Doctoral Dissertation

On Improving Multi-Label Classification via Dimension Reduction

Lu Sun

Division of Computer Science and Information Technology
Graduate School of Information Science and Technology
Hokkaido University

July 2017

Acknowledgements

First and foremost, I would like to express my deepest thanks and appreciation to my supervisor, Professor Mineichi Kudo. I would like to thank you for your strong support and great patience on encouraging my research work during my PhD study at Hokkaido University. Your teaching, comments and cautious attitude towards academic research will motivate and regularize my study and work during the rest of my life. I would also like to thank Associate Professor Atsuyoshi Nakamura for his invaluable discussion and advice in every seminar. My sincere thanks also go to my research committee members, Professor Tetsuo Ono and Professor Masanori Sugimoto, for contributing their ideas and time on improving my research.

I want to thank all the members of the Laboratory of Pattern Recognition and Machine Learning. It really is an interesting, enjoyable and memorable experience to be able to finish my PhD study with all of you. I especially appreciate the supports received through the collaborative work with Dr. Keigo Kimura, whose excellent research work and precious comments always motivate me to challenge myself with harder academic problems. Special thanks to Dr. Sadamori Koujaku and Dr. Ryo Watanabe for their useful advice in zasshikai and helps on setting up my working environment.

My thanks also go to the helps received from Japanese people over the past almost four years. I gratefully acknowledge the selfless helps from the members of the medical translation team in Sapporo, Mrs Kumiko Umezawa and Mrs Midori Toizumi. I am also highly grateful to the collaborative program between China Scholarship Council and Hokkaido University, which provides me adequate financial support for the living and study in Japan over the past years. As a Chinese, I would like to thank my dear country, China, on providing a peaceful and prosperous environment for me to be able to follow my dreams.

Thanks to my parents for their financial and mental supports for my academic pursuit over the long years. A lot of thanks to my wife for believing in me and encouraging me to pursue my dream in the challenging period of the passing seven years. And finally to my little baby, both your cry and smile always stimulate me to make persistent efforts and progress for our better future.

Publications

Peer Reviewed Journal Papers

- Keigo Kimura, Lu Sun and Mineichi Kudo. MLC Toolbox: A MATLAB/OCTAVE Library for Multi-Label Classification. *Journal of Machine Learning Research*, 2017. (preprint in arXiv)
- Lu Sun, Mineichi Kudo and Keigo Kimura. READER: Robust Semi-Supervised Multi-Label Dimension Reduction. *IEICE Transactions on Information and Systems*, 2017. (accepted)
- Lu Sun and Mineichi Kudo. Optimization of Classifier Chains via Conditional Likelihood Maximization. *Pattern Recognition*, 2016. (submitted in June 2016, under review)
- Lu Sun and Mineichi Kudo. Multi-Label Classification by Polytree-Augmented Classifier Chains with Label-Dependent Features. *Pattern Analysis and Applications*, 2016. (submitted in June 2016, under review)

Peer Reviewed Conference Papers

- Batzaya Norov-Erdene, Mineichi Kudo, Lu Sun and Keigo Kimura. Locality in Multi-Label Classification Problems. *In Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*, December 2016.
- Keigo Kimura, Mineichi Kudo, Lu Sun and Sadamori Koujaku. Fast Random k-labelsets for Large-Scale Multi-Label Classification. *In Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*, December 2016.
- Keigo Kimura, Mineichi Kudo and Lu Sun. Simultaneous Nonlinear Label-Instance Embedding for Multi-label Classification. *In Proceedings of the joint IAPR International Workshops on Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition (S+SