | $\boldsymbol{x}_7$ | $\{\lambda_1\}$ |
| 8 | $\boldsymbol{x}_8$ | $\{\lambda_3\}$ |

Figure 2.6: The illustration of *select-random* transformation.

| Sample | Feature | Label Set |
|---|---|---|
| 4 | $\boldsymbol{x}_4$ | $\{\lambda_5\}$ |
| 6 | $\boldsymbol{x}_6$ | $\{\lambda_4\}$ |

Figure 2.7: The illustration of *ignore* transformation.

## 2.4 Maximum Margin Output Coding

From the PT perspective, Hsu et al. [2009] use random transformation to project the original label space into a low dimensional label space. A regression model is trained on each transformed label. The compressed sensing technique is used to recover multi-labels from the regression output. To capture label dependency, canonical correlation analysis (CCA)and maximum margin output coding (MMOC) are respectively proposed by Zhang and Schneider [2011] and Zhang and Schneider [2012] to encode the label vectors. PLST Tai and Lin [2012] uses PCA to project the output to a lower dimension and applies the linear projection in the decoding procedure; its performance is inferior to CCA and MMOC in our experiments.

The work that is most relevant to this thesis is MMOC, which studies output coding for multi-label prediction and focuses on the following two important problems:

- **Discriminative:** The coding technique needs to be discriminative: encodings for different outputs should be significantly different from each other, such that the codeword for the correct output will not be confused with incorrect ones. This principle corresponds to the concept of code distance in coding theory and is related to good error-correcting capabilities Cover and Thomas [2006].

- **Predictable:** In output coding, codewords need to be predicted from the input other than through a channel, and it is critical that codewords are easy to predict. From the channel coding perspective, having predictable codewords corresponds

| Sample | Feature | Label $\lambda_1$ |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | 1 |
| 2 | $\boldsymbol{x}_2$ | 0 |
| 3 | $\boldsymbol{x}_3$ | 0 |
| 4 | $\boldsymbol{x}_4$ | 0 |
| 5 | $\boldsymbol{x}_5$ | 1 |
| 6 | $\boldsymbol{x}_6$ | 0 |
| 7 | $\boldsymbol{x}_7$ | 1 |
| 8 | $\boldsymbol{x}_8$ | 1 |

Figure 2.8: The first data set produced by the BR method.

| Sample | Feature | Label $\lambda_2$ |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | 1 |
| 2 | $\boldsymbol{x}_2$ | 0 |
| 3 | $\boldsymbol{x}_3$ | 1 |
| 4 | $\boldsymbol{x}_4$ | 0 |
| 5 | $\boldsymbol{x}_5$ | 0 |
| 6 | $\boldsymbol{x}_6$ | 0 |
| 7 | $\boldsymbol{x}_7$ | 0 |
| 8 | $\boldsymbol{x}_8$ | 0 |

Figure 2.9: The second data set produced by the BR method.

to reducing the channel error. In multi-label classification, finding predictable codewords provides an opportunity to exploit the dependency structure in the label space.

Based on structural SVMs McAllester [2006]; Tsochantaridis et al. [2005], Zhang and Schneider [2012] propose a maximum margin output coding method to make output codes that are both discriminative and predictable for multi-label classification. Specifically, MMOC embeds both the output and input in the same space by ensuring that the distance between the embedded input and the correct embedded output is less than the distance between the embedded input and any other embedded output with a large margin. This principle is naturally captured by maximizing the margin between the prediction distance to correct and incorrect encodings. Extensive empirical results demonstrate the superiority of MMOC.

Though MMOC has shown improved prediction performance, the training process involves an exponential number of constraints w.r.t. the number of labels, which is impractical for real-world applications, such as image annotation. MMOC is

| Sample | Feature | Label $\lambda_3$ |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | 0 |
| 2 | $\boldsymbol{x}_2$ | 1 |
| 3 | $\boldsymbol{x}_3$ | 1 |
| 4 | $\boldsymbol{x}_4$ | 0 |
| 5 | $\boldsymbol{x}_5$ | 0 |
| 6 | $\boldsymbol{x}_6$ | 0 |
| 7 | $\boldsymbol{x}_7$ | 1 |
| 8 | $\boldsymbol{x}_8$ | 1 |

Figure 2.10: The third data set produced by the BR method.

| Sample | Feature | Label $\lambda_4$ |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | 0 |
| 2 | $\boldsymbol{x}_2$ | 0 |
| 3 | $\boldsymbol{x}_3$ | 0 |
| 4 | $\boldsymbol{x}_4$ | 0 |
| 5 | $\boldsymbol{x}_5$ | 1 |
| 6 | $\boldsymbol{x}_6$ | 1 |
| 7 | $\boldsymbol{x}_7$ | 0 |
| 8 | $\boldsymbol{x}_8$ | 1 |

Figure 2.11: The fourth data set produced by the BR method.

proposed based on structural SVMs McAllester [2006]; Tsochantaridis et al. [2005], and McAllester [2006] has already proved that structural SVMs fail to converge on the optimal decoder even with infinite training data. Furthermore, MMOC apply mean-field approximation Zhang and Schneider [2011] in the decoding step, its prediction process is still expensive when there are many labels. Thus, it is non-trivial to break the bottlenecks of MMOC, and develop efficient and consistent algorithms for solving multi-label learning tasks.

## 2.5 Prediction Complexity Categories

Much effort has been focussed on prediction tasks, like image prediction Boutell et al. [2004] and video prediction Song et al. [2005]. Usually, these prediction tasks can be formulated as a multi-label prediction problem Tsoumakas et al. [2010]. According to the prediction complexity, we divide those methods into several categories:

The first category includes some popular methods, such as Binary Relevance (BR)

| Sample | Feature | Label $\lambda_5$ |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | 1 |
| 2 | $\boldsymbol{x}_2$ | 1 |
| 3 | $\boldsymbol{x}_3$ | 1 |
| 4 | $\boldsymbol{x}_4$ | 1 |
| 5 | $\boldsymbol{x}_5$ | 0 |
| 6 | $\boldsymbol{x}_6$ | 0 |
| 7 | $\boldsymbol{x}_7$ | 0 |
| 8 | $\boldsymbol{x}_8$ | 1 |

Figure 2.12: The fifth data set produced by the BR method.

| Sample | Feature | Class |
|---|---|---|
| 1 | $\boldsymbol{x}_1$ | $\lambda_{1,2,5}$ |
| 2 | $\boldsymbol{x}_2$ | $\lambda_{3,5}$ |
| 3 | $\boldsymbol{x}_3$ | $\lambda_{2,3,5}$ |
| 4 | $\boldsymbol{x}_4$ | $\lambda_5$ |
| 5 | $\boldsymbol{x}_5$ | $\lambda_{1,4}$ |
| 6 | $\boldsymbol{x}_6$ | $\lambda_4$ |
| 7 | $\boldsymbol{x}_7$ | $\lambda_{1,3}$ |
| 8 | $\boldsymbol{x}_8$ | $\lambda_{1,3,4,5}$ |

Figure 2.13: The transformed data set using the label powerset method.

Tsoumakas et al. [2010] and Classifier Chain (CC) Read et al. [2009], the number of predictions scale linearly with the length of label vector. Duan et al. [2015] have proposed a novel probabilistic cascaded method for mapping label sets in a source taxonomy to label sets in a target taxonomy. All these methods would make thousands of predictions when there are thousands of labels, so they cannot handle many labels.

The second category includes the encoding-decoding strategy. For example, Zhang and Schneider [2011, 2012] first use different projection methods to transform the original label space into another space, and recover the original multiple labels using an expensive decoding process, which involves solving a quadratic programming (QP) problem on a space with a combinatorial nature.

The third category includes Label Powerset (LP) Tsoumakas et al. [2010] and its variant Tsoumakas and Vlahavas [2007b]. LP reduces a multi-label prediction problem into a multi-class prediction problem by treating each distinct label set as one of the classes for a transformed multi-class learning task. One can then train a single multi-class classifier (Tsoumakas et al. [2010]) or many binary classifiers (e.g. one-

| Class | Probability | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $\lambda_4$ | $\lambda_5$ |
|---|---|---|---|---|---|---|
| $\lambda_{1,2,5}$ | 0.1 | 1 | 0 | 0 | 0 | 1 |
| $\lambda_{3,5}$ | 0.0 | 0 | 0 | 1 | 0 | 1 |
| $\lambda_{2,3,5}$ | 0.3 | 0 | 1 | 1 | 0 | 1 |
| $\lambda_5$ | 0.1 | 0 | 0 | 0 | 0 | 1 |
| $\lambda_{1,4}$ | 0.2 | 1 | 0 | 0 | 1 | 0 |
| $\lambda_4$ | 0.0 | 0 | 0 | 0 | 1 | 0 |
| $\lambda_{1,3}$ | 0.1 | 1 | 0 | 1 | 0 | 0 |
| $\lambda_{1,3,4,5}$ | 0.2 | 1 | 0 | 1 | 1 | 1 |
| Sum of the probability | | 0.6 | 0.3 | 0.6 | 0.4 | 0.7 |

Figure 2.14: An example of obtaining a ranking among labels using label powerset method with probability outputs.

vs-all or one-vs-one) for the transformed multi-class prediction problem. The number of transformed classes is upper bounded by $\min(m, 2^q)$ where $m$ means the size of training data set and $q$ means the number of labels. For large values of $m$ and $q$, it imposes an extremely high training complexity.

The last category is tree-based algorithms, which have prediction costs that are logarithmic in the number of labels. The multi-label random forest (MLRF) algorithm Agrawal et al. [2013] first learns random trees in a feature space to accelerate prediction efficiency based on a distributed system over 1,000 computer nodes. FastXML Prabhu and Varma [2014] is the most recently advanced technique. FastXML outperforms MLRF, but our extensive experiment results verify that it generally underperforms in terms of multi-label prediction performance. Homer Tsoumakas et al. [2008] is developed to use a divide-and-conquer-strategy to divide original problem into $k$ sub-problems. Some literature Madjarov et al. [2012]; Tsoumakas et al. [2008] has shown Homer achieves superior prediction performance. Unfortunately, we cannot get the results of Homer on medium-sized data sets within one week in our experiment. Thus, it is imperative to develop an efficient, yet accurate multi-label prediction algorithm with the minimum number of predictions. This thesis will employ the techniques from Huffman coding and Shannon-Fano Coding to address this issue. Next, we review these coding strategies.

### 2.5.1 Huffman Coding

Huffman coding [Huffman, 1952] is one of the most widespread bottom-up encoding algorithm for data compression. Huffman coding uses a specific method for choosing the representation for each symbol, resulting in a prefix code, that is, the bit string representing some particular symbol is never a prefix of the bit string representing any other symbol. Given a set of symbols and their weights (usually proportional to probabilities), Huffman coding aims to find a set of prefix codewords with minimum

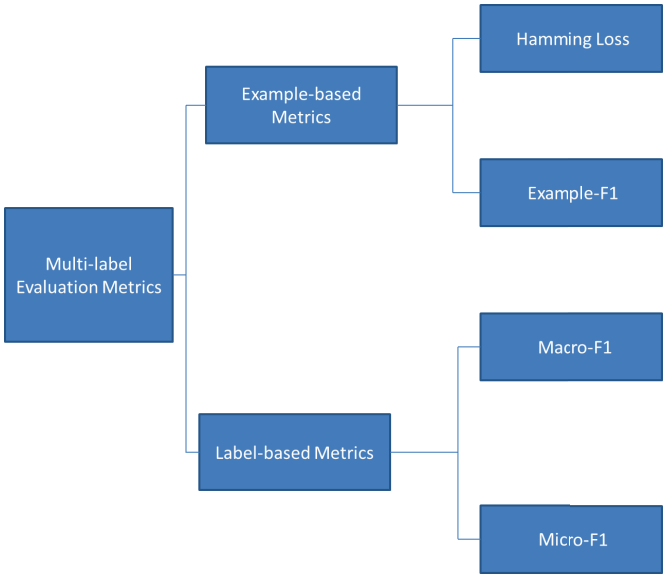


Figure 2.15: Categorization of representative evaluation metric used in this thesis.

expected codeword length. Based on the frequency of occurrence of each symbol, the principle of Huffman coding is to use a lower number of bits to encode the symbol that occurs more frequently. The detailed procedure of Huffman coding can be referred to Huffman [1952]. This chapter first uses the idea of Huffman coding to deal with the label powerset prediction problems.

### 2.5.2 Shannon-Fano Coding

Shannon-Fano coding [Shannon, 1948] is a top-down encoding technique for constructing a prefix code based on a set of symbols and their weights. Shannon-Fano coding arranges the symbols in order from biggest weight to smallest weight, and then divides them into two sets whose total weights are roughly comparable. Then, we encode symbols in the first set as zero and symbols in the second set as one. As long as any sets with more than one member remain, the same process is repeated on these sets. The detailed procedure of Shannon-Fano coding can be referred to Shannon [1948]. Analogous to the Shannon-Fano coding strategy, this chapter first proposes a novel coding tree algorithm to deal with multi-label prediction problems.

## 2.6 Evaluation Metric

In traditional supervised learning, the prediction performance of the learning models is usually evaluated by conventional metrics such as error rate, F-measure and area

under the ROC curve (AUC). Because each instance can belong to multiple labels simultaneously, the performance evaluation in multi-label learning settings is much more complicated than traditional supervised learning. Therefore, a number of evaluation metrics are developed for multi-label learning. These evaluation metrics can be generally categorized into two groups: example-based metrics Ghamrawi and McCallum [2005]; Schapire and Singer [2000a] and label-based metrics Tsoumakas and Vlahavas [2007a].

In this thesis, we consider the following evaluation measurements Mao et al. [2013]; Zhang and Zhou [2014] to measure the prediction performance of all methods fairly:

- Hamming Loss: computes the average zero-one score for all the labels and instances.
- Example-F1: computes the F-1 score for all the labels of each testing sample and then takes the average of the F-1 score.
- Macro-F1: calculates the F-1 score for each label and then takes the average of the F-1 score.
- Micro-F1: computes true positives, true negatives, false positives and false negatives over all labels, and then calculates an overall F-1 score.

The smaller the value of Hamming Loss, the better the performance, while the larger the value of the other three measurements, the better the performance. Figure 2.15 summarize the categorization of representative evaluation metric used in this thesis.

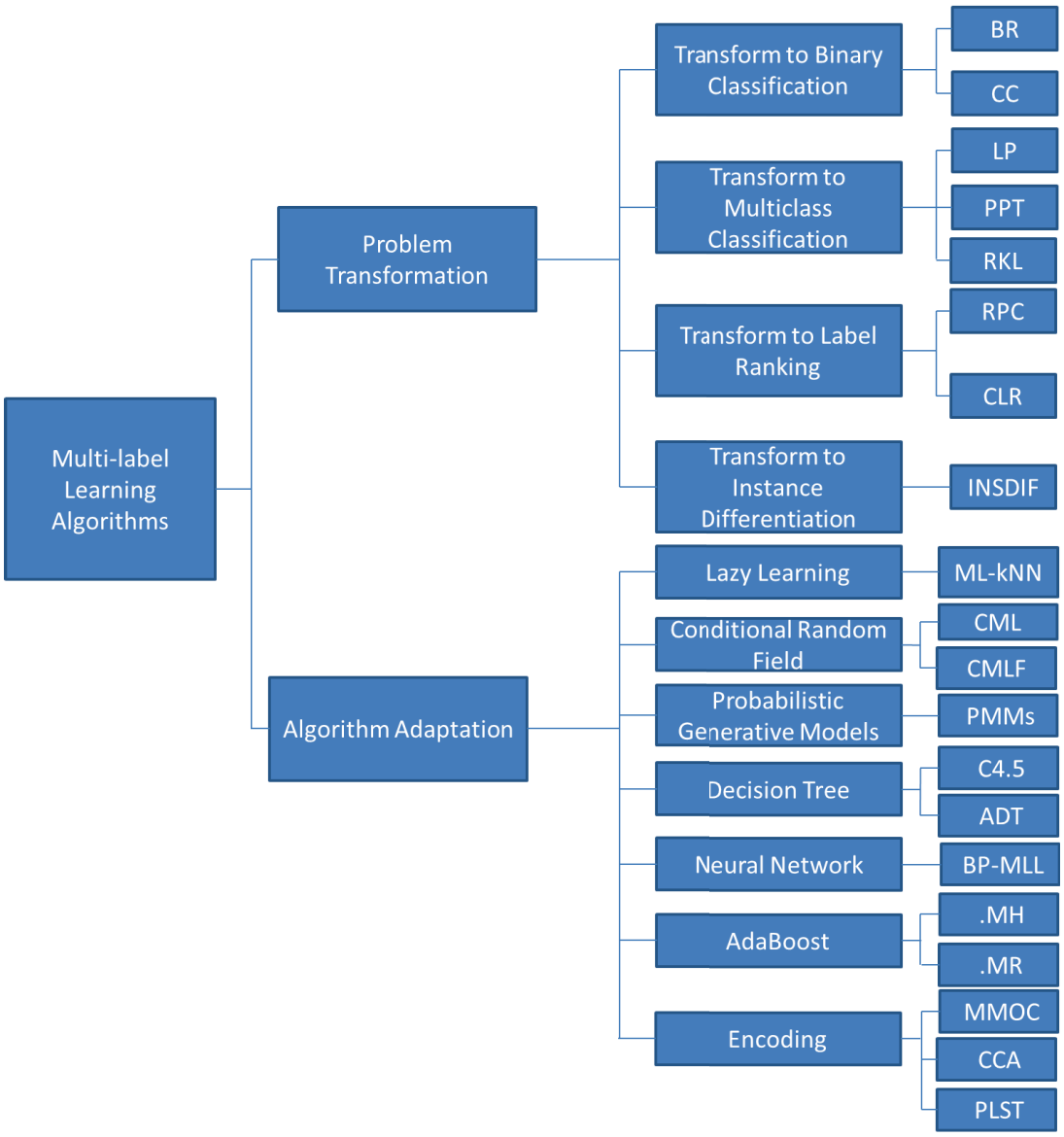


Figure 2.16: Categorization of representative algorithm adaptation and problem transformation algorithms reviewed in this Chapter.

# Chapter 3

# On the Optimality of Classifier Chain for Multi-label Classification

## 3.1 Motivations

One popular strategy for multi-label classification is to reduce the original problem into many binary classification problems. Many works have followed this strategy. For example, binary relevance (BR) Tsoumakas et al. [2010] is a simple approach for multi-label learning which independently trains a binary classifier for each label. Recently, Dembczynski et al. [2010] have shown that methods of multi-label learning which explicitly capture label dependency will usually achieve better prediction performance. Therefore, modeling the label dependency is one of the major challenges in multi-label classification problems. Many multi-label learning models Dembczynski et al. [2010]; Guo and Gu [2011]; Huang and Zhou [2012]; Kang et al. [2006]; Liu and Tsang [2015a]; Read et al. [2009]; Tan et al. [2015]; Zhang and Schneider [2012] have been developed to capture label dependency. Amongst them, the *classifier chain* (CC) model is one of the most popular methods due to its simplicity and promising experimental results Read et al. [2009].

CC works as follows: One classifier is trained for each label. For the $(i+1)$th label, each instance is augmented with the 1st, 2nd, $\cdots$, $i$th label as the input to train the $(i+1)$th classifier. Given a new instance to be classified, CC firstly predicts the value of the first label, then takes this instance together with the predicted value as the input to predict the value of the next label. CC proceeds in this way until the last label is predicted. However, here is the question: *Does the label order affect the performance of CC?* Apparently yes, because different classifier chains involve different classifiers trained on different training sets. Thus, to reduce the influence of the label order, Read et al. [2009] proposed the *ensembled classifier chain* (ECC) to average the multi-label predictions of CC over a set of random chain ordering. Since the performance of CC is

sensitive to the choice of label order, there is another important question: *Is there any globally optimal classifier chain which can achieve the optimal prediction performance for CC?* If yes, *how can the globally optimal classifier chain be found?*

To answer the last two questions, we first generalize the CC model over a random label order. We then present a theoretical analysis of the generalization error for the proposed generalized model. Our results show that the upper bound of the generalization error depends on the sum of reciprocal of square of the margin over the labels. Thus, we can answer the second question: the globally optimal CC exists only when the minimization of the upper bound is achieved over this CC. To find the globally optimal CC, we can search over $q!$ different label orders[1], where $q$ denotes the number of labels, which is computationally infeasible for a large $q$. In this chapter, we propose the *dynamic programming based classifier chain* (CC-DP) algorithm to simplify the search algorithm, which requires $\mathcal{O}(q^3nd)$ time complexity. To speed up the training process, a *greedy classifier chain* (CC-Greedy) algorithm is proposed to find a locally optimal CC, where the time complexity of the CC-Greedy algorithm is $\mathcal{O}(q^2nd)$. Furthermore, we propose Tree-DP and Tree-Greedy algorithms to further speed up CC-DP and CC-Greedy, respectively, which scale linearly with $q$. The main contributions of this work are:

1. We generalize the classifier chain model over a random label order.

2. A theoretical analysis of the generalization error is provided for the generalized model. Our results show that the upper bound of the generalization error is dependent on the sum of the reciprocal of square of margin over the labels.

3. Based on our results, the CC-DP algorithm is proposed to find the globally optimal classifier chain which will achieve the globally optimal prediction performance for CC. We propose a CC-Greedy algorithm to speed up the training process by finding a local optimal classifier chain. Furthermore, we propose Tree-DP and Tree-Greedy algorithms to further speed up CC-DP and CC-Greedy, respectively, which scale linearly with $q$.

4. Extensive experiments on a number of real-world multi-label data sets from various domains demonstrate that our proposed CC-DP algorithm outperforms BR, CC and ECC, and the CC-Greedy algorithm achieves comparable prediction performance with CC-DP.

[1]! represents the factorial notation.

## 3.2 Proposed Model and Generalization Error Analysis

### 3.2.1 Generalized Classifier Chain

Assume $\mathbf{x}_t \in \mathbb{R}^d$ is a real vector representing an input or instance (feature) for $t \in \{1, \cdots, n\}$. $n$ denotes the number of training samples. $\mathcal{Y}_t \subseteq \{\lambda_1, \lambda_2, \cdots, \lambda_q\}$ is the corresponding output (label). $\mathbf{y}_t \in \{0, 1\}^q$ is used to represent the label set $\mathcal{Y}_t$, where $\mathbf{y}_t(j) = 1$ if and only if $\lambda_j \in \mathcal{Y}_t$.

We generalize the CC model over a random label order, called *generalized classifier chain* (GCC) model. Assume the labels $\{\lambda_1, \lambda_2, \cdots, \lambda_q\}$ are randomly reordered as $\{\zeta_1, \zeta_2, \cdots, \zeta_q\}$, where $\zeta_j = \lambda_k$ means label $\lambda_k$ moves to position $j$ from $k$. In the GCC model, classifiers are also linked along a chain where each classifier $h_j$ deals with the binary classification problem for label $\zeta_j$ ($\lambda_k$). GCC follows the same training and testing procedures as CC, while the only difference is the label order. In the GCC model, for input $\mathbf{x}_t$, $\mathbf{y}_t(j) = 1$ if and only if $\zeta_j \in \mathcal{Y}_t$.

### 3.2.2 Generalization Error Analysis

In this section, we analyze the generalization error bound of the multi-label classification problem using GCC based on the techniques developed for the generalization performance of classifiers with a large margin Shawe-Taylor et al. [1998] and perceptron decision tree Bennett et al. [2000].

Let $\mathbf{X}$ represent the input space. Both $\mathbf{s}$ and $\bar{\mathbf{s}}$ are $m$ samples drawn independently according to an unknown distribution $D$. We denote logarithms to base 2 by $\log$. If $\mathcal{S}$ is a set, $|\mathcal{S}|$ denotes its cardinality. $\|\cdot\|$ means the $l_2$ norm. We train a support vector machine(SVM) for each label $\zeta_j$. Let $\{\mathbf{x}_t\}_{t=1}^n$ as the feature and $\{\mathbf{y}_t(\zeta_j)\}_{t=1}^n$ as the label, the output parameter of SVM is defined as $[\mathbf{w}_j, b_j] = SVM(\{\mathbf{x}_t, \mathbf{y}_t(\zeta_1), \cdots, \mathbf{y}_t(\zeta_{j-1})\}_{t=1}^n, \{\mathbf{y}_t(\zeta_j)\}_{t=1}^n)$. The margin for label $\zeta_j$ is defined as:

$$\gamma^j = \frac{1}{||\mathbf{w}_j||^2} \tag{3.1}$$

We begin with the definition of the fat shattering dimension.

**Definition 1** (Kearns and Schapire [1990])**.** *Let $\mathcal{H}$ be a set of real valued functions. We say that a set of points $P$ is $\gamma$-shattered by $\mathcal{H}$ relative to $r = (r_p)_{p \in P}$ if there are real numbers $r_p$ indexed by $p \in P$ such that for all binary vectors $b$ indexed by $P$, there is a function $f_b \in \mathcal{H}$ satisfying*

$$f_b(p) = \begin{cases} \geq r_p + \gamma & \text{if } b_p = 1 \\ \leq r_p - \gamma & \text{otherwise} \end{cases}$$

*The fat shattering dimension $fat(\gamma)$ of the set $\mathcal{H}$ is a function from the positive real numbers to the integers which maps a value $\gamma$ to the size of the largest $\gamma$-shattered set, if this is finite, or infinity otherwise.*

Assume $\mathcal{H}$ is the real valued function class and $h \in \mathcal{H}$. $l(y, h(x))$ denotes the loss function. The expected error of $h$ is defined as $er_D[h] = E_{(x,y)\sim D}[l(y, h(x))]$, where $(x, y)$ drawn from the unknown distribution $D$. Here we select 0-1 loss function. So, $er_D[h] = P_{(x,y)\sim D}(h(x) \neq y)$. $er_{\mathbf{s}}[h]$ is defined as $er_{\mathbf{s}}[h] = \frac{1}{n}\sum_{t=1}^{n}[\![y_t \neq h(x_t)]\!]$.[1]

Suppose $\mathcal{N}(\epsilon, \mathcal{H}, \mathbf{s})$ is the $\epsilon$-covering number of $\mathcal{H}$ with respect to the $l_\infty$ pseudo-metric measuring the maximum discrepancy on the sample $\mathbf{s}$. The notion of the covering number can be referred to Appendix A.1. We introduce the following general corollary regarding the bound of the covering number:

**Corollary 1** (Shawe-Taylor et al. [1998])**.** *Let $\mathcal{H}$ be a class of functions $X \to [a, b]$ and $D$ a distribution over $X$. Choose $0 < \epsilon < 1$ and let $d = fat(\epsilon/4) \leq em$. Then*

$$E(\mathcal{N}(\epsilon, \mathcal{H}, \boldsymbol{s})) \leq 2\Big(\frac{4m(b-a)^2}{\epsilon^2}\Big)^{d\log(2em(b-a)/(d\epsilon))} \tag{3.2}$$

*where the expectation $E$ is over samples $\boldsymbol{s} \in X^m$ drawn according to $D^m$.*

We study the generalization error bound of the specified GCC with the specified number of labels and margins. Let $G$ be the set of classifiers of GCC, $G = \{h_1, h_2, \cdots, h_q\}$. $er_{\mathbf{s}}[G]$ denotes the fraction of the number of errors that GCC makes on $\mathbf{s}$. Define $\hat{\mathbf{x}} \in \mathbf{X} \times \{0, 1\}$, $\hat{h}_j(\hat{\mathbf{x}}) = h_j(\mathbf{x})(1-\mathbf{y}(j)) - h_j(\mathbf{x})\mathbf{y}(j)$. If an instance $\mathbf{x} \in \mathbf{X}$ is correctly classified by $h_j$, then $\hat{h}_j(\hat{\mathbf{x}}) < 0$. Moreover, we introduce the following proposition:

**Proposition 1.** *If an instance $\boldsymbol{x} \in \boldsymbol{X}$ is misclassified by a GCC model, then $\exists h_j \in G, \hat{h}_j(\hat{\mathbf{x}}) \geq 0$.*

**Lemma 1.** *Given a specified GCC model with $q$ labels and with margins $\gamma^1, \gamma^2, \cdots, \gamma^q$ for each label satisfying $k_i = fat(\gamma^i/8)$, where $fat$ is continuous from the right. If GCC has correctly classified $m$ multi-labeled examples $\boldsymbol{s}$ generated independently according to the unknown (but fixed) distribution $D$ and $\bar{\boldsymbol{s}}$ is a set of another $m$ multi-labeled examples, then we can bound the following probability to be less than $\delta$: $P^{2m}\{\boldsymbol{s}\bar{\boldsymbol{s}} : \exists$ a GCC model, it correctly classifies $\boldsymbol{s}$, fraction of $\bar{\boldsymbol{s}}$ misclassified $> \epsilon(m, q, \delta)\} < \delta$, where $\epsilon(m, q, \delta) = \frac{1}{m}(Q\log(32m) + \log\frac{2^q}{\delta})$ and $Q = \sum_{i=1}^{q} k_i \log(\frac{8em}{k_i})$.*

[1] The expression $[\![y_t \neq h(x_t)]\!]$ evaluates to 1 if $y_t \neq h(x_t)$ is true and to 0 otherwise.

*Proof.* (of Lemma 1). Suppose $G$ is a GCC model with $q$ labels and with margins $\gamma^1, \gamma^2, \cdots, \gamma^q$, the probability event in Lemma 1 can be described as

$$A = \{\mathbf{s}\bar{\mathbf{s}} : \exists G, k_i = fat(\gamma^i/8), er_{\mathbf{s}}[G] = 0, er_{\bar{\mathbf{s}}}[G] > \epsilon\}.$$

Let $\hat{\mathbf{s}}$ and $\hat{\bar{\mathbf{s}}}$ denote two different set of $m$ examples, which are drawn i.i.d. from the distribution $D \times \{0, 1\}$. Applying the definition of $\hat{\mathbf{x}}$, $\hat{h}$ and Proposition 1, the event can also be written as $A = \{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}} : \exists G, \hat{\gamma}^i = \gamma^i/2, k_i = fat(\hat{\gamma}^i/4), er_{\mathbf{s}}[G] = 0, r_i = max_t \hat{h}_i(\hat{\mathbf{x}}_t), 2\hat{\gamma}^i = -r_i, |\{\hat{\mathbf{y}} \in \hat{\bar{\mathbf{s}}} : \exists h_i \in G, \hat{h}_i(\hat{\mathbf{y}}) \geq 2\hat{\gamma}^i + r_i\}| > m\epsilon\}$. Here, $-max_t \hat{h}_i(\hat{\mathbf{x}}_t)$ means the minimal value of $|h_i(\mathbf{x})|$ which represents the margin for label $\zeta_i$, so $2\hat{\gamma}^i = -r_i$. Let $\gamma_{k_i} = min\{\gamma' : fat(\gamma'/4) \leq k_i\}$, so $\gamma_{k_i} \leq \hat{\gamma}^i$, we define the following function:

$$\pi(\hat{h}) = \begin{cases} 0 & \text{if } \hat{h} \geq 0 \\ -2\gamma_{k_i} & \text{if } \hat{h} \leq -2\gamma_{k_i} \\ \hat{h} & \text{otherwise} \end{cases}$$

so $\pi(\hat{h}) \in [-2\gamma_{k_i}, 0]$. Let $\pi(\hat{G}) = \{\pi(\hat{h}) : h \in G\}$.

Let $B^{k_i}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}$ represent the minimal $\gamma_{k_i}$-cover set of $\pi(\hat{G})$ in the pseudo-metric $d_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}$. We have that for any $h_i \in G$, there exists $\tilde{f} \in B^{k_i}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}$, $|\pi(\hat{h}_i(\hat{\mathbf{z}})) - \pi(\tilde{f}(\hat{\mathbf{z}}))| < \gamma_{k_i}$, for all $\hat{\mathbf{z}} \in \hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}$. For all $\hat{\mathbf{x}} \in \hat{\mathbf{s}}$, by the definition of $r_i$, $\hat{h}_i(\hat{\mathbf{x}}) \leq r_i = -2\hat{\gamma}^i$, and $\gamma_{k_i} \leq \hat{\gamma}^i$, $\hat{h}_i(\hat{\mathbf{x}}) \leq -2\gamma_{k_i}$, $\pi(\hat{h}_i(\hat{\mathbf{x}})) = -2\gamma_{k_i}$, so $\pi(\tilde{f}(\hat{\mathbf{x}})) < -2\gamma_{k_i} + \gamma_{k_i} = -\gamma_{k_i}$. However, there are at least $m\epsilon$ points $\hat{\mathbf{y}} \in \hat{\bar{\mathbf{s}}}$ such that $\hat{h}_i(\hat{\mathbf{y}}) \geq 0$, so $\pi(\tilde{f}(\hat{\mathbf{y}})) > -\gamma_{k_i} > max_t \pi(\tilde{f}(\hat{\mathbf{x}}_t))$. Since $\pi$ only reduces separation between output values, we conclude that the inequality $\tilde{f}(\hat{\mathbf{y}}) > max_t \tilde{f}(\hat{\mathbf{x}}_t)$ holds. Moreover, the $m\epsilon$ points in $\hat{\bar{\mathbf{s}}}$ with the largest $\tilde{f}$ values must remain for the inequality to hold. By the permutation argument, at most $2^{-m\epsilon}$ of the sequences obtained by swapping corresponding points satisfy the conditions for fixed $\tilde{f}$.

As for any $h_i \in G$, there exists $\tilde{f} \in B^{k_i}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}$, so there are $|B^{k_i}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}|$ possibilities of $\tilde{f}$ that satisfy the inequality for $k_i$. Note that $|B^{k_i}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}|$ is a positive integer which is usually bigger than 1 and by the union bound, we get the following inequality:

$$P(A) \leq (E(|B^{k_1}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}|) + \cdots + E(|B^{k_q}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}|))2^{-m\epsilon} \leq (E(|B^{k_1}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}|) \times \cdots \times E(|B^{k_q}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}|))2^{-m\epsilon}$$

Since every set of points $\gamma$-shattered by $\pi(\hat{G})$ can be $\gamma$-shattered by $\hat{G}$, so $fat_{\pi(\hat{G})}(\gamma) \leq fat_{\hat{G}}(\gamma)$, where $\hat{G} = \{\hat{h} : h \in G\}$. Hence, by Corollary 1 (setting $[a, b]$ to $[-2\gamma_{k_i}, 0]$, $\epsilon$ to $\gamma_{k_i}$ and $m$ to $2m$),

$$E(|B^{k_i}_{\hat{\mathbf{s}}\hat{\bar{\mathbf{s}}}}|) = E(\mathcal{N}(\gamma_{k_i}, \pi(\hat{G}), \hat{\mathbf{s}}\hat{\bar{\mathbf{s}}})) \leq 2(32m)^{d\log(\frac{8em}{d})}$$

where $d = fat_{\pi(\hat{G})}(\gamma_{k_i}/4) \leq fat_{\hat{G}}(\gamma_{k_i}/4) \leq k_i$. Thus $E(|B^{k_i}_{\hat{\mathbf{s}}\hat{\mathbf{s}}}|) \leq 2(32m)^{k_i \log(\frac{8em}{k_i})}$, and we obtain

$$P(A) \leq (E(|B^{k_1}_{\hat{\mathbf{s}}\hat{\mathbf{s}}}|) \times \cdots \times E(|B^{k_q}_{\hat{\mathbf{s}}\hat{\mathbf{s}}}|))2^{-m\epsilon} \leq \prod_{i=1}^{q} 2(32m)^{k_i \log(\frac{8em}{k_i})} = 2^q(32m)^Q$$

where $Q = \sum_{i=1}^{q} k_i \log(\frac{8em}{k_i})$. And so $(E(|B^{k_1}_{\hat{\mathbf{s}}\hat{\mathbf{s}}}|) \times \cdots \times E(|B^{k_q}_{\hat{\mathbf{s}}\hat{\mathbf{s}}}|))2^{-m\epsilon} < \delta$ provided

$$\epsilon(m, q, \delta) \geq \frac{1}{m}\Big(Q \log(32m) + \log \frac{2^q}{\delta}\Big)$$

as required. □

Lemma 1 applies to a particular GCC model with a specified number of labels and a specified margin for each label. In practice, we will observe the margins after running the GCC model. Thus, we must bound the probabilities uniformly over all of the possible margins that can arise to obtain a practical bound. The generalization error bound of the multi-label classification problem using GCC is shown as follows:

**Theorem 1.** *Suppose a random $m$ multi-labeled sample can be correctly classified using a GCC model, and suppose this GCC model contains $q$ classifiers with margins $\gamma^1, \gamma^2, \cdots, \gamma^q$ for each label. Then we can bound the generalization error with probability greater than $1 - \delta$ to be less than*

$$\frac{130R^2}{m}\Big(Q' \log(8em) \log(32m) + \log \frac{2(2m)^q}{\delta}\Big)$$

*where $Q' = \sum_{i=1}^{q} \frac{1}{(\gamma^i)^2}$ and $R$ is the radius of a ball containing the support of the distribution.*

Before proving Theorem 1, we state one key Symmetrization lemma and Theorem 2.

**Lemma 2** (Symmetrization)**.** *Let $\mathcal{H}$ be the real valued function class. $\boldsymbol{s}$ and $\bar{\boldsymbol{s}}$ are $m$ samples both drawn independently according to the unknown distribution $D$. If $m\epsilon^2 \geq 2$, then*

$$P_{\boldsymbol{s}}(\sup_{h \in \mathcal{H}} |er_D[h] - er_{\boldsymbol{s}}[h]| \geq \epsilon) \ \leq \ 2P_{\boldsymbol{s}\bar{\boldsymbol{s}}}(\sup_{h \in \mathcal{H}} |er_{\bar{\boldsymbol{s}}}[h] - er_{\boldsymbol{s}}[h]| \geq \epsilon/2) \tag{3.3}$$

The proof details of this lemma can be found in Appendix A.2.

**Theorem 2** (Bartlett and Shawe-Taylor [1998])**.** *Let $\mathcal{H}$ be restricted to points in a ball of $\mathscr{M}$ dimensions of radius $R$ about the origin, then*

$$fat_{\mathcal{H}}(\gamma) \leq \min\Big\{\frac{R^2}{\gamma^2}, \mathscr{M} + 1\Big\} \tag{3.4}$$

*Proof.* (of Theorem 1). We must bound the probabilities over different margins. We first use Lemma 2 to bound the probability of error in terms of the probability of the discrepancy between the performance on two halves of a double sample. Then we combine this result with Lemma 1. We must consider all possible patterns of $k_i$'s for label $\zeta_i$. The largest value of $k_i$ is $m$. Thus, for fixed $q$, we can bound the number of possibilities by $m^q$. Hence, there are $m^q$ of applications of Lemma 1.

Let $c_i = \{\gamma^1, \gamma^2, \cdots, \gamma^q\}$ denote the $i$-th combination of margins varied in $\{1, \cdots, m\}^q$. $\mathcal{G}$ denotes a set of GCC models. The generalization error of $G$ can be represented as $er_D[G]$ and $er_{\mathbf{s}}[G]$ is 0, where $G \in \mathcal{G}$. The uniform convergence bound of the generalization error is

$$P_{\mathbf{s}}(\sup_{G\in\mathcal{G}} |er_D[G] - er_{\mathbf{s}}[G]| \geq \epsilon)$$

Applying Lemma 2,

$$P_{\mathbf{s}}(\sup_{G\in\mathcal{G}} |er_D[G] - er_{\mathbf{s}}[G]| \geq \epsilon) \leq 2P_{\mathbf{s}\bar{\mathbf{s}}}(\sup_{G\in\mathcal{G}} |er_{\bar{\mathbf{s}}}[G] - er_{\mathbf{s}}[G]| \geq \epsilon/2)$$

Let $J_{c_i} = \{\mathbf{s}\bar{\mathbf{s}} : \exists$ a GCC model $G$ with $q$ labels and with margins $c_i : k_i = fat(\gamma^i/8), er_{\mathbf{s}}[G] = 0, er_{\bar{\mathbf{s}}}[G] \geq \epsilon/2\}$. Clearly,

$$P_{\mathbf{s}\bar{\mathbf{s}}}(\sup_{G\in\mathcal{G}} |er_{\bar{\mathbf{s}}}[G] - er_{\mathbf{s}}[G]| \geq \epsilon/2) \leq P^{m^q}\Big(\bigcup_{i=1}^{m^q} J_{c_i}\Big)$$

As $k_i$ still satisfies $k_i = fat(\gamma^i/8)$, Lemma 1 can still be applied to each case of $P^{m^q}(J_{c_i})$. Let $\delta_k = \delta/m^q$. Applying Lemma 1 (replacing $\delta$ by $\delta_k/2$), we get:

$$P^{m^q}(J_{c_i}) < \delta_k/2$$

where $\epsilon(m, k, \delta_k/2) \geq 2/m(Q\log(32m) + \log\frac{2\times 2^q}{\delta_k})$ and $Q = \sum_{i=1}^{q} k_i \log(\frac{4em}{k_i})$. By the union bound, it suffices to show that $P^{m^q}(\bigcup_{i=1}^{m^q} J_{c_i}) \leq \sum_{i=1}^{m^q} P^{m^q}(J_{c_i}) < \delta_k/2 \times m^q = \delta/2$. Applying Lemma 2,

$$\begin{aligned} P_{\mathbf{s}}(\sup_{G\in\mathcal{G}} |er_D[G] - er_{\mathbf{s}}[G]| \geq \epsilon) &\leq 2P_{\mathbf{s}\bar{\mathbf{s}}}(\sup_{G\in\mathcal{G}} |er_{\bar{\mathbf{s}}}[G] - er_{\mathbf{s}}[G]| \geq \epsilon/2) \\ &\leq 2P^{m^q}\Big(\bigcup_{i=1}^{m^q} J_{c_i}\Big) < \delta \end{aligned}$$

Thus, $P_{\mathbf{s}}(\sup_{G\in\mathcal{G}} |er_D[G] - er_{\mathbf{s}}[G]| \leq \epsilon) \geq 1 - \delta$. Let $R$ be the radius of a ball containing the support of the distribution. Applying Theorem 2, we get $k_i = fat(\gamma^i/8) \leq 65R^2/(\gamma^i)^2$. Note that we have replaced the constant $8^2 = 64$ by 65 in

order to ensure the continuity from the right required for the application of Lemma 1. We have upperbounded $\log(8em/k_i)$ by $\log(8em)$. Thus,

$$\begin{aligned} er_D[G] &\leq 2/m\Big(Q\log(32m) + \log\frac{2(2m)^q}{\delta}\Big) \\ &\leq \frac{130R^2}{m}\Big(Q'\log(8em)\log(32m) + \log\frac{2(2m)^q}{\delta}\Big) \end{aligned}$$

where $Q' = \sum_{i=1}^{q} \frac{1}{(\gamma^i)^2}$. □

Given the training data size and the number of labels, Theorem 1 reveals one important factor in reducing the generalization error bound for the GCC model: the minimization of the sum of reciprocal of square of the margin over the labels. Thus, we obtain the following Corollary:

**Corollary 2** (Globally Optimal Classifier Chain)**.** *Suppose a random $m$ multi-labeled sample with $q$ labels can be correctly classified using a GCC model, this GCC model is the globally optimal classifier chain if and only if the minimization of $Q'$ in Theorem 1 is achieved over this classifier chain.*

**Remark.** By exploiting the relationship between labels, our proposed GCC model is able to achieve lower generalization error bound.

Given the number of labels $q$, there are $q!$ different label orders. It is very expensive to find the globally optimal CC, which can minimize $Q'$, by searching over all of the label orders. Next, we discuss two simple algorithms.

## 3.3 Optimal Classifier Chain Algorithm

In this section, we propose two simple algorithms for finding the optimal CC based on our result in Section 3.2. To clearly state the algorithms, we redefine the margins with label order information. Given label set $\mathcal{M} = \{\lambda_1, \lambda_2, \cdots, \lambda_q\}$, suppose a GCC model contains $q$ classifiers. Let $o_i (1 \leq o_i \leq q)$ denote the order of $\lambda_i$ in the GCC model, $\gamma_i^{o_i}$ represents the margin for label $\lambda_i$, with previous $o_i - 1$ labels as the augmented input. If $o_i = 1$, then $\gamma_i^1$ represents the margin for label $\lambda_i$, without augmented input. Then $Q'$ is redefined as $Q' = \sum_{i=1}^{q} \frac{1}{(\gamma_i^{o_i})^2}$.

### 3.3.1 Dynamic Programming Algorithm

To simplify the search algorithm mentioned before, we propose the CC-DP algorithm to find the globally optimal CC. Note that $Q' = \sum_{i=1}^{q} \frac{1}{(\gamma_i^{o_i})^2} = \frac{1}{(\gamma_q^{o_q})^2} + \cdots + \big[\frac{1}{(\gamma_{k+1}^{o_{k+1}})^2} + \sum_{j=1}^{k} \frac{1}{(\gamma_j^{o_j})^2}\big]$, we explore the idea of DP to iteratively optimize $Q'$ over a subset of $\mathcal{M}$

with the length of $1, 2, \cdots, q$. Finally, we can obtain the optimal $Q'$ over $\mathcal{M}$. Assume $i \in \{1, \cdots, q\}$. Let $V(i, \eta)$ be the optimal $Q'$ over a subset of $\mathcal{M}$ with the length of $\eta (1 \leq \eta \leq q)$, where the label order is ending by label $\lambda_i$. Suppose $M_i^\eta$ represent the corresponding label set for $V(i, \eta)$. When $\eta = q$, $V(i, q)$ be the optimal $Q'$ over $\mathcal{M}$, where the label order is ending by label $\lambda_i$. The DP equation is written as:

$$V(i, \eta + 1) = \min_{j \neq i, \lambda_i \notin M_j^\eta} \left\{ \frac{1}{(\gamma_i^{\eta+1})^2} + V(j, \eta) \right\} \tag{3.5}$$

where $\gamma_i^{\eta+1}$ is the margin for label $\lambda_i$, with $M_j^\eta$ as the augmented input. The initial condition of DP is: $V(i, 1) = \frac{1}{(\gamma_i^1)^2}$ and $M_i^1 = \{\lambda_i\}$. Then, the optimal $Q'$ over $\mathcal{M}$ can be obtained by solving $\min_{i \in \{1, \cdots, q\}} V(i, q)$. Assume the training of linear SVM takes $\mathcal{O}(nd)$. The CC-DP algorithm is shown as the following bottom-up procedure: from the bottom, we first compute $V(i, 1) = \frac{1}{(\gamma_i^1)^2}$, which takes $\mathcal{O}(nd)$. Then we compute $V(i, 2) = \min_{j \neq i, \lambda_i \notin M_j^1} \{\frac{1}{(\gamma_i^2)^2} + V(j, 1)\}$, which requires at most $\mathcal{O}(qnd)$, and set $M_i^2 = M_j^1 \cup \{\lambda_i\}$. Similarly, it takes at most $\mathcal{O}(q^2 nd)$ time complexity to calculate $V(i, q)$. Last, we iteratively solve this DP Equation, and use $\min_{i \in \{1, \cdots, q\}} V(i, q)$ to get the optimal solution, which requires at most $\mathcal{O}(q^3 nd)$ time complexity.

**Theorem 3** (Correctness of CC-DP)**.** *$Q'$ can be minimized by CC-DP, which means this Algorithm can find the globally optimal CC.*

The proof can be referred to in Appendix A.3.

### 3.3.2 Greedy Algorithm

We propose a CC-Greedy algorithm to find a locally optimal CC to speed up the CC-DP algorithm. To save time, we construct only one classifier chain with the locally optimal label order. Based on the training instances, we select the label from $\{\lambda_1, \lambda_2, \cdots, \lambda_q\}$ as the first label, if the maximum margin can be achieved over this label, without augmented input. The first label is denoted by $\zeta_1$. Then we select the label from the remainder as the second label, if the maximum margin can be achieved over this label with $\zeta_1$ as the augmented input. We continue in this way until the last label is selected. Finally, this algorithm will converge to the locally optimal CC. We present the details of the CC-Greedy algorithm in Appendix A.4, where the time complexity of this algorithm is $\mathcal{O}(q^2 nd)$.

### 3.3.3 Tree-Based Algorithm

For multi-label problem, CC-DP and CC-Greedy are very time-consuming for the data sets with many labels. We propose Tree-DP and Tree-Greedy algorithms to further

speed up CC-DP and CC-Greedy, respectively, which scale linearly with $q$. We create a tree recursively in a top-down manner.

Assume that $\{\mathbf{x}_t, \mathbf{y}_t\}_{t=1}^n$ is the input data for the root node. Suppose that the label set $\{\lambda_1, \lambda_2, \cdots, \lambda_q\}$ in the root node is randomly split into two subsets with about the same size for left and right child nodes: $leftset = \{\lambda_1, \cdots, \lambda_{q/2}\}$ and $rightset = \{\lambda_{q/2+1}, \cdots, \lambda_q\}$. A training example can be considered annotated with $leftset$ and $rightset$ if it is annotated with at least one of the labels in $leftset$ and $rightset$, respectively. In this way, $leftset$ and $rightset$ can be seen as two labels. Then, in the root node, we train the CC with $leftset$ and $rightset$ as two labels using CC-DP or CC-Greedy. After that, left and right child nodes only keep the examples that are annotated with $leftset$ and $rightset$, respectively. This approach recurses into each child node that contains more than a single label.

Starting from the root node, we use the trained CC classifier on this node for prediction and we follow the recursive process. Finally, this process may lead to the prediction of some labels with corresponding to some leaves. We provide the following corollary pertaining to the Tree-DP.

**Corollary 3.** *After building a tree using the Tree-DP algorithm, we can find the globally optimal CC in each decision node of the tree.*

*Proof.* Given the structure of the tree, according to Theorem 3, we can find the globally optimal label order in each decision node using CC-DP algorithm. □

For each internal node, we only deal with two labels, thus the training time of Tree-DP and Tree-Greedy only take $\mathcal{O}(8nd)$ and $\mathcal{O}(4nd)$, respectively. The number of internal nodes in such a tree is equal to $q - 1$. In total, Tree-DP and Tree-Greedy take $\mathcal{O}(8(q-1)nd)$ and $\mathcal{O}(4(q-1)nd)$ training time, respectively. Assume that the testing instance goes along $\iota$ paths in our tree during the testing procedure and the depth of the tree is $\log(q)$. In each decision node, we take $\mathcal{O}(d)$ time for testing. Totally, the testing time for Tree-DP and Tree-Greedy is $\mathcal{O}(\iota \log(q) d)$.

## 3.4 Complexity Analysis

Assume that the training of linear SVM takes $\mathcal{O}(nd)$ time complexity. Following the running time analysis in Read et al. [2009], assume $q < d$, CC will takes $\mathcal{O}(ndq)$ time complexity. Let $\psi = \max\{n, d\}$. Table 3.1 reports the training and testing time complexity of the methods used in this chapter. From Table 3.1, we can see that our proposed algorithms are much faster than CCA and MMOC in terms of both training and testing time complexity, and achieve the same testing time complexity with BR, CC and ECC. Through the training time for our algorithms is slower than BR, CC and ECC. Our extensive empirical studies demonstrate that our algorithms achieve superior performance than those baselines.

Table 3.1: Time complexity comparisons among CC-Greedy, CC-DP and other baselines.

| Method | Training time complexity | Testing time complexity |
|---|---|---|
| BR,CC,ECC | $\mathcal{O}(ndq)$ | $\mathcal{O}(dq)$ |
| CCA | $\mathcal{O}(\psi^3 + n(d^2 + q^2 + dq))$ | $\mathcal{O}(q^3)$ |
| MMOC | $\mathcal{O}(nq^3 + nq^2d + n^4)$ | $\mathcal{O}(q^3)$ |
| CC-Greedy | $\mathcal{O}(q^2nd)$ | $\mathcal{O}(dq)$ |
| CC-DP | $\mathcal{O}(q^3nd)$ | $\mathcal{O}(dq)$ |
| Tree-Greedy | $\mathcal{O}(4(q-1)nd)$ | $\mathcal{O}(\iota \log(q)d)$ |
| Tree-DP | $\mathcal{O}(8(q-1)nd)$ | $\mathcal{O}(\iota \log(q)d)$ |

## 3.5 Experiment

In this section, we perform experimental studies on a number of benchmark data sets from different domains to evaluate the performance of our proposed algorithms for multi-label classification. All the methods are implemented in Matlab and all experiments are conducted on a workstation with a 3.2GHZ Intel CPU and 4GB main memory running 64-bit Windows platform.

### 3.5.1 Data Sets and Baselines

We conduct experiments on some real-world data sets with various domains from three websites.[1][2][3] Following the experimental settings in Dembczynski et al. [2010] and Zhang and Schneider [2012], we preprocess the eurlex_sm and eurlex_ed data sets. The statistics on those data sets are presented in Table 3.2. We compare our algorithms with some baseline methods: BR, CC, ECC, CCA Zhang and Schneider [2011] and MMOC Zhang and Schneider [2012]. To perform a fair comparison, we use the same linear classification/regression package LIBLINEAR Fan et al. [2008] with $l_2$-regularized square hinge loss (primal) to train the classifiers for all the methods. ECC is averaged over several CC predictions with random order and the ensemble size in ECC is set to ten according to Dembczynski et al. [2010]; Read et al. [2009]. In our experiment, the running time of PCC and EPCC Dembczynski et al. [2010] on most data sets, like slashdot and yahoo_art, takes more than one week. From the results in Dembczynski et al. [2010], ECC is comparable with EPCC and outperforms PCC, so we do not

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

[2] http://meka.sourceforge.net/#datasets

[3] http://cse.seu.edu.cn/people/zhangml/Resources.htm#data

Table 3.2: Data sets used in the experiments of Chapter 3.

| Data | # inst. | # attr. | # labels | Domain |
|---|---|---|---|---|
| yeast | 2,417 | 103 | 14 | biology |
| image | 2,000 | 294 | 5 | image |
| slashdot | 3,782 | 1,079 | 22 | text |
| enron | 1,702 | 1,001 | 53 | text |
| LLog | 799 | 1,004 | 10 | linguistics |
| yahoo_art | 6,849 | 23,146 | 10 | art |
| eurlex_sm_10 | 11,454 | 5,000 | 10 | text |
| eurlex_ed_10 | 6,540 | 5,000 | 10 | text |
| eurlex_sm | 19,348 | 5,000 | 201 | text |
| eurlex_ed | 19,348 | 5,000 | 3,993 | text |

Table 3.3: The Example-F1 results of CC-Greedy, CC-DP and other baselines on the various data sets (mean $\pm$ standard deviation). The best results are in bold. Numbers in square brackets indicate the rank.

| Data set | BR | CC | ECC | CCA | MMOC | CC-Greedy | CC-DP |
|---|---|---|---|---|---|---|---|
| yeast | 0.6076 $\pm$ 0.019[6] | 0.5850$\pm$ 0.033[7] | 0.6096$\pm$ 0.018[5] | 0.6109 $\pm$ 0.024[4] | 0.6132 $\pm$ 0.021[3] | **0.6144**$\pm$ 0.021[1] | 0.6135$\pm$ 0.015[2] |
| image | 0.5247 $\pm$ 0.025[7] | **0.5991**$\pm$ 0.021[1] | 0.5947$\pm$ 0.015[4] | 0.5947 $\pm$ 0.009[4] | 0.5960 $\pm$ 0.012[3] | 0.5939$\pm$ 0.021[6] | 0.5976$\pm$ 0.015[2] |
| slashdot | 0.4898 $\pm$ 0.024[6] | 0.5246$\pm$ 0.028[4] | 0.5123$\pm$ 0.027[5] | 0.5260 $\pm$ 0.021[3] | 0.4895 $\pm$ 0.022[7] | 0.5266$\pm$ 0.022[2] | **0.5268**$\pm$ 0.022[1] |
| enron | 0.4792 $\pm$ 0.017[7] | 0.4799$\pm$ 0.011[6] | 0.4848$\pm$ 0.014[4] | 0.4812 $\pm$ 0.024[5] | **0.4940** $\pm$ 0.016[1] | 0.4894$\pm$ 0.016[2] | 0.4880$\pm$ 0.015[3] |
| LLog | 0.3138 $\pm$ 0.022[6] | 0.3219$\pm$ 0.028[4] | 0.3223$\pm$ 0.030[3] | 0.2978 $\pm$ 0.026[7] | 0.3153 $\pm$ 0.026[5] | 0.3269$\pm$ 0.023[2] | **0.3298**$\pm$ 0.025[1] |
| yahoo_art | 0.4840 $\pm$ 0.023[5] | 0.5013$\pm$ 0.022[4] | 0.5070$\pm$ 0.020[3] | - | - | 0.5131$\pm$ 0.015[2] | **0.5135**$\pm$ 0.020[1] |
| eurlex_sm_10 | 0.8594 $\pm$ 0.003[5] | **0.8609**$\pm$ 0.004[1] | 0.8606$\pm$ 0.003[3] | - | - | 0.8600$\pm$ 0.004[4] | **0.8609**$\pm$ 0.004[1] |
| eurlex_ed_10 | 0.7170 $\pm$ 0.012[5] | 0.7176$\pm$ 0.012[4] | 0.7183$\pm$ 0.013[2] | - | - | 0.7183$\pm$ 0.013[2] | **0.7190**$\pm$ 0.013[1] |
| Average Rank | 5.88 | 3.88 | 3.63 | 4.60 | 3.80 | 2.63 | 1.50 |

consider PCC and EPCC here. CCA and MMOC are two state-of-the-art encoding-decoding Hsu et al. [2009] methods. We cannot get the results of CCA and MMOC on yahoo_art, eurlex_sm_10 and eurlex_ed_10 data sets in one week.

We consider the Example-F1, Macro-F1 and Micro-F1 to measure the prediction performance of all methods fairly. We perform 5-fold cross-validation on each data set and report the mean and standard error of each evaluation measurement.

### 3.5.2 Prediction Performance

The results of Example-F1, Macro-F1 and Micro-F1 for our method and baseline approaches in respect of the different data sets are reported in Tables 3.3, 3.4 and 3.5. From these results, we can see that:

Table 3.4: The Macro-F1 results of CC-Greedy, CC-DP and other baselines on the various data sets (mean ± standard deviation). The best results are in bold. Numbers in square brackets indicate the rank.

| Data set | BR | CC | ECC | CCA | MMOC | CC-Greedy | CC-DP |
|---|---|---|---|---|---|---|---|
| yeast | 0.3543 ± 0.014[4] | **0.3993**± 0.027[1] | 0.3763± 0.015[2] | 0.3496 ± 0.017[5] | 0.3431 ± 0.016[7] | 0.3441± 0.016[6] | 0.3596± 0.020[3] |
| image | 0.5852 ± 0.012[7] | **0.6013**± 0.018[1] | 0.5988± 0.010[4] | 0.6010 ± 0.009[2] | 0.5975 ± 0.007[6] | 0.5987± 0.019[5] | 0.6010± 0.014[2] |
| slashdot | 0.3416 ± 0.014[4] | 0.3485± 0.015[2] | 0.3331± 0.011[7] | **0.3512** ± 0.018[1] | 0.3334 ± 0.009[6] | 0.3431± 0.010[3] | 0.3408± 0.008[5] |
| enron | 0.2089 ± 0.024[2] | 0.2066± 0.022[5] | 0.2088± 0.022[3] | 0.1594 ± 0.027[6] | 0.1539 ± 0.017[7] | **0.2090**± 0.024[1] | 0.2082± 0.022[4] |
| LLog | 0.3452 ± 0.030[2] | 0.3428± 0.033[4] | 0.3425± 0.039[5] | 0.3189 ± 0.035[7] | 0.3303 ± 0.040[6] | 0.3448± 0.032[3] | **0.3471**± 0.035[1] |
| yahoo_art | 0.4836 ± 0.014[4] | 0.4816± 0.013[5] | 0.4851± 0.015[3] | - | - | 0.4876± 0.012[2] | **0.4884**± 0.015[1] |
| eurlex_sm_10 | 0.8546 ± 0.002[5] | 0.8558± 0.002[2] | 0.8554± 0.002[3] | - | - | 0.8550± 0.002[4] | **0.8559**± 0.003[1] |
| eurlex_ed_10 | 0.7201 ± 0.008[5] | 0.7202± 0.008[4] | 0.7205± 0.009[3] | - | - | 0.7208± 0.009[2] | **0.7217**± 0.008[1] |
| Average Rank | 4.13 | 3.00 | 3.75 | 4.20 | 6.40 | 3.25 | 2.25 |

- BR generally underperforms. Our experiment provides empirical evidence that the label correlations exist in many real word data sets and because BR ignores the information about the correlations between the labels, BR achieves poor performance on most data sets.
- CC improves the performance of BR, however, it underperforms ECC. This result verifies the answer to our first question stated in Section 3.1: the label order does affect the performance of CC; ECC, which averages over several CC predictions with random order, improves the performance of CC.
- CC-DP and CC-Greedy outperforms CCA and MMOC. This studies verify that optimal CC achieve competitive results compared with state-of-the-art encoding-decoding approaches.
- Our proposed CC-DP and CC-Greedy algorithms are successful on most data sets. This empirical result also verifies the answers to the last two questions stated in Section 3.1: the globally optimal CC exists and CC-DP can find the globally optimal CC which achieves the best prediction performance; the CC-Greedy algorithm achieves comparable prediction performance with CC-DP, while it requires lower time complexity than CC-DP. Our extensive empirical studies show that our algorithms achieve superior performance than those baselines.

### 3.5.3 Training Time and Testing Time

Figures 3.1 and 3.2 show the training and testing time of CC-Greedy, CC-DP and baseline methods on various data sets, respectively. According to these two figures, we can see that:

Table 3.5: The Micro-F1 results of CC-Greedy, CC-DP and other baselines on the various data sets (mean ± standard deviation). The best results are in bold. Numbers in square brackets indicate the rank.

| Data set | BR | CC | ECC | CCA | MMOC | CC-Greedy | CC-DP |
|---|---|---|---|---|---|---|---|
| yeast | 0.6320 ± 0.019[4] | 0.6185± 0.029[7] | 0.6306± 0.017[5] | **0.6362** ± 0.025[1] | 0.6361 ± 0.021[2] | 0.6303± 0.022[6] | 0.6328± 0.017[3] |
| image | 0.5840 ± 0.015[7] | 0.5994± 0.017[2] | 0.5955± 0.012[5] | **0.6003** ± 0.010[1] | 0.5958± 0.011[4] | 0.5946± 0.019[6] | 0.5980± 0.013[3] |
| slashdot | 0.5233 ± 0.024[6] | 0.5278± 0.027[3] | 0.5175± 0.025[7] | **0.5844** ± 0.022[1] | 0.5720 ± 0.022[2] | 0.5266± 0.023[5] | 0.5272± 0.023[4] |
| enron | 0.5052 ± 0.013[6] | 0.5013± 0.009[7] | 0.5056± 0.010[5] | 0.5335 ± 0.015[2] | **0.5401** ± 0.010[1] | 0.5104± 0.013[3] | 0.5096± 0.012[4] |
| LLog | **0.3768** ± 0.028[1] | 0.3712± 0.030[6] | 0.3730± 0.035[5] | 0.3623 ± 0.027[7] | 0.3760 ± 0.027[3] | 0.3744± 0.028[4] | 0.3762± 0.029[2] |
| yahoo_art | 0.5122 ± 0.017[5] | 0.5130± 0.016[4] | 0.5156± 0.018[3] | - | - | **0.5184**± 0.013[1] | **0.5184**± 0.017[1] |
| eurlex_sm_10 | 0.8718 ± 0.001[5] | 0.8727± 0.001[2] | 0.8725± 0.001[3] | - | - | 0.8722± 0.001[4] | **0.8733**± 0.002[1] |
| eurlex_ed_10 | 0.7419 ± 0.009[5] | 0.7421± 0.009[4] | 0.7424± 0.010[3] | - | - | 0.7425± 0.010[2] | **0.7432**± 0.010[1] |
| Average Rank | 4.88 | 4.38 | 4.50 | 2.40 | 2.40 | 3.88 | 2.38 |

- Our proposed algorithms are much faster than CCA and MMOC in terms of both training and testing time.
- CC-Greedy and CC-DP achieve comparable testing time with BR, CC and ECC. Through the training time of our algorithms is slower than BR, CC and ECC, our extensive empirical studies show that our algorithms achieve superior prediction performance than those baselines.
- CC-Greedy algorithm is much faster than CC-DP in terms of training time, whereas it achieves comparable prediction performance with CC-DP.

### 3.5.4 Results of Many Labels

This subsection studies the performance of Tree-Greedy, Tree-DP and other baselines on eurlex_sm and eurlex_ed data sets with many labels. We cannot get the results of CCA and MMOC on eurlex_sm and eurlex_ed data sets in one week. And we also cannot get the results of ECC on eurlex_ed data set in one week. Prediction performance of Tree-Greedy, Tree-DP and baselines is reported in Tables 3.6, 3.7 and 3.8. From the results, we can see that: our proposed Tree-Greedy and Tree-DP algorithms consistently outperform BR, CC and ECC on the data sets with many labels.

Training and testing time of Tree-Greedy, Tree-DP and baselines on eurlex_sm and eurlex_ed data sets are shown in Figure 3.3. According to this figure, we can observe that compared to BR, CC and ECC, our algorithms maintain the testing time over an acceptable threshold, while we are much faster than baselines in terms of training time.

### 3.5.5 Comparisons with Deep Learning Methods

ADIOS Cissé et al. [2016] is a state-of-the-art deep learning architecture for solving multiple class and label tasks. Unlike traditional deep learning methods that use a flat

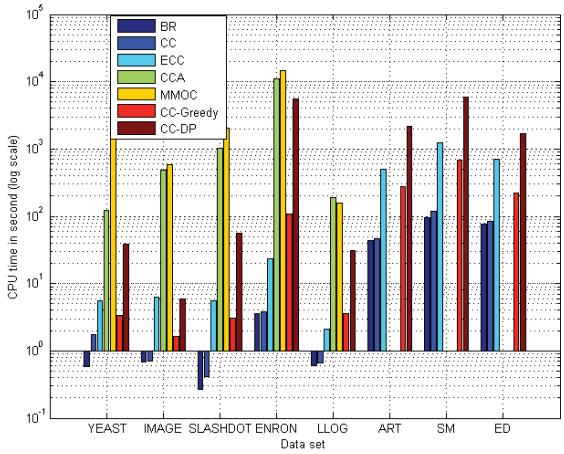


Figure 3.1: The training time of CC-DP, CC-Greedy and other baselines on various data sets. yahoo_art, eurlex_sm_10 and eurlex_ed_10 are abbreviated to ART, SM and ED, respectively.

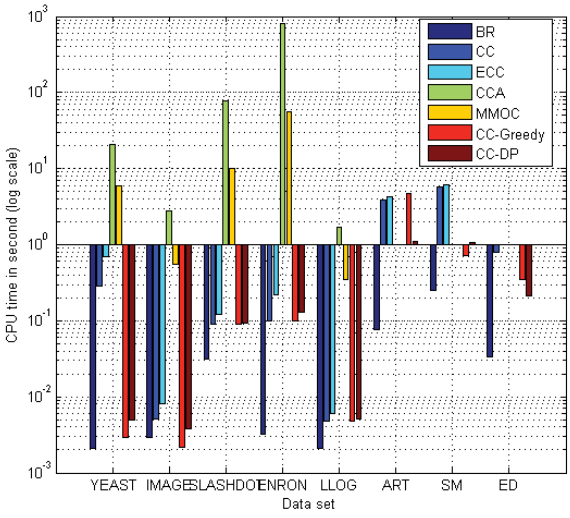


Figure 3.2: The testing time of CC-DP, CC-Greedy and other baselines on various data sets. yahoo_art, eurlex_sm_10 and eurlex_ed_10 are abbreviated to ART, SM and ED, respectively.

Table 3.6: The Example-F1 results on eurlex_sm and eurlex_ed data sets (mean ± standard deviation). The best results are in bold. Numbers in square brackets indicate the rank. "-" denotes the training time is more than one week.

| Data set | BR | CC | ECC | Tree-Greedy | Tree-DP |
|---|---|---|---|---|---|
| eurlex_sm | 0.6970±0.02[5] | 0.7233±0.01[4] | 0.7263±0.01[3] | 0.7292±0.01[2] | **0.7301**±0.01[1] |
| eurlex_ed | 0.4345±0.03[4] | 0.4528±0.02[3] | - | 0.4550±0.01[2] | **0.4563**±0.01[1] |
| Average Rank | 4.5 | 3.5 | 3 | 2 | 1 |

Table 3.7: The Macro-F1 results on eurlex_sm and eurlex_ed data sets (mean ± standard deviation). The best results are in bold. Numbers in square brackets indicate the rank. "-" denotes the training time is more than one week.

| Data set | BR | CC | ECC | Tree-Greedy | Tree-DP |
|---|---|---|---|---|---|
| eurlex_sm | 0.4777±0.02[5] | 0.4785±0.02[4] | 0.4800±0.02[3] | 0.4817±0.01[2] | **0.4834**±0.01[1] |
| eurlex_ed | 0.1660±0.01[4] | 0.1812±0.00[3] | - | 0.1844±0.00[2] | **0.1848**±0.00[1] |
| Average Rank | 4.5 | 3.5 | 3 | 2 | 1 |

output layer, ADIOS aims to capture the complex dependency between labels/classes to improve deep learning methods. Their approach is to split the label/class set into two subsets, $G_1$ and $G_2$, such that given $G_1$, the labels/classes in $G_2$ are independent. Our strategy to leverage label/class dependency is very different from that of ADIOS. We can extend the classifier chain model for multi-class classification (CCMC) and find the optimal class ordering, which can be referred to our journal paper "An Easy-to-hard Learning Paradigm for Multiple Classes and Multiple Labels". After that, we use the predictions of classifiers from easier classes to train the classifiers for harder classes. As such, the assumptions and constraints used in ADOIS are not applicable in our model.

This subsection conducts the experiments on the ILSVRC2012 data set[1]. It contains 1,000 object categories. Due to the limit of computational resources, we randomly sample 58,700 training instances and 31,300 testing instances from the ILSVRC2012 data set. We use the source code provided by the authors of ADIOS with default parameters. According to Cissé et al. [2016], we use one hidden layer with 1024 rectified linear units (ReLUs) Glorot et al. [2011] between inputs and $G_1$, and another 512-dimensional ReLUs between the hidden layer before $G_1$ and $G_2$ as well as direct connections between $G_1$ and $G_2$. We also compare with VGG Simonyan and Zisserman [2014] and residual nets (ResNet) He et al. [2016]. Both VGG and ResNet use a flat output layer, in which do not model the dependency between the classes Cissé et al. [2016], so the rich structure information among classes is missing

[1]http://www.image-net.org/challenges/LSVRC/2012/

Table 3.8: The Micro-F1 results on eurlex_sm and eurlex_ed data sets (mean ± standard deviation). The best results are in bold. Numbers in square brackets indicate the rank. "-" denotes the training time is more than one week.

| Data set | BR | CC | ECC | Tree-Greedy | Tree-DP |
|---|---|---|---|---|---|
| EURLEX_SM | 0.7321±0.01[5] | 0.7422±0.02[3] | 0.7363±0.01[4] | 0.7440±0.01[2] | **0.7454**±0.01[1] |
| EURLEX_ED | 0.4200±0.01[4] | 0.4477±0.01[3] | - | 0.4527±0.00[2] | **0.4549**±0.00[1] |
| AVERAGE RANK | 4.5 | 3 | 4 | 2 | 1 |

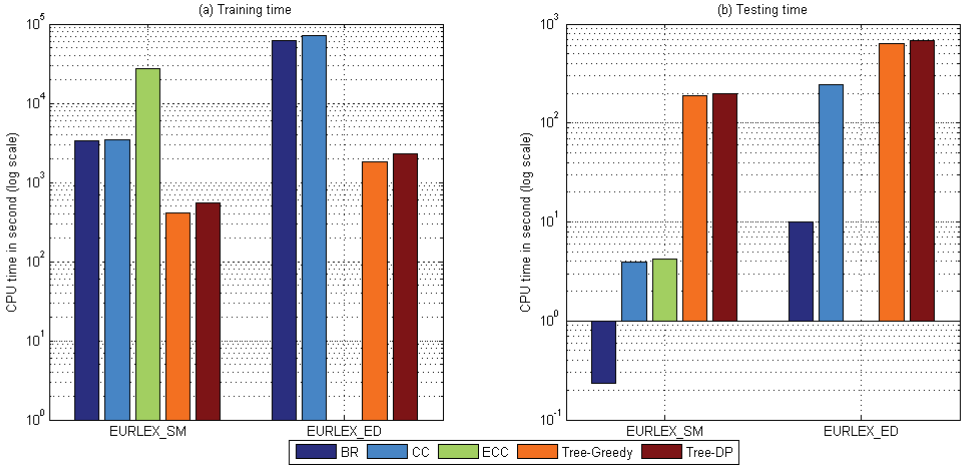


Figure 3.3: The training and testing time of BR, CC, ECC, Tree-Greedy and Tree-DP on eurlex_sm and eurlex_ed data sets.

in VGG and ResNet. We use the source code provided by the respective authors with default parameters. Following He et al. [2016], we use the 34-layer residual nets due to the limit of computational resources, and also extract 2048-dimensional features by the ResNet-34. According to Simonyan and Zisserman [2014], we extract 4096-dimensional features from the 16-layer of the VGG. Here, $\omega$ is set to 10, 30 and 50 for our method.

The classification results are reported in Table 3.9. From this table, we can observe that

- ADIOS outperforms VGG and ResNet-34, which verifies ADIOS's claim: existing deep learning approaches do not take into account the often unknown but nevertheless rich relationships between classes, this knowledge about the rich class structure (and other deep structure in data) is sometime referred to as dark knowledge (e.g. by Hinton et al. [2015] and Ba and Caruana [2014]).

Table 3.9: Testing error rate (in %) of VGG, ResNet-34, ADIOS and CCMC-FG on the ILSVRC2012 data set.

| METHOD | VGG | RESNET-34 | ADIOS | CCMC-FG+VGG FEATURES | CCMC-FG+RESNET FEATURES |
|---|---|---|---|---|---|
| TESTING ERROR | 23.95 | 21.51 | 21.28 | 23.98 ($\omega = 10$) | 21.85 ($\omega = 10$) |
| | | | | 22.67 ($\omega = 30$) | 20.11 ($\omega = 30$) |
| | | | | 20.73 ($\omega = 50$) | 19.84 ($\omega = 50$) |

- Without the restriction of the assumptions and constraints used in ADOIS, our method achieves better performance than ADIOS with the increasing value of $\omega$.
- CCMC-FG with ResNet features obtains better performance than CCMC-FG with VGG features, which demonstrates that our proposed method can be further improved based on better features.
- Based on the deep learning features, CCMC-FG consistently improves VGG and ResNet-34 with the increasing value of $\omega$. The results validate our analysis and the better ordering of classes obtains the better performance. Note that the above mentioned results were obtained using 58,700 training data points. We conclude that with limited data, the usage of structure information is helpful. We conjecture that this advantage may ultimately vanish as more and more data becomes available for training.

## 3.6 Summary of This Chapter

To improve the performance of multi-label classification, a plethora of models have been developed to capture label correlations. Amongst them, classifier chain is one of the most popular approaches due to its simplicity and good prediction performance. Instead of proposing a new learning model, we discuss three important questions in this work regarding the optimal classifier chain stated in Section 3.1. To answer these questions, we first propose a generalized CC model. We then provide a theoretical analysis of the generalization error for the proposed generalized model. Based on our results, we obtain the answer to the second question: the globally optimal CC exists only if the minimization of the upper bound is achieved over this CC. It is very expensive to search over $q!$ different label orders to find the globally optimal CC. Thus, we propose the CC-DP algorithm to simplify the search algorithm, which requires $\mathcal{O}(q^3nd)$ complexity. To speed up the CC-DP algorithm, we propose a CC-Greedy algorithm to find a locally optimal CC, where the time complexity of the CC-Greedy algorithm is $\mathcal{O}(q^2nd)$. Furthermore, we propose Tree-DP and Tree-Greedy algorithms to further speed up CC-DP and CC-Greedy, respectively, which scale linearly with $q$.

Comprehensive experiments on some real-world multi-label data sets from different domains verify our theoretical studies and the effectiveness of proposed algorithms.

# Chapter 4

# Large Margin Metric Learning for Multi-label Classification

## 4.1 Motivations

A simple approach to multi-label learning is binary relevance (BR) Tsoumakas et al. [2010], which trains a binary classifier for each label independently. To deal with many labels, Hsu et al. [2009] assume that label vectors have a little support. In other words, each label vector can be projected into a lower dimensional compressed label space, which can be deemed as **encoding**. A regression is then learned for each compressed label. Lastly, the compressed sensing (CS) is used to **decode** the labels from the regression outputs of each testing instance.

Many works have recently been developed in this **encoding-decoding** paradigm. These works mainly use different projection methods to transform the original label space into another effective label space as encoding. For example, principal label space transformation (PLST) Tai and Lin [2012] uses principal component analysis (PCA) to project the output labels into a lower dimension. Canonical correlation analysis (CCA) and maximum margin output coding (MMOC) are proposed in Zhang and Schneider [2011] and Zhang and Schneider [2012] to encode the original label space. Learning models are then trained in the transformed label space. To accurately recover the labels from the prediction of the learning models, expensive decoding schemes are usually required. Due to the combinatorial nature of the label space, exact decoding is usually intractable. Even when approximate inference Zhang and Schneider [2011] is used, the decoding step is still very time-consuming when there are many labels.

Comprehensive experiments on large scale image databases show that the $k$ nearest neighbor ($k$NN) algorithm achieves superior performance when handling many class problems Deng et al. [2010]. The performance of $k$NN depends significantly on the metric used to compute the distance between different instances. Moreover, Kwok and

Tsang [2003] and Weinberger and Saul [2009] show that the single-label prediction performance of $k$NN can be improved by learning a distance metric to satisfy the following constraint: Two nearby instances from different classes will be pushed further apart with a large margin. However, our experiment results show that existing advanced metric learning techniques, such as the popular DML Weinberger and Saul [2009], cannot provide an appropriate distance metric for multi-label classification, because nearby instances may differ by only a few labels or by hundreds in a multi-label setting. Based on structural SVMs McAllester [2006]; Tsochantaridis et al. [2005], MMOC Zhang and Schneider [2012] has recently been proposed to learn the proper metric for multi-label problems. Unfortunately, MMOC suffers from two major limitations: 1) Inconsistent performance: McAllester [2006] has already proved that structural SVMs fail to converge on the optimal decoder even with infinite training data. 2) Prohibitive computational cost: the training of MMOC involves a complex quadratic programming (QP) problem over the combinatorial space, and its computational cost on the data sets with many labels is prohibitive. Therefore, it is non-trivial to break the bottlenecks of MMOC, and develop efficient and consistent algorithms for solving multi-label learning tasks.

This chapter systematically studies how to efficiently learn an appropriate distance metric for solving multi-label learning tasks with provable guarantee. To achieve our goal, we present a novel large margin metric learning paradigm for multi-label classification to project both the input and output into the same embedding space. Moreover, we enforce the constraint that, in the embedding space, the distance between input $x^{(a)}$ and its correct output $y^{(a)}$ should be smaller than the distance between $x^{(a)}$ and the output $y^{(b)}$ of the nearest neighbors of $x^{(a)}$ with at least a margin measured by $\Delta(y^{(a)}, y^{(b)})$, the difference between $y^{(a)}$ and $y^{(b)}$. Thus, two nearby instances from different outputs will be pushed further apart by $\Delta(y^{(a)}, y^{(b)})$.

The main contributions in this chapter are:

1. To avoid the inconsistent performance of MMOC, we present a novel large margin metric learning paradigm for multi-label classification. Our theoretical analysis shows that our proposed model converges to the optimal solutions, and also reduces the generalization error for multi-label classification.

2. To incorporate the feature and label correlations, we project both the input and output to the same embedding space, in which the input and output can be compared. A large margin formulation with $k$ nearest neighbor constraints is proposed to learn the embedding space. Lastly, we transform the formulation to metric learning Kulis [2013]; Yang and Jin [2006] for multi-label problems.

3. After transformation, our optimization problem is reduced to a semidefinite programming problem. To handle many labels, the accelerated proximal

gradient (APG) method Beck and Teboulle [2009]; Toh and Yun [2009] is adapted to solve the reduced problem.

4. To avoid the expensive decoding step, we select $k$ nearest neighbors from the training set for each testing instance in the embedding space and make a rapid prediction based on the labels of those $k$ nearest neighbors.

5. Experiments on a number of real-world multi-label data sets demonstrate that our method outperforms state-of-the-art approaches and is more efficient than methods that are based on the expensive decoding step.

## 4.2 Large Margin Metric Learning

### 4.2.1 Preliminaries

Assume $x^{(i)} \in \mathbb{R}^{p\times 1}$ is a real vector representing an input (instance), $y^{(i)} \in \{0,1\}^{q\times 1}$ is a real vector representing the corresponding output ($i \in \{1 \dots n\}$). $n$ denotes the number of training samples. The input matrix is $X \in \mathbb{R}^{n\times p}$ and the output matrix is $Y \in \{0,1\}^{n\times q}$. $Nei(i)$ is the output set of $k$ nearest neighbors of input instance $x^{(i)}$. For output encoding, $V = (v_1, v_2, ...., v_d) \in \mathbb{R}^{q\times d}$ ($d < q$) is the projection matrix that maps each output vector $y^{(i)}$ ($q$ dimension) to $V^T y^{(i)}$ ($d$ dimension). Let $P \in \mathbb{R}^{p\times q}$ also be the projection matrix. For input encoding, each input vector $x^{(i)}$ ($p$ dimension) is projected to $V^T P^T x^{(i)}$ ($d$ dimension). Then $x^{(i)}$ and $y^{(i)}$ can be compared in the projection space ($d$ dimension).

A simple linear regression model for BR is to learn the matrix $P$ through the following formulation:

$$argmin_{P\in R^{p\times q}} \frac{1}{2}||P^T X^T - Y^T||_F^2 \tag{4.1}$$

where $||\cdot||_F$ is the Frobenius norm. However, this does not consider the relationships between labels. To reduce the noise in the data set, MMOC Zhang and Schneider [2012] incorporates the feature and label correlations and proposes a maximum margin output coding formulation Eq. (4.2) to learn the projection matrix.

$$\begin{aligned} &argmin_{V\in R^{q\times d},\{\xi_i\geq 0\}_{i=1}^n} \frac{1}{2}||V||_F^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ &s.t. \quad ||V^T P^T x^{(i)} - V^T y^{(i)}||_2^2 + \Delta(y^{(i)}, y) - \xi_i \\ &\qquad \leq ||V^T P^T x^{(i)} - V^T y||_2^2, \forall y \in \{0,1\}^q, \forall i \end{aligned} \tag{4.2}$$

Eq. (4.2) involves an exponentially large number of constraints. To address the combinatorial nature of the label space $\{0,1\}^q$, Zhang and Schneider [2012] use the

overgenerating technique Finley and Joachims [2008] with the cutting plane method. Training involves solving a box-constrained quadratic programming (QP) problem for each training sample $i$, which is time-consuming, and testing also involves solving a QP on $\{0,1\}^q$ space. Even if approximate inference Zhang and Schneider [2011] is used to solve this QP problem, it is still computationally expensive.

### 4.2.2 Proposed Formulation

Inspired by $k$NN and MMOC, we propose the following large margin metric learning with $k$ nearest neighbor constraints, or LM-$k$NN for short, to learn the projection matrix.

If the encoding scheme works well, the distance between the codeword of $x^{(i)}$ ($V^T P^T x^{(i)}$) and the codeword of $y^{(i)}$ ($V^T y^{(i)}$) should tend to 0 and be less than the distance between the codeword of $x^{(i)}$ and the codeword of any other output ($V^T y$). Then, the following large margin formulation is presented to learn projection matrix $V$:

$$\begin{aligned} & argmin_{V \in R^{q \times d}, \{\xi_i \geq 0\}_{i=1}^n} \frac{1}{2}||V||_F^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i^2 \\ & s.t. \quad ||V^T P^T x^{(i)} - V^T y^{(i)}||_2^2 + \Delta(y^{(i)}, y) - \xi_i \\ & \qquad \leq ||V^T P^T x^{(i)} - V^T y||_2^2, \forall y \in Nei(i), \forall i \end{aligned} \tag{4.3}$$

where $C$ is a positive constant that controls the trade-off between square-hinge loss function and regularizer. The constraints in Eq. (4.3) guarantee that the distance between the codeword of $x^{(i)}$ and the codeword of $y^{(i)}$ is less than the distance between the codeword of $x^{(i)}$ and the codeword of any other output. To give Eq. (4.3) more robustness, we add the loss function $\Delta(y^{(i)}, y)$ as the margin and minus the slack variable $\xi_i$ in the left side of the constraints. Given the distance function $||V^T P^T x^{(i)} - V^T y^{(i)}||_2^2$ as the compatibility function, Eq. (4.3) can be viewed as an adapted form of structural SVMs Tsochantaridis et al. [2005]. The loss function is defined as $\Delta(y^{(i)}, y) = ||y^{(i)} - y||_1$ Zhang and Schneider [2012], where $||\cdot||_1$ means the $l_1$ norm. Following Weinberger and Saul [2009], we use Euclidean metric to measure the distances between instances $x^{(i)}$ and $x^{(j)}$ and then learn a new distance metric, which improves the performance of $k$NN.

Define a $q \times q$ symmetric positive semidefinite matrix (denoted by $S_q^+$) $Q$: $Q = VV^T \in S_q^+$ and $\phi_{x^{(i)}, y^{(i)}} = P^T x^{(i)} - y^{(i)}$. We can transform Eq. (4.3) to the following

metric learning Kulis [2013]; Yang and Jin [2006] problem:

$$\begin{aligned} & argmin_{Q\in S_q^+,\{\xi_i\geq 0\}_{i=1}^n} \frac{1}{2}trace(Q) + \frac{C}{n}\sum_{i=1}^{n}\xi_i^2 \\ & s.t. \quad \phi_{x^{(i)},y^{(i)}}^T Q \phi_{x^{(i)},y^{(i)}} + \Delta(y^{(i)},y) - \xi_i \\ & \qquad\qquad \leq \phi_{x^{(i)},y}^T Q \phi_{x^{(i)},y}, \forall y \in Nei(i), \forall i \end{aligned} \tag{4.4}$$

### 4.2.3 Accelerated Proximal Gradient Update

Since our objective of Eq. (4.4) is smooth and the number of constrains is linear w.r.t n, we can apply the accelerated proximal gradient (APG) method Beck and Teboulle [2009]; Toh and Yun [2009] to efficiently solve the primal form of Eq. (4.4) with many labels. Let $p(Q) = \frac{1}{2}trace(Q)$ and $f(Q) = \frac{C}{n}\sum_{i=1}^{n}\xi_i^2$ where $\xi_i = \max\{0, \max_{y\in Nei(i)}(\Delta(y^{(i)},y) - (\phi_{x^{(i)},y}^T Q\phi_{x^{(i)},y} - \phi_{x^{(i)},y^{(i)}}^T Q\phi_{x^{(i)},y^{(i)}}))\}$. We define

$$F(Q) = f(Q) + p(Q), Q \in S_q^+ \tag{4.5}$$

The derivative of $f$ is denoted by $\nabla f$. Yuan et al. [2012] show that $\nabla f$ is Lipschitz continuous on $Q$. For any $Z \in S_q^+$, consider the following QP problem of $F(Q)$ at $Z$:

$$\begin{aligned} A_\tau(Q,Z) &= f(Z) + < \nabla f(Z), Q - Z > \\ &\qquad + \frac{\tau}{2}\|Q - Z\|_F^2 + p(Q) \\ &= \frac{\tau}{2}\|Q - G\|_F^2 + p(Q) + f(Z) - \frac{1}{2\tau}\|\nabla f(Z)\|_F^2 \end{aligned} \tag{4.6}$$

where $\tau > 0$ is a constant and $G = Z - \frac{1}{\tau}\nabla f(Z)$. To minimize $A_\tau(Q,Z)$ w.r.t. $Q$, it is reduced to solve Eq. (4.7):

$$argmin_{Q\in S_q^+}\frac{\tau}{2}||Q - G||_F^2 + p(Q) \tag{4.7}$$

To solve Eq. (4.7), we take the derivative of the objective function of Eq. (4.7) w.r.t. $Q$: $\tau(Q - G) + \frac{1}{2}I = 0$, then $Q = G - \frac{1}{2\tau}I$. We take the SVD of $G$ as $G = U\overline{G}U^T$, and $Q = U\overline{G}U^T - \frac{1}{2\tau}UU^T$, then $Q = U(\overline{G} - \frac{1}{2\tau}I)U^T$. We use 0 to replace the negative entries in $\overline{G} - \frac{1}{2\tau}I$. Lastly, we obtain the symmetric positive semidefinite matrix solution of Eq. (4.7), denoted by $S_\tau(G)$. The detailed APG algorithm is shown in Algorithm 1:

In Algorithm 1, let $L_f$ be the Lipschitz constant of $\nabla f$ and $L_f$ estimated as $L_f = 0.01nC$. The optimality condition of Eq. (4.4) is $\nabla F(Q) = 0$. It is usually time-consuming to achieve this condition. In practice, we meet an $\epsilon$-accurate solution instead. The stopping condition of Algorithm 1 is set as:

$$\frac{F(Q^\kappa) - F(Q^{\kappa+1})}{F(Q^\kappa)} \leq \epsilon \tag{4.8}$$

**Algorithm 1** Accelerated Proximal Gradient Algorithm for Solving Eq. (4.4)

**Input:** $\eta \in (0,1)$ is a constant. Choose $Q^0 = Q^{-1} \in S_q^+$. $t^0 = t^{-1} = 1$ and $\kappa = 0$. Choose the Lipschitz constant $L_f$ and set $\tau_0 = L_f$

**Output:** The optimal solution to Eq. (4.4)

1: Set $Z^\kappa = Q^\kappa + \frac{t^{\kappa-1}-1}{t^\kappa}(Q^\kappa - Q^{\kappa-1})$
2: Set $\tau = \eta\tau_\kappa$
3: **for** $j = 0, 1, 2, \ldots,$ **do**
4: Set $G = Z^\kappa - \frac{1}{\tau}\nabla f(Z^\kappa)$, compute $S_\tau(G)$
5: **if** $F(S_\tau(G)) \leq A_\tau(S_\tau(G), Z^\kappa)$, **then**
6: set $\tau_\kappa = \tau$, stop
7: **else**
8: $\tau = \frac{1}{\eta}\tau$
9: **end if**
10: **end for**
11: Set $Q^{\kappa+1} = S_\tau(G)$
12: Compute $t^{\kappa+1} = \frac{1+\sqrt{1+4(t^\kappa)^2}}{2}$. Let $\kappa = \kappa + 1$
13: Quit if stopping condition is achieved. Otherwise, go to step 1

where $\epsilon$ is a small tolerance value. In practice, we set $\epsilon = 0.001$. Lastly, a sublinear convergence rate of algorithm 1 is guaranteed in the following theorem.

**Theorem 4.** *Let $\{Q^\kappa\}$ be the sequences generated by Algorithm 1 and $L_f$ be the Lipschitz constant of $\nabla f$. Then for any $\kappa \geq 1$, we have*

$$F(Q^\kappa) - F(Q^*) \leq \frac{2L_f||Q^0 - Q^*||_F^2}{\eta(\kappa+1)^2} \tag{4.9}$$

*where $Q^* = argmin_Q F(Q)$*

The proof can be adapted from Beck and Teboulle [2009].

### 4.2.4 Prediction

Traditionally, encoding-decoding methods involve the decoding process which usually requires solving QP problem on a combinatorial space. It is computationally expensive. Inspired by metric learning Kulis [2013], we select $k$ nearest neighbors from the training set for each testing instance in the embedding space, and conduct prediction based on the codeword distance with a testing instance and the labels of $k$ nearest neighbors. For a new testing input variable $x$, we find $k$ instances $\{x^{(1)}, \ldots, x^{(k)}\}$ in the training set which have a smaller codeword distance from $x$ than other instances. Based on the codeword distance with $x$ and output of $\{x^{(1)}, \ldots, x^{(k)}\}$

Table 4.1: Time complexity comparisons between LM-$k$NN and other baselines.

| Method | Training Time | Testing Time |
|---|---|---|
| BR | $\mathcal{O}(npq)$ | $\mathcal{O}(pq)$ |
| PLST | $\mathcal{O}(n^3 + q^2n + npq)$ | $\mathcal{O}(q^2 + pq)$ |
| CCA | $\mathcal{O}(\psi^3 + n(p^2 + q^2 + pq))$ | $\mathcal{O}(q^3)$ |
| MMOC | $\mathcal{O}(nq^3 + n^4)$ | $\mathcal{O}(q^3)$ |
| CPLST | $\mathcal{O}(q^3 + nq^2 + p^3)$ | $\mathcal{O}(q^2 + pq)$ |
| $k$NN | - | $\mathcal{O}(pn)$ |
| ML-$k$NN | $\mathcal{O}(n^2p + nq)$ | $\mathcal{O}(pn + q)$ |
| LM-$k$NN | $\mathcal{O}(q^3 + knpq^2)$ | $\mathcal{O}(qn + pq)$ |

, we compute the scores for each label for $x$. Lastly, we make the prediction for multi-label classification problems based on the scores. The equation of the distance between the codeword of $x$ and the codeword of $x^{(i)}$ is $||\hat{M}(x) - \hat{M}(x^{(i)})||_2^2$. It can be computed as $(P^Tx - P^Tx^{(i)})^TQ(P^Tx - P^Tx^{(i)})$. Thus, the testing time is similar to $k$NN and much faster than the encoding-decoding methods. Following the setting in MMOC, we set 0.5 as the threshold without further optimization.

### 4.2.5 Complexity Analysis

**Training time** The formulation of MMOC involves an exponential number of constraints. The authors therefore use the overgenerating technique with the cutting plane method. Lastly, training involves solving a box-constrained QP problem for each training instance and then using CVX[1] to solve a semidefinite programming problem. The time complexity of QP is at least $\mathcal{O}(q^3)$, and from Pedrycz [2002], the training time complexity of MMOC is $\mathcal{O}(nq^3 + n^4)$ for each iteration at least. While the training time of our method (LM-$k$NN) is dominated by the APG algorithm. To achieve an $\epsilon$-solution, the number of iterations needed by APG update is $\mathcal{O}(\frac{1}{\sqrt{\epsilon}})$. The time complexity for each iteration is $\mathcal{O}(q^3 + knpq^2)$.

**Testing time** We analyze the testing time for each testing instance. Both the testing time of CCA and MMOC involve solving QP on $\{0, 1\}^q$ space. It is combinatorial in nature and intractable, thus mean-field approximation is used to obtain the approximated solution iteratively. The time complexity for each iteration is $\mathcal{O}(q^2)$, but it takes many times to converge. The time complexity is $\mathcal{O}(q^3)$ at least. The training and testing time complexity for the methods that are used in this chapter is presented in Table 4.1. Let $\psi = \max\{n, p, q\}$.

[1] http://cvxr.com/cvx/

## 4.3 Generalization Error Analysis

This section will study the excess risk bounds of LM-$k$NN for multi-label classification.

Suppose our learning model LM-$k$NN is characterized by a distribution $\mathcal{D}$ on the space of inputs and labels $\mathcal{X} \times \{0,1\}^q$, where $\mathcal{X} \subseteq \mathbb{R}^p$. Let a sample $(x^{(j)}, y^{(j)})$ be drawn from the distribution $\mathcal{D}$, where $y^{(j)} \in \{0,1\}^q$ ($j \in \{1,\ldots,n\}$) are the ground truth label vectors. Assume $n$ samples $D = \{(x^{(1)}, y^{(1)}), \cdots, (x^{(n)}, y^{(n)})\}$ are drawn i.i.d. $n$ times from the distribution $\mathcal{D}$, which is denoted by $D \sim \mathcal{D}^n$. For two inputs $x^{(z)}, x^{(j)}$ in $\mathcal{X}$, We define $d(x^{(z)}, x^{(j)}) = ||x^{(z)} - x^{(j)}||_2$ as the Euclidean metric in the original input space and $d_{pro}(x^{(z)}, x^{(j)}) = \|V^T M^T x^{(z)} - V^T M^T x^{(j)}\|_2$ as our learned metric in the embedding input space. Let $f^D_{knn_i}(x)$ represent the prediction of $i$-th label for input $x$ using our model LM-$k$NN for multi-label classification (MLC), which is trained on $D$. The MLC prediction performance of LM-$k$NN: $(f^D_{knn_1}(\cdot), \cdots, f^D_{knn_q}(\cdot)) : \mathcal{X} \to \{0,1\}^q$ is then measured in terms of its generalization error, which is its expected loss on a new example $(x, y)$ drawn according to $\mathcal{D}$:

$$E\Big(\sum_{i=1}^{q} \ell(y_i, f^D_{knn_i}(x))\Big) \tag{4.10}$$

where $y_i$ means the $i$-th label and $\ell(y_i, f^D_{knn_i}(x))$ represents the loss function for the $i$-th label. We define the loss function as the following for the analysis.

$$\ell(y_i, f^D_{knn_i}(x)) = P(y_i \neq f^D_{knn_i}(x)) \tag{4.11}$$

For $i$-th label, we define the function as follows:

$$\nu^i_j(x) = P(y_i = j|x), j \in \{0,1\}. \tag{4.12}$$

The Bayes optimal classifier $b^*$ for $i$-th label is defined as

$$b^*_i(x) = \arg\max_{j \in \{0,1\}} \nu^i_j(x) \tag{4.13}$$

Before deriving our results, we first present several important definitions and theorems.

**Definition 2** (Doubling Dimension, Kontorovich and Weiss [2014]; Krauthgamer and Lee [2004]). *Let $(\mathcal{X}, d)$ be a metric space, let $\bar{\lambda}$ be the smallest value such that every ball in $\mathcal{X}$ can be covered by $\bar{\lambda}$ balls of half the radius. The doubling dimension of $\mathcal{X}$ is defined as : $ddim(\mathcal{X}) = \log_2(\bar{\lambda})$.*

**Theorem 5** (Kontorovich and Weiss [2014]; Krauthgamer and Lee [2004]). *Let $(\mathcal{X}, d)$ be a metric space. The diameter of $\mathcal{X}$ is defined as $diam(\mathcal{X}) = \sup_{x,x' \in \mathcal{X}} d(x, x')$. The $\varepsilon$-covering number of $\mathcal{X}$, $\mathcal{N}(\varepsilon, \mathcal{X}, d)$, is bounded by:*

$$\mathcal{N}(\varepsilon, \mathcal{X}, d) \leq \Big(\frac{2diam(\mathcal{X})}{\varepsilon}\Big)^{ddim(\mathcal{X})} \tag{4.14}$$

We provide the following generalization error bound of LM-1NN for MLC:

**Theorem 6.** *Given a metric space $(\mathcal{X}, d_{pro})$, assume function $\nu^i : \mathcal{X} \to \{0, 1\}$ is Lipschitz with constant $L$ with respect to the sup-norm for each label. Suppose $\mathcal{X}$ has a finite doubling dimension: $ddim(\mathcal{X}) = \mathbb{D} < \infty$ and $diam(\mathcal{X}) = 1$. Let $D = \{(x^{(1)}, y^{(1)}), \cdots, (x^{(n)}, y^{(n)})\}$ and $(x, y)$ be drawn i.i.d. from the distribution $\mathcal{D}$. Then, we have*

$$\begin{aligned} &E\Big(\sum_{i=1}^{q} P(y_i \neq f^D_{1nn_i}(x))\Big) \\ &\leq \sum_{i=1}^{q} 2P(b^*_i(x) \neq y_i) + \frac{3qL||M||_F\sqrt{trace(Q)}}{n^{1/(\mathbb{D}+1)}} \end{aligned} \tag{4.15}$$

The proof of this theorem can be found in Appendix A.5.

Inspired by Theorem 19.5 in Shalev-Shwartz and Ben-David [2014], we derive the following lemma of LM-$k$NN for MLC:

**Lemma 3.** *Given metric space $(\mathcal{X}, d_{pro})$, assume function $\nu^i : \mathcal{X} \to \{0, 1\}$ is Lipschitz with constant $L$ with respect to the sup-norm for each label. Suppose $\mathcal{X}$ has a finite doubling dimension: $ddim(\mathcal{X}) = \mathbb{D} < \infty$ and $diam(\mathcal{X}) = 1$. Let $D = \{(x^{(1)}, y^{(1)}), \cdots, (x^{(n)}, y^{(n)})\}$ and $(x, y)$ be drawn i.i.d. from the distribution $\mathcal{D}$. Then, we have*

$$\begin{aligned} &E\Big(\sum_{i=1}^{q} P(y_i \neq f^D_{knn_i}(x))\Big) \\ &\leq \sum_{i=1}^{q}(1 + \sqrt{8/k})P(b^*_i(x) \neq y_i) + \frac{q(6L||M||_F\sqrt{trace(Q)} + k)}{n^{1/(\mathbb{D}+1)}} \end{aligned} \tag{4.16}$$

The proof of this lemma can be found in Appendix A.6. The following corollary reveals important statistical properties of LM-1NN and LM-$k$NN for MLC.

**Corollary 4.** *For MLC, as $n$ goes to infinity, the error of LM-1NN and LM-$k$NN converges to the sum of twice the Bayes error and $1 + \sqrt{8/k}$ times the Bayes error over the labels, respectively.*

Our results reveal that an important factor in reducing the generalization error bound for the LM-1NN and LM-$k$NN is to minimize $trace(Q)$, which is equivalent to optimizing Problem (4.4). Thus, minimizing the objective of our LM-$k$NN tightens the generalization error bound for MLC.

Table 4.2: Data sets used in the experiments of Chapter 4.

| Data Set | # Instances | # Features | # Labels |
|---|---|---|---|
| scene | 2,407 | 294 | 6 |
| cal500 | 502 | 68 | 174 |
| corel5k | 5,000 | 499 | 374 |
| delicious | 16,105 | 500 | 983 |
| EUR-Lex (dc) | 19,348 | 5,000 | 412 |
| EUR-Lex (ed) | 19,348 | 5,000 | 3,993 |

## 4.4 Experiment

In this section, we evaluate the performance of our proposed Large Margin $k$NN (LM-$k$NN) for multi-label prediction. All the methods compared are implemented in MatLab. All experiments are conducted on a workstation with a 3.4GHZ Intel CPU and 32GB main memory running Linux platform.

### 4.4.1 Experimental Setup

#### 4.4.1.1 Data Sets

We conduct experiments on a variety of real-world data sets from different domains[1] (Table 4.2).

- scene Boutell et al. [2004]: Collects images of outdoor scenes.
- cal500 Turnbull et al. [2008]: Contains songs by different artists. Each song is labeled by 174 tags representing genres, instruments, emotions, and other related concepts.
- corel5k Duygulu et al. [2002]: Contains images from Stock Photo CDs. In total, there are 374 labels.
- delicious Tsoumakas et al. [2008]: Contains textual data of web pages along with 983 tags extracted from the del.icio.us social book marking site.
- Eur-Lex Loza Mencía and Fürnkranz [2008b]: Collects documents on European Union law. There are several EuroVoc descriptors, directory codes and types of subject matter to describe the labels. Here, we use two of them which have more labels.

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

#### 4.4.1.2 Baseline Methods

We compare our LM-$k$NN with several state-of-the-art multi-label prediction methods:

- BR Tsoumakas et al. [2010].
- PLST Tai and Lin [2012]: Uses principal component analysis (PCA) for encoding, and rounding for decoding.
- CPLST Chen and Lin [2012a]: Conditional principal label space transformation (CPLST) is proposed to learn the embedding space, which considers both the label and the feature parts.
- CCA Zhang and Schneider [2011]: Uses canonical correlation analysis to encode the label vectors, and mean-field approximation is applied in the decoding step.
- MMOC Zhang and Schneider [2012]: Adapts a maximum margin criterion to learn output coding for multi-labels. Its decoding scheme is the same as CCA.
- $k$NN: We adapt the $k$ nearest neighbor ($k$NN) algorithm to solve multi-label classification problems. Euclidean metric is used to measure the distances between instances.
- ML-$k$NN Zhang and Zhou [2007b]: Based on $k$NN, the maximum posteriori principle is used by this method to determine the labels of the testing instance.

For BR, we use linear classification/regression package LIBLINEAR Fan et al. [2008] with L2-regularized logistic regression (primal) to train the classifier. As with the experimental settings in Zhang and Schneider [2012], the number of output projections $d$ is set to the number of original labels $q$ ($d = q$) for PLST, CCA, MMOC and CPLST and the decoding parameter is set as $\lambda = 1$ for CCA and MMOC. In our experiment, we find that the performance of kNN or ML-kNN have no significant difference on most data sets with varying $k$. Following the setting in Zhang and Zhou [2007b], we set $k = 10$ for $k$NN, ML-$k$NN and our method. $\eta = 0.4$ is set for the APG algorithm and $C = 10$ is set for MMOC and our method. We set the stopping condition threshold as $\epsilon = 0.01$ for EUR-Lex (ed) data set. Besides PLST Tai and Lin [2012], CCA Zhang and Schneider [2011] and MMOC Zhang and Schneider [2012] have also been shown to outperform the original compressed-sensing-based method Hsu et al. [2009], thus we make no comparison with Hsu et al. [2009] in this chapter.

To fairly measure the performance of our method and the baseline methods, we consider Hamming Loss, Micro-F1 and Example-F1 as the evaluation measurements. We perform 10-fold cross-validation on each data set and report the mean and standard error of each evaluation measurement.

Table 4.3: The Hamming Loss results of LM-$k$NN and other baselines on the various data sets (mean $\pm$ standard deviation). The best results are in bold.

| Data Set | BR | PLST | CCA | MMOC | CPLST | $k$NN | ML-$k$NN | LM-$k$NN |
|---|---|---|---|---|---|---|---|---|
| scene | 0.1109 $\pm$ 0.01 | 0.1113 $\pm$ 0.01 | 0.0942 $\pm$ 0.01 | 0.0936 $\pm$ 0.01 | 0.1114 $\pm$ 0.01 | **0.0902** $\pm$ 0.01 | 0.0951 $\pm$ 0.01 | 0.0917 $\pm$ 0.01 |
| cal500 | 0.1557 $\pm$ 0.01 | 0.1471 $\pm$ 0.00 | 0.1680 $\pm$ 0.01 | - | 0.1433 $\pm$ 0.01 | 0.1453 $\pm$ 0.00 | 0.1495 $\pm$ 0.01 | **0.1423** $\pm$ 0.00 |
| corel5k | 0.0200 $\pm$ 0.01 | 0.0097 $\pm$ 0.00 | - | - | 0.0094 $\pm$ 0.00 | 0.0095 $\pm$ 0.00 | 0.0094 $\pm$ 0.00 | **0.0086** $\pm$ 0.00 |
| delicious | 0.0194 $\pm$ 0.00 | 0.0194 $\pm$ 0.00 | - | - | 0.0190 $\pm$ 0.00 | 0.0197 $\pm$ 0.00 | 0.0198 $\pm$ 0.00 | **0.0183** $\pm$ 0.00 |
| EUR-Lex (dc) | 0.0097 $\pm$ 0.00 | 0.0070 $\pm$ 0.00 | - | - | 0.0053 $\pm$ 0.00 | 0.0047 $\pm$ 0.00 | 0.0049 $\pm$ 0.00 | **0.0015** $\pm$ 0.00 |
| EUR-Lex (ed) | 0.0034 $\pm$ 0.00 | 0.0029 $\pm$ 0.00 | - | - | 0.0028 $\pm$ 0.00 | 0.0023 $\pm$ 0.00 | 0.0024 $\pm$ 0.00 | **0.0011** $\pm$ 0.00 |

Table 4.4: The Micro-F1 results of LM-$k$NN and other baselines on the various data sets (mean $\pm$ standard deviation). The best results are in bold.

| Data Set | BR | PLST | CCA | MMOC | CPLST | $k$NN | ML-$k$NN | LM-$k$NN |
|---|---|---|---|---|---|---|---|---|
| scene | 0.6911 $\pm$ 0.03 | 0.5924 $\pm$ 0.03 | 0.7281 $\pm$ 0.03 | 0.7297 $\pm$ 0.03 | 0.6347 $\pm$ 0.02 | 0.7221 $\pm$ 0.02 | **0.7387** $\pm$ 0.03 | 0.7300 $\pm$ 0.03 |
| cal500 | 0.3448 $\pm$ 0.02 | 0.3032 $\pm$ 0.01 | 0.3533 $\pm$ 0.02 | - | 0.3536 $\pm$ 0.02 | **0.3593** $\pm$ 0.01 | 0.3184 $\pm$ 0.02 | 0.3549 $\pm$ 0.02 |
| corel5k | 0.0956 $\pm$ 0.03 | 0.0801 $\pm$ 0.01 | - | - | 0.1022 $\pm$ 0.01 | 0.1006 $\pm$ 0.01 | 0.0278 $\pm$ 0.01 | **0.1834** $\pm$ 0.02 |
| delicious | 0.2447 $\pm$ 0.01 | 0.1423 $\pm$ 0.01 | - | - | 0.1826 $\pm$ 0.00 | 0.2338 $\pm$ 0.01 | 0.1738 $\pm$ 0.01 | **0.3046** $\pm$ 0.01 |
| EUR-Lex (dc) | 0.7211 $\pm$ 0.02 | 0.2304 $\pm$ 0.03 | - | - | 0.3504 $\pm$ 0.03 | 0.6988 $\pm$ 0.01 | 0.6705 $\pm$ 0.01 | **0.7388** $\pm$ 0.01 |
| EUR-Lex (ed) | 0.4566 $\pm$ 0.01 | 0.1059 $\pm$ 0.01 | - | - | 0.1840 $\pm$ 0.01 | 0.4664 $\pm$ 0.01 | 0.3864 $\pm$ 0.01 | **0.4706** $\pm$ 0.02 |

### 4.4.2 Prediction Performance

Tables 4.3, 4.4 and 4.5 list the three measurement results for our method and baseline approaches in respect of the different data sets. Recall that CCA and MMOC are very computationally expensive, especially for solving the QP problem on $\{0, 1\}^q$ space which is combinatorial in nature and intractable during the decoding step, so they cannot be run on most of the larger data sets. In our experiment, the decoding procedure on corel5k data set already takes more than two days for CCA. Because MMOC has to solve a box-constrained QP for each training sample and use CVX to tackle optimization problems during the training step, it runs out of memory even for the cal500 data set with 68 features and 174 labels. From the results, we can see that:

- CCA or MMOC's performance is comparable to the best results on scene and cal500. This means that CCA and MMOC are most successful in small data sets with few labels, but cannot deal with larger data sets with many labels.
- In our experiment, PLST generally underperforms. Thus, PLST with simple linear projection in encoding and decoding procedure is not effective.
- CPLST outperforms PLST, which is consistent with the empirical results in Chen and Lin [2012a].

Table 4.5: The Example-F1 results of LM-$k$NN and other baselines on the various data sets (mean $\pm$ standard deviation). The best results are in bold.

| Data Set | BR | PLST | CCA | MMOC | CPLST | $k$NN | ML-$k$NN | LM-$k$NN |
|---|---|---|---|---|---|---|---|---|
| scene | 0.6169 $\pm$ 0.03 | 0.4588 $\pm$ 0.04 | **0.7320** $\pm$ 0.03 | 0.7025 $\pm$ 0.03 | 0.5390 $\pm$ 0.03 | 0.6815 $\pm$ 0.02 | 0.6874 $\pm$ 0.03 | 0.7101 $\pm$ 0.03 |
| cal500 | 0.3446 $\pm$ 0.02 | 0.3062 $\pm$ 0.01 | 0.3516 $\pm$ 0.02 | - | 0.3513 $\pm$ 0.02 | **0.3561** $\pm$ 0.01 | 0.3216 $\pm$ 0.02 | 0.3517 $\pm$ 0.02 |
| corel5k | 0.0781 $\pm$ 0.02 | 0.0587 $\pm$ 0.01 | - | - | 0.0757 $\pm$ 0.01 | 0.0773 $\pm$ 0.01 | 0.0178 $\pm$ 0.01 | **0.1517** $\pm$ 0.01 |
| delicious | 0.2093 $\pm$ 0.01 | 0.1131 $\pm$ 0.00 | - | - | 0.1530 $\pm$ 0.00 | 0.2094 $\pm$ 0.01 | 0.1518 $\pm$ 0.00 | **0.2665** $\pm$ 0.00 |
| EUR-Lex (dc) | 0.7133 $\pm$ 0.01 | 0.1530 $\pm$ 0.02 | - | - | 0.4461 $\pm$ 0.01 | 0.6569 $\pm$ 0.01 | 0.6038 $\pm$ 0.01 | **0.7230** $\pm$ 0.01 |
| EUR-Lex (ed) | 0.4313 $\pm$ 0.02 | 0.0725 $\pm$ 0.01 | - | - | 0.2429 $\pm$ 0.00 | 0.4207 $\pm$ 0.01 | 0.3385 $\pm$ 0.00 | **0.4630** $\pm$ 0.00 |

Table 4.6: Comparison between DML and LM-$k$NN in terms of Micro-F1 and Example-F1 (mean $\pm$ standard deviation). The best results are in bold.

| Data Set | Micro-F1 | | Example-F1 | |
|---|---|---|---|---|
| | DML | LM-$k$NN | DML | LM-$k$NN |
| scene | 0.3349 $\pm$ 0.01 | **0.7300** $\pm$ 0.03 | 0.3325 $\pm$ 0.02 | **0.7101** $\pm$ 0.03 |
| corel5k | 0.0188 $\pm$ 0.02 | **0.1834** $\pm$ 0.02 | 0.0171 $\pm$ 0.03 | **0.1517** $\pm$ 0.01 |

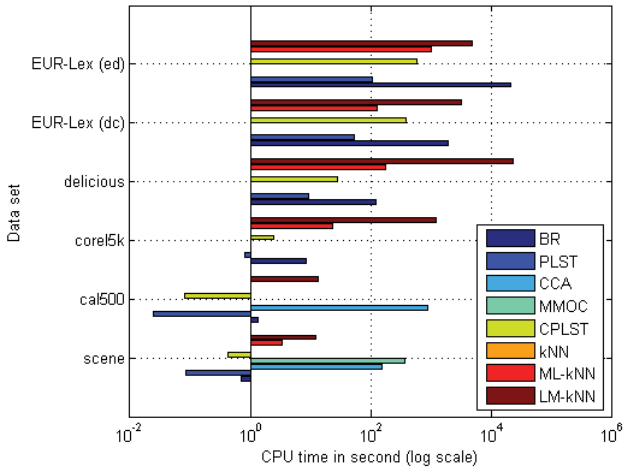


Figure 4.1: The training time of LM-$k$NN and other baseline methods on all data sets.

- BR is inferior to $k$NN and LM-$k$NN on most data sets, such as scene, cal500 and corel5k. BR does not consider the distributions and relationships between labels, so it usually achieves much lower accuracy on many data sets.
- LM-$k$NN outperforms MMOC and other baselines on most data sets in terms of Hamming Loss, which corroborates our theoretical results. Tables 4.4 and 4.5 show that LM-$k$NN also achieves superior performance in terms of both Micro-F1 and Example-F1 measurements. These results demonstrate that LM-$k$NN is the most successful method in terms of different measurements.

### 4.4.3 Comparison with DML

This section compares the performance of our method and DML Weinberger and Saul [2009] on scene and corel5k data sets. The results are shown in Table 4.6. From the results, we can see that DML generally underperforms on multi-label problems, whereas our method provides an appropriate distance metric for multi-label classification.

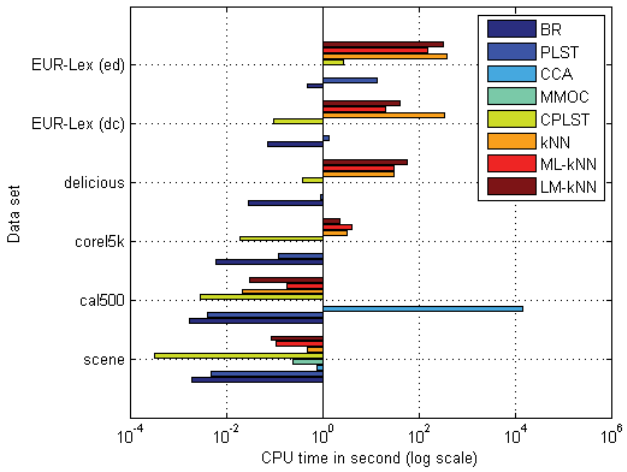


Figure 4.2: The testing time of LM-$k$NN and other baseline methods on all data sets.

### 4.4.4 Training Time and Testing Time

In this section, we compare the time performance of the various methods used in the experiment. Figures 4.1 and 4.2 report the once training and testing times of 10-fold cross-validation for our method and baseline approaches in terms of different data sets respectively. The results illustrate that PLST is fast in terms of training and testing time, which is consistent with the empirical results in Tai and Lin [2012], however, it is not effective in general. BR is fast in terms of testing time, however, it has slow training time and underperforms on the much larger data sets EUR-Lex (dc) and EUR-Lex (ed). Our method LM-$k$NN is almost 10 times faster than CCA and MMOC in terms of both the training and testing times even on smaller data sets, and is comparable to $k$NN and ML-$k$NN in terms of testing time.

## 4.5 Summary of This Chapter

To achieve the better performance than BR, based on structural SVMs, Zhang and Schneider [2012] proposed MMOC to incorporate the feature or label correlations. However, MMOC suffers from two major limitations: 1) Inconsistent performance: McAllester [2006] has already proved that structural SVMs fail to converge on the optimal decoder even with infinite training data. 2) Prohibitive computational cost: the training of MMOC involves a complex quadratic programming (QP) problem over the combinatorial space. Even if the overgenerating technique with the cutting plane method is used to solve this problem, it is still computationally very expensive. Therefore, it is non-trivial to break the bottlenecks of MMOC, and develop efficient

and consistent algorithms for solving multi-label learning tasks. Inspired by $k$NN and MMOC, we propose the large margin formulation with $k$ nearest neighbors constraints to solve the projection matrix which reduces the number of constraints from $\mathcal{O}(n2^q)$ to $\mathcal{O}(nk)$ and is more robust than $k$NN. Our problem is then transformed to a metric learning problem. To handle large scale applications, the accelerated proximal gradient (APG) method is adapted to solve the reduced semidefinite programming problem. Lastly, instead of performing expensive combinatorial optimization or approximate inference for the decoding procedure, we simply adopt the $k$NN strategy, which selects $k$ nearest neighbors from the training set for each testing instance in the projection space and makes rapid prediction based on the labels of those $k$ nearest neighbors. Furthermore, we provide the generalization error analysis for our proposed method. Our theoretical analysis shows that our proposed model converges to the optimal solutions, and also reduces the generalization error for multi-label classification. Overall, extensive experiments on a number of real-world multi-label data sets demonstrate that our method outperforms state-of-the-art approaches for accuracy. For scalability, our method is almost 10 times faster than CCA and MMOC in terms of both the training and testing times, and is comparable to $k$NN and ML-$k$NN in terms of testing time.

# Chapter 5

# Fast Prediction via Multi-Label Coding Tree

## 5.1 Motivations

One central challenging issue for practical multi-label learning is the scalability of prediction. Suppose one needs to predict the presence or absence of $q$ labels for $m$ input instances, simple approaches, like Read et al. [2009]; Tsoumakas et al. [2010], scaling the number of annotations to be linear with the number of labels, take $\mathcal{O}(q \times m)$ time for prediction. However, in real-world applications, like image prediction Deng et al. [2009], $m$ and $q$ can be very large and the cost of those exhaustive approaches quickly becomes prohibitive. We cannot get the results of these exhaustive approaches on the large-scale data sets within one week.

Recently, many works [Agrawal et al., 2013; Liu and Tsang, 2016; Prabhu and Varma, 2014; Tsoumakas et al., 2008] have put more effort on exploring tree-based algorithms to minimize the number of predictions for multi-label predictions with many labels. Among them, FastXML Prabhu and Varma [2014] is the most recently advanced technique, which has shown state-of-the-art rapid predictions. However, our extensive empirical study verifies that FastXML generally underperforms in terms of multi-label prediction performance. Moreover, some literature Madjarov et al. [2012]; Tsoumakas et al. [2008] has shown Homer Tsoumakas et al. [2008] achieves superior prediction performance. Unfortunately, we cannot get the results of Homer on medium-sized data sets within one week in our experiment. The question is: can we design some efficient, yet accurate multi-label prediction algorithms with the minimum number of predictions? This is the problem we address herein.

To answer this question, we first show three important phenomena of real-world multi-label learning data sets with many labels from various application domains[1]. We

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

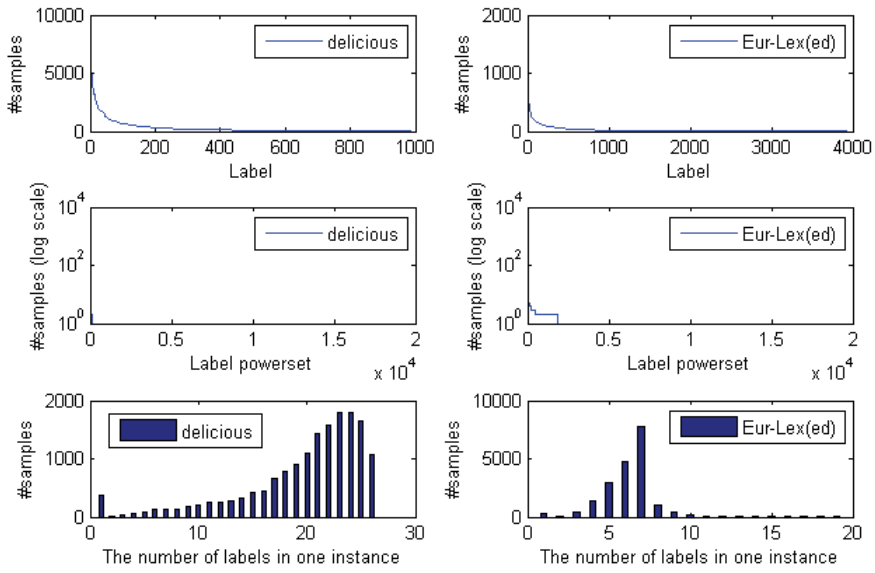


Figure 5.1: **Top**: Frequency of each label on the delicious and Eur-Lex(ed) data sets. **middle**: Frequency of each label powerset on the delicious and Eur-Lex(ed) data sets. **bottom**: Frequency of samples with the specific number of labels on the delicious and Eur-Lex(ed) data sets.

use delicious and Eur-Lex(ed) data sets as examples. The details of those data sets are shown in Table 4.2. The labels of instances usually follow the distribution of Power Law Barabási et al. [2000], as shown at the top of Figure 5.1, where many labels have very low label frequency. The bottom of Figure 5.1 shows the sparsity of labels in each instance, which means that the average number of labels in the data set is small. The middle of Figure 5.1 shows that very few label powersets have high frequency. Note that similar observations can also be found in image prediction tasks Mao et al. [2013].

In real-world problems, although the number of labels would be very large, the first observation indicates that only a few labels appear very frequently; many others are rare. Based on this property, we design a tree-based strategy to reduce the overall predictions as follows. Frequent labels should be predicted first in the decision tree, so there will be fewer predictions involved to predict the frequent labels; and more predictions for rare labels. To achieve this, we develop a multi-label coding tree based strategy, which is analogous to the Shannon-Fano coding Shannon [1948] strategy, to deal with multi-label prediction problems. Moreover, the second observation shows that each instance typically has few labels, which enables our tree model to stop earlier. Thus, fewer predictions will be made along shorter paths in our tree model for most testing instances. The idea of Huffman coding be easily used to deal with the transformed multi-class prediction task based on the property of the last observation. Therefore, our proposed coding tree framework reduces the number of predictions significantly.

Our main contributions are as follows:

1. We observe three important phenomena of real-world multi-label learning data sets.

2. To reduce the number of predictions, based on the properties of multi-label learning data sets, we employ the coding tree based strategy to deal with both the transformed multi-class prediction task and multi-label prediction problems in bottom-up and top-down manners, respectively.

3. We provide a theoretical analysis of the average number of predictions for the proposed methods.

4. Experiments on a number of real-world multi-label learning data sets demonstrate that our methods yield significantly fewer numbers of predictions than state-of-the-art approaches and achieve comparable prediction performance with the best one.

## 5.2 Coding Tree Framework

In this section, we will present the details of our proposed coding tree based algorithms for multi-label prediction problem and the transformed multi-class (label powerset) prediction tasks. Assume $\mathbf{x}_i \in \mathbb{R}^d$ is a real vector representing an input or instance (feature), $\mathcal{Y}_i \subseteq \{1, 2, \cdots, q\}$ is the corresponding output (label), where $q$ is the number of labels ($i \in \{1 \dots m\}$). $m$ denotes the number of training samples. $\mathbf{y}_i \in \{0, 1\}^q$ is used to represent the label set $\mathcal{Y}_i$, where $\mathbf{y}_i(j) = 1$ if and only if $j \in \mathcal{Y}_i$.

### 5.2.1 Label Powerset Prediction

Our goal is to minimize the number of classifiers for binary decisions to enable rapid label powerset prediction for a number of testing instances. Recall that, to minimize the expected codeword length of information, Huffman coding uses shorter codewords to encode more frequent symbols and longer codewords to encode less frequent symbols. The idea of Huffman coding can be easily adapted to deal with the label powerset prediction problems.

Given a random $m$ multi-labeled training sample, we first transform multi-label prediction to multi-class prediction by treating each unique set of labels as one of the classes for transformed multi-class learning task and record the frequency of each transformed class. Let $n$ be the number of transformed classes. A Huffman coding tree is built based on the $n$ classes and their corresponding frequencies. Leaf nodes of this Huffman coding tree are the set of transformed classes, denoted by $S = \{s_1, \cdots, s_n\}$, and the corresponding frequencies are represented by $P = \{p(s_1), \cdots, p(s_n)\}$. $T_{s_i} (i = 1, \cdots, n)$ denotes the training instances associated with each transformed class. If $\mathcal{S}$ is a set, $|\mathcal{S}|$ denotes its cardinality. We build a Huffman coding tree dealing with the label powerset case in a bottom-up manner. The details of the algorithm are presented in Algorithm 2.

More frequent classes are close to the root and less frequent classes are close to the leaf node layer. Thus, the prediction for most testing instances only require a few binary decisions along some shorter paths in our algorithm.

We use a simple example to demonstrate our Huffman coding tree. Assume one needs to predict the presence or absence of four labels $(a, b, c, e)$ for 8 input instances. Suppose $(1, 1, 0, 0)$, $(1, 0, 1, 0)$, $(0, 0, 0, 1)$ and $(0, 0, 1, 0)$ appear 3, 1, 3 and 1 times, respectively. Following Algorithm 2, we build the tree structure as shown in Figure 5.2.

### 5.2.2 Multi-label Prediction

For multi-label prediction problems, the hypothesis space is exponential and we can not observe some combinations of labels in the training data. Thus Huffman coding can not be used to tackle the multi-label prediction problem. To reduce the number of

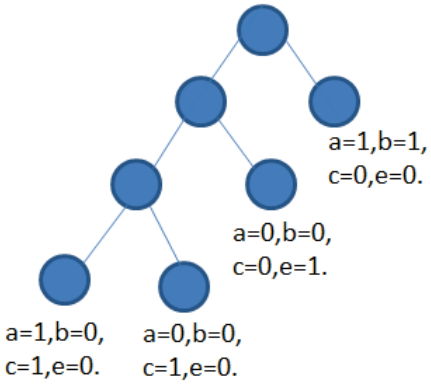


Figure 5.2: Schematic illustration of HCT.

**Algorithm 2** Huffman Coding Tree (HCT) for Label Powerset Prediction

**Input:** Given $n$ transformed classes $S$, corresponding frequencies $P$, and $T_{s_i}(i = 1, \cdots, n)$.

**Output:** A hierarchical tree for multi-label prediction

1: Initialize a priority queue $Q = \emptyset$ with the ascending order of the frequency
2: Create a leaf node for each transformed class $s_i$ and insert it to $Q$
3: **while** $|Q| > 1$ **do**
4: remove the two nodes $s_i$ and $s_j$ of the lowest frequency from $Q$
5: create a new internal node $s_k$ with $s_i$ and $s_j$ as children and its frequency equals to the sum of the two nodes' frequency
6: set the instance $T_{s_i}$ as positive sample and $T_{s_j}$ as negative sample, then train a classifier for the node $s_k$
7: insert the new node $s_k$ to $Q$
8: **end while**
9: The remaining node is the root node and the tree is complete

predictions, we develop a tree-based strategy, which is analogous to the Shannon-Fano coding strategy. We first train an accurate node classifier to predict the frequent label that have more examples. Then, we can leverage the predictions of such classifier to divide the remaining labels into two sets whose cardinality are roughly comparable. As long as any sets with more than one member remain, the same process is repeated on those sets. Our algorithm makes predictions for the ordered labels which accord to the arrangement with the label frequency ranging from high to low. The second property shows that each instance has very few labels, based on our algorithm, so only a few predictions will be made for most testing instances. We start with some definitions associated with the specific nodes in a tree.

Let $g$ represent the node in a tree, assume the label set associated with the node

$g$ is $G_g = \{1, 2, \cdots, q\}$, given a set of training instances $T_g = \{\mathbf{x}_i\}_{i=1}^m$ and the corresponding labels $L_g = \{\mathbf{y}_i\}_{i=1}^m$.

**Definition 3.** *( Zero Label Set) refers to the situation where all the training instances at the node $g$ do not belong to this label set. It defined as: $Zero_g = \{j|$ where all the training instances at the node $g$ do not belong to label $j$ $\}$.*

**Definition 4.** *( One Label Set) refers to the situation where all the training instances at the node $g$ belong to this label set. It defined as: $One_g = \{j|$ where all the training instances at the node $g$ belong to label $j$ $\}$.*

To accelerate the learning process, those zero and one label sets will be thrown away. Thus we define the working label set for node $g$: $W_g = G_g - Zero_g - One_g$. Now, we introduce the learning process for the tree in three cases.

1. **Root node:** Assume $g$ is the root node. We select label $l_g \in W_g$ with the highest frequency among $W_g$ and train a classifier for $l_g$. $g$'s left and right child are denoted as $g_1$ and $g_0$, respectively. The training instances of node $g$ will be partitioned into left child $g_1$ if the values of label $l_g$ of those instances is 1, otherwise the instances will be partitioned into right child $g_0$. The label set associated with node $g_1$ and $g_0$ will be reduced as $W_g \backslash l_g$. The training instance sets for $g_1$ and $g_0$ are denoted by $T_{g_1}$ and $T_{g_0}$.

2. **Internal node:** The training procedure for internal (child) nodes are the same with that for the root node but uses a subset of training instances. The training instances are split recursively into two smaller subsets until our algorithm moves down to the leaf nodes.

3. **Leaf node:** If the working set of the node is empty, then it is a leaf node and stop training for the leaf nodes.

We use the same example, which is aforementioned in Section 5.2.1, to illustrate our Shannon-Fano coding tree. Following Algorithm 3, we train a classifier for label $a$ in the root node. For internal nodes, we train the classifier for label $b$ and $e$ in the left and right child nodes of the root, respectively. The working set in leaf nodes is empty, so we do not train any classifiers for the leaf nodes.

We predict the multi-label output for a new testing instance from the root to leaves and assign the labels of the leaf node to this testing instance. The schematic illustration of our tree is shown in Figure 5.3 and the details of algorithm is presented as follows:

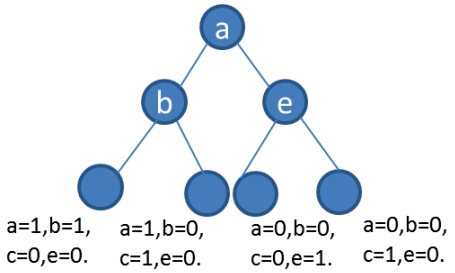


Figure 5.3: Schematic illustration of SFCT.

**Algorithm 3** Shannon-Fano Coding Tree (SFCT) for Multi-label Prediction

**Input:** Given node $g$ and its corresponding $G_g$, $T_g$ and $L_g$.
**Output:** A hierarchical tree for multi-label prediction

1: Compute $Zero_g$, $One_g$ and $W_g$
2: **while** $W_g \neq \emptyset$ **do**
3: select label $l_g \in W_g$ with the highest frequency
4: train a classifier for $l_g$
5: set the label set of left and right child as $W_g \backslash l_g$
6: split the training instances of node $g$ into left child $g_1$ if the values of label $l_g$ of those instances are 1 and call Algorithm 3 with input $(W_g \backslash l_g, T_{g_1})$
7: split the rest of training instances of node $g$ into right child $g_0$ and call Algorithm 3 with input $(W_g \backslash l_g, T_{g_0})$
8: **end while**
9: The tree is complete

## 5.3 Theoretical Analysis

### 5.3.1 Analysis on Number of Predictions

As the prediction efficiency critically depends on the number of prediction times, in this section, we study the average number of predictions for the both cases. We denote logarithms to base 2 by log.

We first introduce the noiseless coding theorem in Roman [1992]. Let $C = \{c_1, \cdots, c_n\}$ be the symbol set of size $n$. $p(c_i)$ is the probability that symbol $c_i$ occurs. After encoding of symbols in $C$, $l(c_i)$ denotes the length of code $c_i$ and the average codeword length is defined as $AveLen(c_1, \cdots, c_n) = \sum_{i=1}^{n} p(c_i)l(c_i)$. The entropy of $\{p(c_1), \cdots, p(c_n)\}$ is defined as $H(p(c_1), \cdots, p(c_n)) = \sum_{i=1}^{n} p(c_i) \log(1/p(c_i))$ (we denote logarithms to base 2 by log). We get the following noiseless coding theorem:

**Theorem 7.** *( The Noiseless Coding Theorem) For any probability distribution*

$\{p(c_1), \cdots, p(c_n)\}$, *we have*

$$H(p(c_1), \cdots, p(c_n)) \leq AveLen(c_1, \cdots, c_n) < H(p(c_1), \cdots, p(c_n)) + 1$$

**Label powerset prediction:** We define $H(P_{lp}) = \sum_{i=1}^{n} p(s_i) \log(1/p(s_i))$. $l(s_i)$ denotes the number of predictions for class $s_i$ and the average number of predictions is defined as $mean(S) = \sum_{i=1}^{n} p(s_i) l(s_i)$. Following Theorem 7, we obtain the bound of the average number of predictions for the label powerset prediction problem using Algorithm 2.

**Theorem 8.** *Assume we transform a random $m$ multi-labeled sample to a multi-class sample which has $n$ classes. Then, a hierarchical tree $T$ is built based on Algorithm 2. We obtain the bound of the average number of predictions for the transformed multi-class prediction problem using $T$:*

$$H(P_{lp}) \leq mean(S) < H(P_{lp}) + 1$$

**Multi-label prediction:** Assume there are $n$ leaf nodes in a hierarchical tree which is built according to Algorithm 3. Those leaf nodes denoted by $S' = \{s'_1, \cdots, s'_n\}$. Assume the frequency $p'_i$ of each unique label set can be associated with leaf node $s'_i$. We define $H(P_l) = \sum_{i=1}^{n} p(s'_i) \log(1/p(s'_i))$. Similar to label powerset prediction problem, we use $mean(S')$ to represent the average number of predictions of Algorithm 3. We obtain the bound of $mean(S')$ for the label case.

**Theorem 9.** *Given a random $m$ multi-labeled sample which has $n$ unique label set. A hierarchical tree $T$ is built based on Algorithm 3. We obtain the bound of the average number of predictions for multi-label prediction problem using $T$:*

$$H(P_l) \leq mean(S') < H(P_l) + 1$$

*Proof.* Assume there are $m$ multi-labeled samples which have $n$ unique label set. We build a hierarchical tree based on $m$ sample according to Algorithm 3. Each node is associated with one label, so the labels associated with nodes in each path of the tree are one of the unique label set, so there are $n$ number of paths and leaf nodes. The frequency of each unique label set can be associated with each leaf node. After this transformation, we apply Theorem 8 to obtain the above bounds. □

Therefore, in the label powerset and multi-label prediction algorithms, we can expect that the number of predictions for each instance are around $H(P_{lp})$ and $H(P_l)$, which are much fewer than $\min(m, 2^q)$ and $q$, respectively.

Table 5.1: Testing time complexity comparisons among HCT, SFCT and other baselines. $\Upsilon, \Psi, \iota, \mathfrak{D}$ : # clusters, the average # instances in each cluster, # learners and the dimension of the embedding space used in SLEEC.

| Method | Worst Testing Time | Average Testing Time |
|---|---|---|
| BR | $\mathcal{O}(dq)$ | $\mathcal{O}(dq)$ |
| LP | $\mathcal{O}(d\psi)$ | - |
| Homer | - | $\mathcal{O}(d\log_k(q))$ |
| FastXML | - | $\mathcal{O}(\mathcal{T}d\log(q))$ |
| SLEEC | - | $\mathcal{O}(\mathfrak{D}\Upsilon\iota + \mathfrak{D}\Psi\iota + d\mathfrak{D})$ |
| HCT | $\mathcal{O}(d\log(\psi))$ | $\mathcal{O}(dH(P_{lp}))$ |
| SFCT | $\mathcal{O}(dq)$ | $\mathcal{O}(dH(P_l))$ |

### 5.3.2 Testing Time Complexity Analysis

Algorithm 3 and Algorithm 2 are denoted as SFCT and HCT, respectively. We provide the testing time complexity analysis for each testing instance. The number of predictions for BR are equal to the number of labels. Each time Homer divides the problem into $k$ sub-problems. Following Tsoumakas et al. [2008], it takes $\mathcal{O}(d\log_k(q))$ on testing time. According to Prabhu and Varma [2014], the average cost of prediction for FastXML is $\mathcal{O}(\mathcal{T}d\log(q))$, where $\mathcal{T}$ is the number of trees in the FastXML ensemble. In the worst case, LP requires $\min(m, 2^q)$ times of predictions respectively. The number of predictions for HCT and SFCT are $\log(\min(m, 2^q))$ and $q$, respectively. As shown in our theoretical analysis, on average, the number of predictions can be further reduced to $H(P_{lp})$ and $H(P_l)$. The testing time complexity for the methods that are used in this chapter is presented in Table 5.1, where $\psi = \min(m, 2^q)$.

## 5.4 Experiment

In this section, we evaluate the performance of our proposed algorithms for multi-label prediction. All experiments are conducted on a workstation with a 3.2GHZ Intel CPU and 4GB main memory running 64-bit Windows.

### 5.4.1 Data Sets and Baselines

We conduct experiments on a variety of real-world data sets from website[1]. The details of data sets are shown in Table 5.2.

We compare our algorithms with several state-of-the-art multi-label annotation methods:

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

Table 5.2: Data sets used in the experiments of Chapter 5.

| Data Set | # Instances | # Features | # Labels |
|---|---|---|---|
| mediamill | 43,907 | 120 | 101 |
| cal500 | 502 | 68 | 174 |
| corel5k | 5,000 | 499 | 374 |
| Eur-Lex (dc) | 19,348 | 5,000 | 412 |
| Eur-Lex (ed) | 19,348 | 5,000 | 3,993 |

- Flat methods: BR Tsoumakas et al. [2010] and LP Tsoumakas et al. [2010]. BR trains a classifier for each label independently. LP trains a single multi-class classifier for the transformed multi-class prediction problem.
- Tree-based methods: Homer Tsoumakas et al. [2008] and FastXML Prabhu and Varma [2014]. Homer uses divide-and-conquer-strategy to divide original problem into $k$ sub-muilti-label prediction problems. FastXML learns a hierarchy over the feature space by directly optimizing a ranking loss function.
- Embedding-based method: Sparse local embedding for extreme classification (SLEEC) [Bhatia et al., 2015] is a state-of-the-art encoding-decoding method.

The codes are provided by the respective authors. According to Tsoumakas et al. [2008], $k$ is chosen in a range of $\{2, 3, \cdots, 8\}$ using cross validation for Homer. Following Prabhu and Varma [2014], $\mathcal{T}$ is fixed to 50 for FastXML. We use the linear classification/regression package LIBLINEAR Fan et al. [2008] with $l_2$-regularized square hinge loss (primal) to train the classifier for both BR and LP. We use the default parameter in LIBLINEAR. According to the original setting in [Bhatia et al., 2015], the number of learners as 15 and the dimension of the embedding space as 50.

To measure the prediction performance of our methods and baseline methods fairly, we consider the standard Example-F1 evaluation measurement Du et al. [2017], which computes the F1 score for all the labels of each testing sample and takes the average over those samples. We perform 10-fold cross-validations on each data set and report the mean and standard error of the Example-F1 measure.

### 5.4.2 Prediction Performance

This subsection studies the prediction performance of various methods on mediamill, cal500, corel5k, Eur-Lex (dc) and Eur-Lex (ed) data sets. In our experiment, we cannot get the results of LP and Homer on Eur-Lex (ed) data set within one week. The Example-F1 results of various methods on all data sets are listed in Table 5.3. From this table, we observe that:

Table 5.3: The Example-F1 Results of HCT, SFCT and other baselines on the various data sets (mean ± standard deviation). The best results are in bold. Numbers in square brackets indicate the rank. ”-” indicates that we can not get the results within one week.

| Data Set | BR | LP | Homer | FastXML | SLEEC | HCT | SFCT |
|---|---|---|---|---|---|---|---|
| mediamill | 0.414±0.00[3] | 0.394±0.01[7] | 0.411±0.00[4] | 0.408±0.00[6] | 0.411±0.00[4] | 0.422±0.01[2] | **0.462**±0.00[1] |
| cal500 | 0.317±0.02[5] | 0.345±0.02[2] | 0.321±0.01[4] | 0.259±0.01[7] | 0.310±0.03[6] | 0.324±0.02[3] | **0.362**±0.01[1] |
| corel5k | 0.100±0.02[5] | 0.138±0.01[2] | 0.113±0.01[4] | 0.056±0.01[7] | 0.084±0.01[6] | 0.123±0.01[3] | **0.141**±0.01[1] |
| EUR-Lex(dc) | 0.713±0.01[4] | 0.659±0.01[5] | 0.625±0.01[6] | 0.516±0.01[7] | 0.716±0.01[2] | 0.715±0.01[3] | **0.727**±0.01[1] |
| EUR-Lex(ed) | 0.268±0.01[4] | - | - | 0.254±0.01[5] | 0.281±0.01[2] | 0.274±0.01[3] | **0.341**±0.01[1] |
| Ave. Rank | 4.2 | 4.0 | 4.5 | 6.4 | 4.0 | 2.8 | 1.0 |

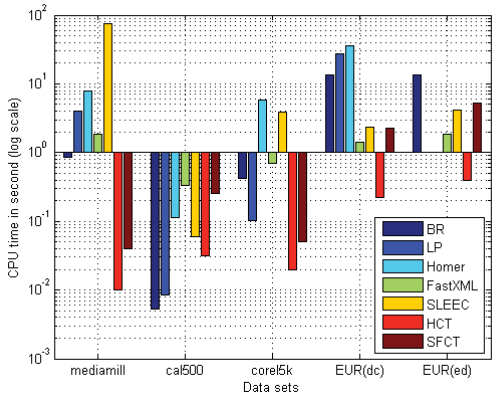


Figure 5.4: Testing time of HCT, SFCT and other baseline methods on all data sets (EUR-Lex is abbreviated to EUR).

- Tree-based methods, such as FastXML, generally underperforms on all data sets.
- SLEEC, which is one of the most advanced embedding method, generally underperforms on small data sets, while it obtains competitive results on medium-sized data sets, such as Eur-Lex (dc) and Eur-Lex (ed).
- HCT and SFCT are two most successful methods, which significantly outperform tree-based algorithms, such as Homer and FastXML.

### 5.4.3 Testing Time

Figure 5.4 shows the testing time of various methods spent on all testing instances per data set. From Figure 5.4, we observe that:

- SLEEC is slower than other baselines on all data sets.

- Tree-based method, such as FastXML, is faster than BR, LP and SLEEC on medium-sized data sets.
- HCT and SFCT are faster than other baselines on all data sets. For example, HCT obtains around sixty times speedup over BR and LP, and about ten times speedup over FastXML on medium-sized data sets, such as Eur-Lex (dc). The results verify our testing time complexity analysis.

## 5.5 Summary of This Chapter

Given the fact that the computational complexity issue prevents many multi-label prediction algorithms from being practical, some tree-based algorithms, like Homer and FastXML, are proposed to minimize the number of predictions. However, they are neither accurate nor efficient enough. To address this issue, we observe three important phenomena of real-world multi-label prediction sets: labels follow the distribution of the Power Law; few labels typically exist for each instance; and very few label powersets have high frequency. Motivated by these three observations, we design Algorithm 2 and 3 to deal with label powerset and multi-label predictions using the bottom-up and top-down manners, respectively. Our theoretical analysis and extensive experiments verify that our algorithms can predict as well as possible without performing unnecessary predictions.

# Chapter 6

# Conclusion and Future Work

In this chapter, we first conclude the entire thesis, and then elaborate the possible trends for future research.

## 6.1 Conclusion

During the past decade, multi-label learning has become a popular machine learning paradigm and significantly attracted the attention of researchers as a result of its wide range of applications, such as automatic image annotation Boutell et al. [2004], document classification Zhang and Zhou [2006], gene function prediction Barutcuoglu et al. [2006], automatic video annotation Qi et al. [2008] and mobile medical recommendations Guo et al. [2016]. State-of-the-art studies have shown that methods of multi-label learning which explicitly capture label dependency will usually achieve better prediction performance, and modeling the label dependency is one of the major challenges in multi-label learning tasks.

Due to its simplicity and promising experimental results, the CC model is one of the most popular methods for capturing label dependency. However, CC suffers from three important problems:

1. Does the label order affect the performance of CC?
2. Is there any globally optimal classifier chain which can achieve the optimal prediction performance for CC?
3. If yes, how can the globally optimal classifier chain be found?

To answer the above questions, Chapter 3 first generalizes the CC model over a random label order. We then present a theoretical analysis of the generalization error for the proposed generalized model. Our results show that the upper bound of

the generalization error depends on the sum of reciprocal of square of the margin over the labels. Thus, we can answer the second question: the globally optimal CC exists only when the minimization of the upper bound is achieved over this CC. To find the globally optimal CC, we can search over $q!$ different label orders, which is computationally infeasible for a large $q$. This thesis first proposes the *dynamic programming based classifier chain* (CC-DP) algorithm to simplify the search algorithm, which requires $\mathcal{O}(q^3nd)$ time complexity. To speed up the training process, a *greedy classifier chain* (CC-Greedy) algorithm is proposed to find a locally optimal CC, where the time complexity of the CC-Greedy algorithm is $\mathcal{O}(q^2nd)$. Furthermore, we propose Tree-DP and Tree-Greedy algorithms to further speed up CC-DP and CC-Greedy, respectively, which scale linearly with $q$. Comprehensive experiments verify our theoretical studies and the effectiveness of proposed algorithms.

Another important branch of methods for capturing label dependency is encoding-decoding paradigm. Based on structural SVMs, MMOC has become one of the most representative encoding-decoding methods and shown promising results for multi-label classification. Unfortunately, MMOC suffers from two major limitations:

1. Inconsistent performance: McAllester [2006] has already proved that structural SVMs fail to converge on the optimal decoder even with infinite training data.

2. Prohibitive computational cost: the training of MMOC involves a complex quadratic programming (QP) problem over the combinatorial space, and its computational cost on the data sets with many labels is prohibitive.

To avoid the inconsistent performance of MMOC, Chapter 4 presents a novel large margin metric learning paradigm for multi-label classification. Our theoretical analysis shows that our proposed model converges to the optimal solutions, and also reduces the generalization error for multi-label classification. To overcome the prohibitive computational cost, we first project both the input and output to the same embedding space, in which the input and output can be compared. A large margin formulation with $k$ nearest neighbor constraints is proposed to learn the embedding space. After transforming our optimization problem to a semidefinite programming problem, accelerated proximal gradient (APG) method is adapted to solve the reduced problem. To avoid the expensive decoding step, we select $k$ nearest neighbors from the training set for each testing instance in the embedding space and make a rapid prediction based on the labels of those $k$ nearest neighbors. Extensive experiments on a number of real-world multi-label data sets corroborate our theoretical results, and demonstrate that our method outperforms state-of-the-art approaches for accuracy. For scalability, our method is almost 10 times faster than CCA and MMOC in terms of both the training and testing times, and is comparable to $k$NN and ML-$k$NN in terms of testing time.

The prediction of most multi-label learning methods either scales linearly with the number of labels or involves an expensive decoding process, which usually requires

solving a combinatorial optimization. Such approaches become unacceptable when tackling thousands of labels, and are impractical for real-world applications, such as document annotation. It is imperative to design an efficient, yet accurate multi-label learning algorithm with the minimum number of predictions. To solve this problem, Chapter 5 first observes three important phenomena of real-world data sets from a variety of application domains: many labels have very low label frequency; each instance has few labels; and very few label powersets have high frequency. To reduce the number of predictions, based on the properties of multi-label learning data sets, we build a coding tree framework to deal with multi-label classification problems and the transformed multi-class classification task in top-down and bottom-up approaches, respectively.

Our goal is to minimize the number of classifiers for binary decisions to enable rapid label powerset prediction for a number of testing instances. Recall that, to minimize the expected codeword length of information, Huffman coding uses shorter codewords to encode more frequent symbols and longer codewords to encode less frequent symbols. Therefore, the idea of Huffman coding can be easily adapted to deal with the label powerset prediction problems. For multi-label prediction problems, the hypothesis space is exponential and we can not observe some combinations of labels in the training data. Thus Huffman coding can not be used to tackle the multi-label prediction problem. To reduce the number of predictions, we develop a tree-based strategy, which is analogous to the Shannon-Fano coding strategy. We first train an accurate node classifier to predict the frequent label that have more examples. Then, we can leverage the predictions of such classifier to divide the remaining labels into two sets whose cardinality are roughly comparable. As long as any sets with more than one member remain, the same process is repeated on those sets. Our algorithm makes predictions for the ordered labels which accord to the arrangement with the label frequency ranging from high to low. The second property shows that each instance has very few labels, based on our algorithm, so only a few predictions will be made for most testing instances. Lastly, comprehensive experiments verify that our methods achieve comparable or better performance without performing unnecessary predictions.

We summarize our conclusions in Figure 6.1.

## 6.2 Future Work

Although the methods and algorithms proposed in this thesis have achieved encouraging results to some extent, some issues still remain open and should be further investigated:

- **An Easy-to-hard Learning Paradigm.** Curriculum learning [Bengio et al.,

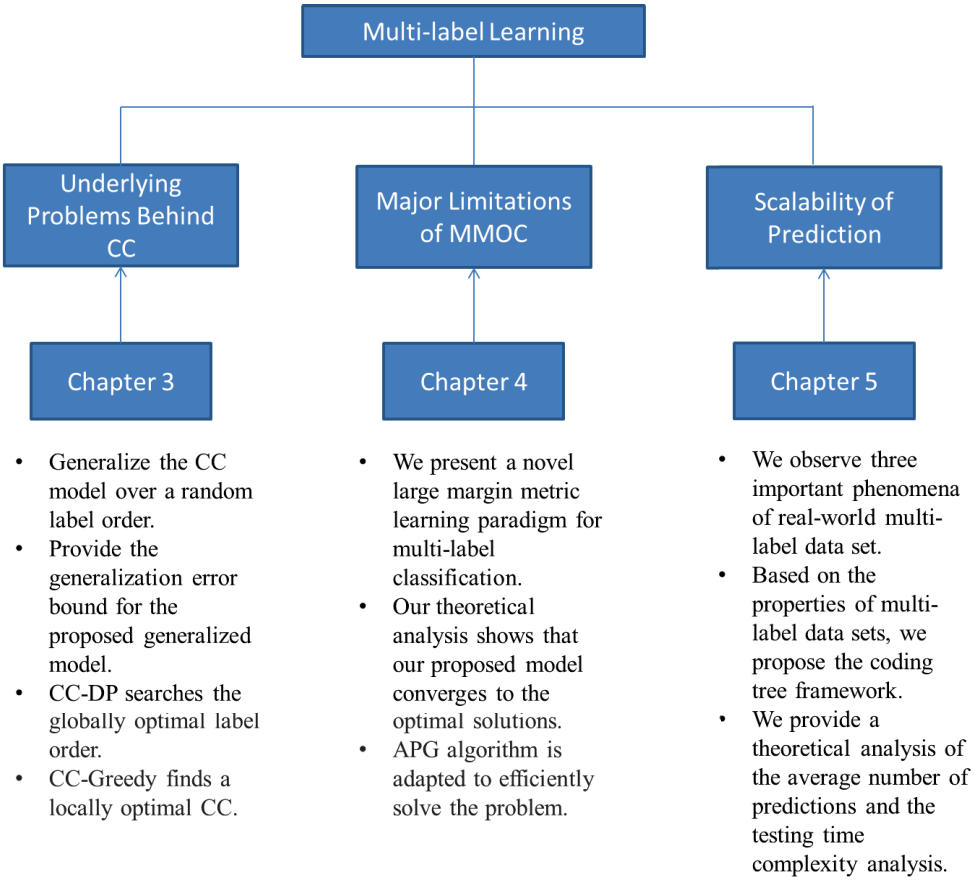


Figure 6.1: The conclusions of this thesis.

2009] can be seen as a sequence of training criteria. Each training criterion in the sequence is associated with a different set of weights on the training examples, or more generally, on a reweighting of the training distribution. Initially, the weights favor easier examples that can be learned most easily. The next training criterion involves a slight change in the weighting of examples that increases the probability of sampling slightly more difficult examples. Overall, curriculum learning aims to find easier examples. However, up to now, curriculum learning has not defined what easy examples meant, or equivalently, how to sort examples into a sequence that illustrates the simpler concepts first. Inspired by curriculum learning, based on our theoretical analysis of CC, we are able to propose an easy-to-hard learning paradigm for multiclass and multi-label classification to automatically identify easy and hard classes or labels and then utilize the predictions from simpler classes or labels to help solve harder classes or labels. Multi-task learning [Pan et al., 2017; Zhou et al., 2012] involves

multiple tasks with different sets of instances. So multi-task learning is different from multiclass and multi-label classification. In the future, we will exploit the proposed easy-to-hard learning paradigm for multi-tasks learning problems.

- **Metric Learning for Multi-output Tasks.** Multi-output learning Álvarez et al. [2012]; Xioufis et al. [2012] aims to predict multiple outputs for an input, where the output values are characterized by diverse data types, such as binary, nominal, ordinal and real-valued variables. This occurs in a number of applications. For example, in document classification Schapire and Singer [2000a], the output for a document may cover a range of topics, such as *News*, *Finance* and *Sport*. The input space is usually composed of variables related to physical properties for computer emulation Fricker et al. [2011], such as Tail tip mass or Wing tip mass, while the corresponding output consists of pairs of masses and stiffness for several structural modes of vibration for an aircraft model. In sensor network analysis Osborne et al. [2008], sensors, which are located close to one another to achieve similar readings, are used to output different environmental variables such as rain or shine, temperature, wind speed, tide height, and so on. It is necessary to consider the trends and correlations observed in previous data to predict the multiple outputs of a sensor that is temporarily unavailable and the future value of multiple environmental parameters. In concept-based information retrieval Egozi et al. [2011], given a user query (e.g., *Steve Jobs*, containing a set of concepts such as STEVE JOBS, MACINTOSH, HISTORY OF APPLE INC.), the system outputs a ranked list of documents ordered by their relevance to the query, and each retrieved document is expected to contain a set of semantic concepts that are most similar to the concepts in the user query. Such applications therefore intrinsically deal with multiple outputs and can be solved by multi-output learning methods Xioufis et al. [2012]; Zhang and Schneider [2011, 2012]. In the future, we will exploit the proposed large margin metric learning paradigm for multi-output tasks with nominal and ordinal values.

- **Coding Tree.** Huffman coding [Huffman, 1952] is one of the most widespread bottom-up encoding algorithm for data compression. Huffman coding uses a specific method for choosing the representation for each symbol, resulting in a prefix code, that is, the bit string representing some particular symbol is never a prefix of the bit string representing any other symbol. Given a set of symbols and their weights (usually proportional to probabilities), Huffman coding aims to find a set of prefix codewords with minimum expected codeword length. Based on the frequency of occurrence of each symbol, the principle of Huffman coding is to use a lower number of bits to encode the symbol that occurs more frequently. Shannon-Fano coding [Shannon, 1948] is a top-down encoding technique for

constructing a prefix code based on a set of symbols and their weights. Shannon-Fano coding arranges the symbols in order from biggest weight to smallest weight, and then divides them into two sets whose total weights are roughly comparable. Then, we encode symbols in the first set as zero and symbols in the second set as one. As long as any sets with more than one member remain, the same process is repeated on these sets. To the best of our knowledge, we are the first to use Huffman coding and Shannon-Fano coding in multi-label domain. In the future, we will explore other advanced coding techniques and theories for multi-label learning.

- **Random Projections.** Embedding approaches have become one of the most pervasive techniques for multi-label classification. However, the training process of embedding methods usually involves a complex quadratic or semidefinite programming problem, or the model may even involve an NP-hard problem. Thus, such methods will not be the first choice for a number of practical applications. To avoid complex training process, in the future, we will explore some advanced random projection methods with provable guarantee for multi-label learning.

- **Cross-View Learning.** Encoding-decoding methods have shown promising performance in multi-label prediction, as they are able to discover the dependency of labels. However, most of them ignore the correlations between the input and output, such that their learned embeddings are not well aligned, which leads to degradation in prediction performance. In the future, we will explore the modelling of multi-label learning, from the perspective of cross-view learning [Wu et al., 2014], that explores the correlations between the input and output. Specifically, we will learn the semantic common subspace and the view-specific mappings within one framework. The semantic similarity structure among the embeddings is further preserved, ensuring that close embeddings share similar labels. Furthermore, we will try advanced learning-based hashing technique [Shen et al., 2017] to generate compact binary representations to improve the prediction efficiency.

# Appendix A

# Appendix

## A.1 Covering Numbers

**Definition 5** (Covering Numbers, Shawe-Taylor et al. [1998])**.** *Let $(\mathcal{X}, d)$ be a metric space, $A$ be a subset of $\mathcal{X}$ and $\varepsilon > 0$. A set $B \subseteq X$ is an $\varepsilon$-cover for $A$, if for every $a \in A$, there exists $b \in B$ such that $d(a, b) < \varepsilon$. The $\varepsilon$-covering number of $A$, $\mathcal{N}(\varepsilon, A, d)$, is the minimal cardinality of an $\varepsilon$-cover for $A$ (if there is no such finite cover then it is defined as $\infty$).*

In Chapter 3, suppose $\mathcal{N}(\epsilon, \mathcal{H}, \mathbf{s})$ be the $\epsilon$-covering number of $\mathcal{H}$ with respect to the $l_\infty$ pseudo-metric measuring the maximum discrepancy on the sample **s**, that is, with respect to the distance $d(f, g) = \max_{1 \leq t \leq m} |f(x_t) - g(x_t)|$, for $f, g \in \mathcal{H}$.

## A.2 Proof of Lemma 2

*Proof.* For each $\mathbf{s}$, let $\bar{h}_{\mathbf{s}}$ be a function for which $|er_D[\bar{h}_{\mathbf{s}}] - er_{\mathbf{s}}[\bar{h}_{\mathbf{s}}]| \geq \epsilon$ if such a function exists, and any fixed function in $\mathcal{H}$ otherwise. Then

$$\begin{aligned}
&P_{\mathbf{s}\bar{\mathbf{s}}}(\sup_{h\in\mathcal{H}}|er_{\bar{\mathbf{s}}}[h] - er_{\mathbf{s}}[h]| \geq \epsilon/2) \\
\geq &P_{\mathbf{s}\bar{\mathbf{s}}}(|er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}] - er_{\mathbf{s}}[\bar{h}_{\mathbf{s}}]| \geq \epsilon/2) \\
\geq &P_{\mathbf{s}\bar{\mathbf{s}}}(\{|er_D[\bar{h}_{\mathbf{s}}] - er_{\mathbf{s}}[\bar{h}_{\mathbf{s}}]| \geq \epsilon\} \bigcap \{|er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}] - er_D[\bar{h}_{\mathbf{s}}]| \leq \epsilon/2\}) \\
= &E_{\mathbf{s}}[[\![(|er_D[\bar{h}_{\mathbf{s}}] - er_{\mathbf{s}}[\bar{h}_{\mathbf{s}}]| \geq \epsilon)]\!] P_{\bar{\mathbf{s}}}(|er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}] - er_D[\bar{h}_{\mathbf{s}}]| \leq \epsilon/2)]
\end{aligned} \tag{A.1}$$

Now the conditional probability inside can be bounded using Chebyshev's inequality:

$$P_{\bar{\mathbf{s}}}(|er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}] - er_D[\bar{h}_{\mathbf{s}}]| \leq \epsilon/2) \geq 1 - \frac{\mathbf{Var}_{\bar{\mathbf{s}}}[er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}]]}{\epsilon^2/4} \tag{A.2}$$

Since $\bar{\mathbf{s}} \sim D^m$ and $er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}]$ is $1/m$ times a Binomial random variable with parameters $(m, er_D[\bar{h}_{\mathbf{s}}])$, we have $\mathbf{Var}_{\bar{\mathbf{s}}}[er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}]] = \frac{er_D[\bar{h}_{\mathbf{s}}](1-er_D[\bar{h}_{\mathbf{s}}])}{m} \leq \frac{1}{4m}$. This gives

$$P_{\bar{\mathbf{s}}}(|er_{\bar{\mathbf{s}}}[\bar{h}_{\mathbf{s}}] - er_D[\bar{h}_{\mathbf{s}}]| \leq \epsilon/2) \geq 1 - \frac{1}{m\epsilon^2} \geq \frac{1}{2} \tag{A.3}$$

whenever $m\epsilon^2 \geq 2$. Thus we get

$$\begin{aligned}
P_{\mathbf{s}\bar{\mathbf{s}}}(\sup_{h\in\mathcal{H}}|er_{\bar{\mathbf{s}}}[h] - er_{\mathbf{s}}[h]| \geq \epsilon/2) \geq &\frac{1}{2}P_{\mathbf{s}}(|er_D[\bar{h}_{\mathbf{s}}] - er_{\mathbf{s}}[\bar{h}_{\mathbf{s}}]| \geq \epsilon) \\
= &\frac{1}{2}P_{\mathbf{s}}(\sup_{h\in\mathcal{H}}|er_D[h] - er_{\mathbf{s}}[h]| \geq \epsilon)
\end{aligned} \tag{A.4}$$

where the last step of Eq. (A.4) is by definition of $\bar{h}_{\mathbf{s}}$. □

## A.3 Proof of Theorem 3

*Proof.* We proof the theorem using the mathematical induction. For $i \in \{1, \cdots, q\}$,

**Case 1:** $V(i,1) = \frac{1}{(\gamma_i^1)^2}$, where $\gamma_i^1$ is the margin for label $\lambda_i$, without augmented input and $M_i^1 = \{\lambda_i\}$.

**Case 2:** $V(i,2) = \min_{j \neq i, \lambda_i \notin M_j^1} \{(\frac{1}{(\gamma_i^2)^2} + V(j,1)\}$, where $\gamma_i^2$ is the margin for label $\lambda_i$, with $M_j^1$ as the augmented input. As in case 1, we already calculated $V(i,1)$, so we can easily find the solution of $V(i,2)$. Assume $V(j,1)$ is the optimal value for computing $V(i,2)$, then we can get $M_i^2 = M_j^1 \cup \{\lambda_i\}$.

**Case 3:** Assume $V(i,k-1), k \leq q$ is the optimal $Q'$ over a subset of $\mathcal{M}$ with the length of $k-1$, where the label order is ending by label $\lambda_i$ and $M_i^{k-1}$ denote the corresponding label set for $V(i,k-1)$.

**Case 4:** $V(i,k) = \min_{j \neq i, \lambda_i \notin M_j^{k-1}} \{\frac{1}{(\gamma_i^k)^2} + V(j,k-1)\}$, where $\gamma_i^k$ is the margin for label $\lambda_i$, with $M_j^{k-1}$ as the augmented input. Based on the assumption in case 3, we can obtain $V(i,k), i \in \{1, \cdots, q\}$. Thus, we can find the optimal $Q'$ over $\mathcal{M}$ by using CC-DP algorithm. □

## A.4 CC-Greedy algorithm

To speed up the CC-DP algorithm, we propose a CC-Greedy algorithm to find a locally optimal CC.

Based on the training instances, we select the label from $\{\lambda_1, \lambda_2, \cdots, \lambda_q\}$ as the first label, the maximum margin can be achieved over this label, without augmented input. The first label is denoted by $\zeta_1$. Then, we select the label from the remainder as the second label, if the maximum margin can be achieved over this label with $\zeta_1$ as the augmented input. We continue in this way until the last label is selected. Finally, this algorithm will converge to the locally optimal CC, which requires at most $\mathcal{O}(q^2nd)$ time complexity. This section presents the details of CC-Greedy algorithm:

**Algorithm 4** Greedy algorithm for locally optimal CC (CC-Greedy)

**Input:** training data $\{\mathbf{x}_t, \mathbf{y}_t\}_{t=1}^n$ with size $n$ and label set $\{\lambda_1, \lambda_2, \cdots, \lambda_q\}$.
Set $\mathcal{M} = \{\lambda_1, \lambda_2, \cdots, \lambda_q\}$.
**for** $\lambda_j \in \mathcal{M}$ **do**
  Calculate $[\mathbf{w}_j, b] = SVM(\{\mathbf{x}_t\}_{t=1}^n, \{\mathbf{y}_t(\lambda_j)\}_{t=1}^n)$.
  Calculate $\gamma_j^1$.
**end for**
Calculate $\nu = \arg_{\lambda_j \in \mathcal{M}} \min \frac{1}{(\gamma_j^1)^2}$.
Set $\mathcal{M} = \mathcal{M} - \{\lambda_\nu\}$
Set $C[1] = \lambda_\nu$.
**for** $s = 2$ **to** $q$ **do**
  **for** $\lambda_k \in \mathcal{M}$ **do**
    Calculate $[\mathbf{w}_k, b] = SVM(\{\mathbf{x}_t, \mathbf{y}_t(C[1]), \cdots, \mathbf{y}_t(C[s-1])\}_{t=1}^n, \{\mathbf{y}_t(\lambda_k)\}_{t=1}^n)$.
    Calculate $\gamma_k^s$.
  **end for**
  Calculate $\nu = \arg_{\lambda_k \in \mathcal{M}} \min \frac{1}{(\gamma_k^s)^2}$ .
  Set $\mathcal{M} = \mathcal{M} - \{\lambda_\nu\}$.
  Set $C[s] = \lambda_\nu$.
**end for**
Output this locally optimal CC.

## A.5 Proof of Theorem 6

*Proof.*

$$
\begin{aligned}
&E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(\sum_{i=1}^{q} P(y_i \neq f^D_{1nn_i}(x))\Big) \\
&= \sum_{i=1}^{q} E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(P(y_i \neq f^D_{1nn_i}(x))\Big)
\end{aligned}
\tag{A.5}
$$

Now, we focus on $P(y_i \neq f^D_{1nn_i}(x))$ for the $i$-th label. Given $x, x' \in \mathcal{X}$, due to $\nu^i(\cdot)$ is Lipschitz with constant $L$ with respect to the sup-norm, we have $||\nu^i(x) - \nu^i(x')||_\infty = \sup_{j\in\{0,1\}} |\nu^i_j(x) - \nu^i_j(x')| \leq L d_{pro}(x, x')$ and

$$
\begin{aligned}
&P(y_i \neq y'_i|x, x') \\
&= P(y_i = 1|x)P(y'_i = 0|x') + P(y_i = 0|x)P(y'_i = 1|x') \\
&= \sum_{j\in\{0,1\}} \nu^i_j(x)(1 - \nu^i_j(x')) \\
&\leq \sum_{j\in\{0,1\}} \nu^i_j(x)(1 - \nu^i_j(x) + L d_{pro}(x, x')) \\
&= \sum_{j\in\{0,1\}} \nu^i_j(x)(1 - \nu^i_j(x)) + L d_{pro}(x, x')
\end{aligned}
\tag{A.6}
$$

As $d_{pro}(x, x') = ||V^T M^T x - V^T M^T x'||_2 \leq ||M||_F ||V||_F ||x - x'||_2 \leq ||M||_F \sqrt{trace(Q)} d(x, x')$, we get

$$
\begin{aligned}
&P(y_i \neq y'_i|x, x') \\
&\leq \sum_{j\in\{0,1\}} \nu^i_j(x)(1 - \nu^i_j(x)) + L||M||_F \sqrt{trace(Q)} d(x, x')
\end{aligned}
\tag{A.7}
$$

Assume $(x', y')$ is the nearest neighbor of $(x, y)$ in $D$: $(x', y') = \arg\min_{(x^{(i)},y^{(i)})\in D} d_{pro}(x, x^{(i)})$.

Then, we have $E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(P(y_i \neq f^D_{1nn_i}(x))\Big) = E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(P(y_i \neq y'_i)\Big)$. Following Eq.(A.6), we get:

$$\begin{aligned}
&E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(P(y_i \neq f^D_{1nn_i}(x))\Big) \\
&\leq E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(\sum_{j\in\{0,1\}} \nu^i_j(x)(1-\nu^i_j(x))\Big) \\
&+ L||M||_F\sqrt{trace(Q)}E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(d(x,x')\Big)
\end{aligned} \tag{A.8}$$

Assume the solution of $\arg\max_{j\in\{0,1\}} \nu^i_j(x)$ is 1. The first term of right side of Eq.(A.8) does not depend on $D$. Thus

$$\begin{aligned}
&E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(\sum_{j\in\{0,1\}} \nu^i_j(x)(1-\nu^i_j(x))\Big) \\
&= E_{(x,y)\sim\mathcal{D}}\Big(\nu^i_1(x)(1-\nu^i_1(x)) + \nu^i_0(x)(1-\nu^i_0(x))\Big) \\
&\leq E_{(x,y)\sim\mathcal{D}}(1-\nu^i_1(x)) + E_{(x,y)\sim\mathcal{D}}(\nu^i_0(x)) \\
&= 2E_{(x,y)\sim\mathcal{D}}(1-\nu^i_1(x)) = 2P(b^*_i(x) \neq y_i)
\end{aligned} \tag{A.9}$$

Then, we start to bound the second term of right side of Eq.(A.8). Let $\{C_1, \cdots, C_N\}$ be an $\varepsilon$-cover of $\mathcal{X}$ of cardinality $N = \mathcal{N}(\varepsilon, \mathcal{X}, d)$. Given a sampling $D$, for $x \in C_i$ such that $D \cap C_i \neq \emptyset$, we have $d(x, x') \leq \varepsilon$. While for $x \in C_i$ such that $D \cap C_i = \emptyset$, we have $d(x, x') \leq diam(\mathcal{X}) = 1$. The expression $[\![D \cap C_i \neq \emptyset]\!]$ evaluates to 1 if

$D \cap C_i \neq \emptyset$ is true and to 0 otherwise. Thus, we have

$$
\begin{aligned}
&E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(d(x,x')\Big) \\
&\leq E_{D\sim\mathcal{D}^n}\Big(\sum_{j=1}^{N} P(C_j)(\varepsilon[\![D\cap C_j\neq\emptyset]\!]+[\![D\cap C_j=\emptyset]\!])\Big) \\
&\leq \sum_{j=1}^{N} P(C_j)\Big(\varepsilon E_{D\sim\mathcal{D}^n}([\![D\cap C_j\neq\emptyset]\!])+E_{D\sim\mathcal{D}^n}([\![D\cap C_j=\emptyset]\!])\Big)
\end{aligned}
\tag{A.10}
$$

Since $P(C_j)E_{D\sim\mathcal{D}^n}([\![D\cap C_j=\emptyset]\!]) = P(C_j)(1-P(C_i))^n \leq 1/en$, where $e$ is the exponent constant. This result, Eq.(A.10) and Theorem 5 in Chapter 4 imply that

$$
\begin{aligned}
E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(d(x,x')\Big) &\leq \Big(\varepsilon+\frac{N}{en}\Big) \\
&\leq \Big(\varepsilon+\frac{1}{en}(\frac{2}{\varepsilon})^{\mathbb{D}}\Big)
\end{aligned}
\tag{A.11}
$$

By setting $\varepsilon = 2n^{-\frac{1}{\mathbb{D}+1}}$, we get

$$
E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(d(x,x')\Big) \leq \frac{3}{n^{1/(\mathbb{D}+1)}} \tag{A.12}
$$

Eq.(A.5), Eq.(A.8), Eq.(A.9) and Eq.(A.12) imply the result. □

## A.6 Proof of Lemma 3

*Proof.* We adapt the proof for Theorem 19.5 in Shalev-Shwartz and Ben-David [2014]. We first focus on $E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(P(y_i \neq f^D_{knn_i}(x))\Big)$ for the $i$-th label. For each $x \in \mathcal{X}$ and training set $D = \{(x^{(1)},y^{(1)}),\cdots,(x^{(n)},y^{(n)})\}$, let $\pi_1(x),\cdots,\pi_n(x)$ be a reordering of $\{1,\cdots,n)\}$ according to their distance to $x$, $d_{pro}$. That is, for all $j<m$, $d_{pro}(x,x_{\pi_j(x)}) \leq d_{pro}(x,x_{\pi_{j+1}(x)})$. Let $\{C_1,\cdots,C_N\}$ be an $\varepsilon$-cover of $\mathcal{X}$ of cardinality

$N = \mathcal{N}(\varepsilon, \mathcal{X}, d)$. Eq.(19.3) in Shalev-Shwartz and Ben-David [2014] implies that

$$
\begin{aligned}
&E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(P(y_i \neq f^D_{knn_i}(x))\Big) \\
&\leq E_{D\sim\mathcal{D}^n}\Big(\sum_{j:|C_j\cap D|<k} P(C_j)\Big) \\
&+ \max_z P_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(y_i \neq f^D_{knn_i}(x)| \\
&\qquad\qquad \forall z \in [k], d_{pro}(x, x_{\pi_z(x)}) \leq ||M||_F\sqrt{trace(Q)}\varepsilon\Big)
\end{aligned}
\tag{A.13}
$$

Following the proof of Theorem 19.5 in Shalev-Shwartz and Ben-David [2014], the first term of right side of Eq.(A.13) is bounded by $\frac{2Nk}{en}$. The second term of right side of Eq.(A.13) is bounded by $(1+\sqrt{8/k})P(b^*_i(x) \neq y_i) + 3L||M||_F\sqrt{trace(Q)}\varepsilon$. By setting $\varepsilon = 2n^{-\frac{1}{\mathbb{D}+1}}$ and combining Theorem 5 in Chapter 4, we get

$$
\begin{aligned}
&E_{D\sim\mathcal{D}^n,(x,y)\sim\mathcal{D}}\Big(P(y_i \neq f^D_{knn_i}(x))\Big) \\
&\leq (1+\sqrt{8/k})P(b^*_i(x) \neq y_i) + \frac{6L||M||_F\sqrt{trace(Q)}+k}{n^{1/(\mathbb{D}+1)}}
\end{aligned}
\tag{A.14}
$$

Applying Eq.(A.14) for each label and take the sum to derive the result. □

# References


Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the International World Wide Web Conference*, pages 13–24, May 2013. 21, 59

Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for vector-valued functions: A review. *Foundation and Trends in Machine Learning*, 4(3): 195–266, 2012. 75

Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NIPS*, pages 2654–2662, 2014. 41

A.-L. Barabási, R. Albert, H. Jeong, and G. Bianconi. Power-law distribution of the world wide web. *Science*, 287(54561):2115, 2000. 61

Peter L. Bartlett and John Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers. In *Advances in Kernel Methods - Support Vector Learning*, pages 43–54. MIT Press, 1998. 30

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, 2006. 1, 5, 71

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, 2(1):183–202, 2009. 7, 46, 48, 49

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, pages 41–48, 2009. 73

Kristin P. Bennett, Nello Cristianini, John Shawe-Taylor, and Donghui Wu. Enlarging the margins in perceptron decision trees. *Machine Learning*, 41(3):295–313, 2000. 27

Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. In *NIPS*, pages 730–738, 2015. 68

Matthew R. Boutell, Jiebo Luo, Xipeng Shen, and Christopher M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004. 1, 5, 10, 19, 53, 71

Serhat Selcuk Bucak, Rong Jin, and Anil K. Jain. Multi-label multiple kernel learning by stochastic approximation: Application to visual object recognition. In *NIPS*, pages 325–333, 2010. 5

Weizhu Chen, Jun Yan, Benyu Zhang, Zheng Chen, and Qiang Yang. Document transformation for multi-label feature selection in text categorization. In *ICDM*, pages 451–456, 2007. 10

Yao-Nan Chen and Hsuan-Tien Lin. Feature-aware label space dimension reduction for multi-label classification. In *NIPS*, pages 1538–1546, 2012a. 54, 55

Yao-Nan Chen and Hsuan-Tien Lin. Feature-aware label space dimension reduction for multi-label classification. In *NIPS*, pages 1538–1546, 2012b. 4

Patrick Marques Ciarelli, Elias Oliveira, and Evandro O. T. Salles. Multi-label incremental learning applied to web page categorization. *Neural Computing and Applications*, 24(6):1403–1419, 2014. 5

Moustapha Cissé, Maruan Al-Shedivat, and Samy Bengio. ADIOS: architectures deep in output space. In *ICML*, pages 2770–2779, 2016. 38, 40

Amanda Clare and Ross D. King. Knowledge discovery in multi-label phenotype data. In *PKDD*, pages 42–53, 2001. 13

Francesco De Comité, Rémi Gilleron, and Marc Tommasi. Learning multi-label alternating decision trees from texts and data. In *MLDM*, pages 35–49, 2003. 13

Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006. 17

Krzysztof Dembczynski, Weiwei Cheng, and Eyke Hüllermeier. Bayes optimal multilabel classification via probabilistic classifier chains. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning*, pages 279–286, Haifa, Israel, 2010. Omnipress. 4, 15, 25, 35

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 6, 59

Jia Deng, Alexander C. Berg, Kai Li, and Li Fei-Fei. What does classifying more than 10,000 image categories tell us? In *ECCV*, 2010. 44

Bo Du, Mengfei Zhang, Lefei Zhang, Ruimin Hu, and Dacheng Tao. PLTD: patch-based low-rank tensor decomposition for hyperspectral images. *IEEE Trans. Multimedia*, 19(1):67–79, 2017. 68

Lei Duan, Satoshi Oyama, Masahito Kurihara, and Haruhiko Sato. Crowdsourced semantic matching of multi-label annotations. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 3483–3489, 2015. 20

Susan T. Dumais, John C. Platt, David Hecherman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM*, pages 148–155, 1998. 13

Pinar Duygulu, Kobus Barnard, João F. G. de Freitas, and David A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *7th European Conference on Computer Vision*, pages 97–112, 2002. 53

Ofer Egozi, Shaul Markovitch, and Evgeniy Gabrilovich. Concept-based information retrieval using explicit semantic analysis. *ACM Transactions on Information Systems*, 29:8, 2011. 75

André Elisseeff and Jason Weston. A kernel method for multi-labelled classification. In *NIPS*, pages 681–687, 2001. 13

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9:1871–1874, 2008. 35, 54, 68

Thomas Finley and Thorsten Joachims. Training structural SVMs when exact inference is intractable. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, pages 304–311, 2008. 47

Thomas E. Fricker, Jeremy E. Oakley, Neil D. Sims, and Keith Worden. Probabilistic uncertainty analysis of an FRF of a structure using a Gaussian process emulator. *Mechanical Systems and Signal Processing*, 25(8):2962–2975, 2011. 75

Nadia Ghamrawi and Andrew McCallum. Collective multi-label classification. In *CIKM*, pages 195–200, 2005. 13, 23

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS*, pages 315–323, 2011. 40

Chen Gong, Dacheng Tao, Wei Liu, Liu Liu, and Jie Yang. Label propagation via teaching-to-learn and learning-to-teach. *IEEE Trans. Neural Netw. Learning Syst.*, 28(6):1452–1465, 2017. 4

Leo Grady and Gareth Funka-Lea. Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials. In *Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis*, pages 230–245, 2004. 5

Li Guo, Bo Jin, Ruiyun Yu, Cuili Yao, Chonglin Sun, and Degen Huang. Multi-label classification methods for green computing and application for mobile medical recommendations. *IEEE Access*, 4:3201–3209, 2016. 3, 5, 71

Yuhong Guo and Suicheng Gu. Multi-label classification using conditional dependency networks. In Toby Walsh, editor, *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, pages 1300–1305, Barcelona, Catalonia, Spain, 2011. AAAI Press. 4, 14, 25

Guangjie Han, Yuhui Dong, Hui Guo, Lei Shu, and Dapeng Wu. Cross-layer optimized routing in wireless sensor networks with duty cycle and energy harvesting. *Wireless Communications and Mobile Computing*, 15(16):1957–1981, 2015. 3

Guangjie Han, Li Liu, Jinfang Jiang, Lei Shu, and Gerhard P. Hancke. Analysis of energy-efficient connected target coverage algorithms for industrial wireless sensor networks. *IEEE Trans. Industrial Informatics*, 13(1):135–143, 2017. 3

Jianjun He, Hong Gu, and Zhelong Wang. Multi-instance multi-label learning based on gaussian process with application to visual mobile robot navigation. *Inf. Sci.*, 190:162–177, 2012. 5

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 40, 41

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. 41

Daniel Hsu, Sham Kakade, John Langford, and Tong Zhang. Multi-label prediction via compressed sensing. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, pages 772–780. Curran Associates, Inc., 2009. 17, 36, 44, 54

Shaoli Huang, Zhe Xu, Dacheng Tao, and Ya Zhang. Part-stacked CNN for fine-grained visual categorization. In *CVPR*, pages 1173–1182, 2016. 1

Sheng-Jun Huang and Zhi-Hua Zhou. Multi-label learning by exploiting label correlations locally. In *AAAI*, 2012. 4, 14, 15, 25

David A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40(9):1098–1101, 1952. 21, 22, 75

Eyke Hüllermeier, Johannes Fürnkranz, Weiwei Cheng, and Klaus Brinker. Label ranking by learning pairwise preferences. *Artif. Intell.*, 172(16-17):1897–1916, 2008. 12

Feng Kang, Rong Jin, and Rahul Sukthankar. Correlated label propagation with application to multi-label learning. In *CVPR*, pages 1719–1726, 2006. 4, 14, 15, 25

Michael J. Kearns and Robert E. Schapire. Efficient distribution-free learning of probabilistic concepts. In *Proceedings of the 31st Symposium on the Foundations of Computer Science*, pages 382–391, Los Alamitos, CA, 1990. IEEE Computer Society Press. 27

Aryeh Kontorovich and Roi Weiss. Maximum margin multiclass nearest neighbors. In *ICML*, pages 892–900, 2014. 51

Robert Krauthgamer and James R. Lee. Navigating nets: Simple algorithms for proximity search. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 798–807, 2004. 51

Brian Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013. 7, 45, 48, 49

James T. Kwok and Ivor W. Tsang. Learning with idealized kernels. In *ICML*, pages 400–407, 2003. 44

Weiwei Liu and Ivor W. Tsang. Large margin metric learning for multi-label prediction. In *AAAI*, pages 2800–2806, 2015a. 4, 7, 25

Weiwei Liu and Ivor W. Tsang. On the optimality of classifier chain for multi-label classification. In *NIPS*, pages 712–720, 2015b. 6

Weiwei Liu and Ivor W. Tsang. Sparse perceptron decision tree for millions of dimensions. In *AAAI*, pages 1881–1887, 2016. 59

Weiwei Liu and Ivor W. Tsang. Making decision trees feasible in ultrahigh feature and label dimensions. *Journal of Machine Learning Research*, 18:1–36, 2017. 7

Weiwei Liu, Zhi-Hong Deng, Longbing Cao, Xiaoran Xu, He Liu, and Xiuwen Gong. Mining top K spread sources for a specific topic and a given node. *IEEE Trans. Cybernetics*, 45(11):2472–2483, 2015a. 2

Weiwei Liu, Zhi-Hong Deng, Xiuwen Gong, Frank Jiang, and Ivor W. Tsang. Effectively predicting whether and when a topic will become prevalent in a social network. In *AAAI*, pages 210–216, 2015b. 2

Xinwang Liu, Lei Wang, Jianping Yin, Yong Dou, and Jian Zhang. Absent multiple kernel learning. In *AAAI*, pages 2807–2813, 2015c. 13

Eneldo Loza Mencía and Johannes Fürnkranz. Pairwise learning of multilabel classifications with perceptrons. In *IJCNN*, pages 2899–2906, 2008a. 12

Eneldo Loza Mencía and Johannes Fürnkranz. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In *ECML/PKDD*, pages 50–65, 2008b. 53

Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and Saso Dzeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104, Sep 2012. 21, 59

Qi Mao, Ivor Wai-Hung Tsang, and Shenghua Gao. Objective-guided image annotation. *IEEE Transactions on Image Processing*, 22(4):1585–1597, 2013. 23, 61

David McAllester. Generalization bounds and consistency for structured labeling. In Bakir Gökhan H., Hofmann Thomas, Schölkopf Bernhard, Smola Alexander J., Taskar Ben, and Vishwanathan S. V. N., editors, *Predicting Structured Data*, chapter 11, pages 247–262. MIT Press, 2006. 5, 18, 19, 45, 57, 72

Milind R. Naphade, John R. Smith, Jelena Tesic, Shih-Fu Chang, Winston H. Hsu, Lyndon S. Kennedy, Alexander G. Hauptmann, and Jon Curtis. Large-scale concept ontology for multimedia. *IEEE MultiMedia*, 13(3):86–91, 2006. 2

Daniel Neagu, Shuai Zhang, and Catalin Balescu. A multi-label voting algorithm for neuro-fuzzy classifier ensembles with applications in visual arts data mining. In *Proceedings of the Fifth International Conference on Intelligent Systems Design and Applications*, pages 245–250, 2005. 5

Michael A. Osborne, Stephen J. Roberts, Alex Rogers, Sarvapali D. Ramchurn, and Nicholas R. Jennings. Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes. In *Proceedings of the International Conference on Information Processing in Sensor Networks*, pages 109–120, 2008. 75

Shirui Pan, Jia Wu, Xingquan Zhu, Guodong Long, and Chengqi Zhang. Task sensitive feature exploration and learning for multitask graph classification. *IEEE Trans. Cybernetics*, 47(3):744–758, 2017. 74

Witold Pedrycz. Advances in kernel methods. *Neurocomputing*, 47(1-4):303–304, 2002. 50

Yashoteja Prabhu and Manik Varma. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 263–272, August 2014. 21, 59, 67, 68

Guo-Jun Qi, Xian-Sheng Hua, Yong Rui, Jinhui Tang, Tao Mei, Meng Wang, and Hong-Jiang Zhang. Correlative multi-label video annotation with temporal kernels. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 5(1), 2008. 2, 5, 71

Nagulan Ratnarajah and Anqi Qiu. Multi-label segmentation of white matter structures: Application to neonatal brains. *NeuroImage*, 102:913–922, 2014. 4, 5

Jesse Read. A pruned problem transformation method for multi-label classification. In *New Zealand Computer Science Research Student Conference*, pages 143–150, 2008. 11

Jesse Read, Bernhard Pfahringer, Geoffrey Holmes, and Eibe Frank. Classifier chains for multi-label classification. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, pages 254–269, 2009. 4, 5, 6, 15, 16, 20, 25, 34, 35, 59

Steven Roman. *Coding and Information Theory*. Springer Verlag, New York, 1992. 65

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*, pages 318–362. MIT Press, 1986. 13

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW*, pages 851–860, 2010. 3

Robert E. Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2-3):135–168, 2000a. 1, 23, 75

Robert E. Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000b. 13

Axel Schulz, Loza Mencía Eneldo, and Benedikt Schmidt. A rapid-prototyping framework for extracting small-scale incident-related information in microblogs: Application of multi-label classification on tweets. *Inf. Syst.*, 57:88–110, 2016. 3, 5

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, 2014. 52, 83, 84

Claude Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948. 22, 61, 75

Zheng Shao, Yuya Hirayama, Yoshihiro Yamanishi, and Hiroto Saigo. Mining discriminative patterns from graph data with multiple labels and its application to quantitative structure-activity relationship (QSAR) models. *Journal of Chemical Information and Modeling*, 55(12):2519–2527, 2015. 5

John Shawe-Taylor, Peter L. Bartlett, Robert C. Williamson, and Martin Anthony. Structural risk minimization over data-dependent hierarchies. *TIT*, 44(5):1926–1940, 1998. 27, 28, 77

Xiao-Bo Shen, Weiwei Liu, Ivor W. Tsang, Fumin Shen, and Quan-Sen Sun. Compressed k-means for large-scale clustering. In *AAAI*, pages 2527–2533, 2017. 76

Alessio Signorini, Alberto Maria Segre, and Philip M. Polgreen. The use of twitter to track levels of disease activity and public concern in the u.s. during the influenza a h1n1 pandemic. *PLoS ONE*, 6(5):e19467, 2011. 3

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 40, 41

Cees Snoek, Marcel Worring, Jan-Mark Geusebroek, Dennis Koelma, Frank J. Seinstra, and Arnold W. M. Smeulders. The semantic pathfinder: Using an authoring metaphor for generic multimedia indexing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1678–1689, 2006. 2, 3

Yan Song, Xian-Sheng Hua, Li-Rong Dai, and Meng Wang. Semi-automatic video annotation based on active learning with multiple complementary predictors. In *Multimedia Information Retrieval*, pages 97–104, 2005. 19

Andreas P. Streich and Joachim M. Buhmann. Classification of multi-labeled data: A generative approach. In *ECML/PKDD*, pages 390–405, 2008. 13

Farbound Tai and Hsuan-Tien Lin. Multilabel classification with principal label space transformation. *Neural Computation*, 24(9):2508–2542, 2012. 17, 44, 54, 57

Mingkui Tan, Qinfeng Shi, Anton van den Hengel, Chunhua Shen, Junbin Gao, Fuyuan Hu, and Zhen Zhang. Learning graph structure for multi-label image classification via clique generation. In *CVPR*, pages 4100–4109, 2015. 25

Kim-Chuan Toh and Sangwoon Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. Technical report, National University of Singapore, 2009. 7, 46, 48

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6: 1453–1484, 2005. 5, 18, 19, 45, 47

Grigorios Tsoumakas and Ioannis P. Vlahavas. Random $k$ -labelsets: An ensemble method for multilabel classification. In *ECML*, pages 406–417, 2007a. 11, 23

Grigorios Tsoumakas and Ioannis P. Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *Proceedings of the 18th European Conference on Machine Learning*, pages 406–417, Warsaw, Poland, 2007b. Springer-Verlag. 20

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proc. ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, 2008. 21, 53, 59, 67, 68

Grigorios Tsoumakas, Min-Ling Zhang, and Zhi-Hua Zhou. Tutorial on learning from multi-label data. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2009. 1

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis P. Vlahavas. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer, 2010. 1, 6, 10, 11, 19, 20, 25, 44, 54, 59, 68

Douglas Turnbull, Luke Barrington, David A. Torres, and Gert R. G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):467–476, 2008. 53

Naonori Ueda and Kazumi Saito. Parametric mixture models for multi-labeled text. In *NIPS*, pages 721–728, 2002. 13

Rajasekar Venkatesan, Meng Joo Er, Mihika Dave, Mahardhika Pratama, and Shiqian Wu. A novel online multi-label classifier for high-speed streaming data applications. *CoRR*, abs/1609.00086, 2016. 5

Sarah Vieweg, Amanda Lee Hughes, Kate Starbird, and Leysia Palen. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *CHI*, pages 1079–1088, 2010. 3

Shibiao Wan, Man-Wai Mak, Bai Zhang, Yue Wang, and Sun-Yuan Kung. Ensemble random projection for multi-label classification with application to protein subcellular localization. In *ICASSP*, pages 5999–6003, 2014. 5

Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009. 45, 47, 56

Jia Wu, Zhibin Hong, Shirui Pan, Xingquan Zhu, Zhihua Cai, and Chengqi Zhang. Multi-graph-view learning for graph classification. In *ICDM*, pages 590–599, 2014. 76

Gerhard Wunder, Peter Jung, Martin Kasparick, Thorsten Wild, Frank Schaich, Yejian Chen, Stephan ten Brink, Ivan Gaspar, Nicola Michailow, Andreas Festag, Luciano Leonel Mendes, Nicolas Cassiau, Dimitri Ktenas, Marcin Dryjanski, Slawomir Pietrzyk, Bertalan Eged, Peter Vago, and F. Wiedmann. 5gnow: non-orthogonal, asynchronous waveforms for future mobile applications. *IEEE Communications Magazine*, 52(2):97–105, 2014. 3

Eleftherios Spyromitros Xioufis, William Groves, Grigorios Tsoumakas, and Ioannis P. Vlahavas. Multi-label classification methods for multi-target regression. *arXiv:1211.6581*, 2012. 75

Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. *Michigan State Universiy*, 2, 2006. 7, 45, 48

Guo-Xun Yuan, Chia-Hua Ho, and Chih-Jen Lin. An improved GLMNET for L1-regularized logistic regression. *JMLR*, 13:1999–2030, 2012. 48

Min-Ling Zhang and Kun Zhang. Multi-label learning by exploiting label dependency. In *KDD*, pages 999–1008, 2010. 14

Min-Ling Zhang and Zhi-Hua Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1338–1351, 2006. 5, 13, 71

Min-Ling Zhang and Zhi-Hua Zhou. Multi-label learning by instance differentiation. In *AAAI*, pages 669–674, 2007a. 12

Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007b. 12, 54

Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Trans. Knowledge and Data Engineering*, 26(8):1819–1837, 2014. 4, 23

Yi Zhang and Jeff G. Schneider. Multi-label output codes using canonical correlation analysis. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 873–882, 2011. 17, 19, 20, 35, 44, 47, 54, 75

Yi Zhang and Jeff G. Schneider. Maximum margin output coding. In *ICML*, pages 1575–1582, 2012. 4, 17, 18, 20, 25, 35, 44, 45, 46, 47, 54, 57, 75

Jiayu Zhou, Jianhui Chen, and Jieping Ye. Tutorial: Multi-task learning: Theory, algorithms, and applications. In *SDM*, 2012. 74

Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 2012. 12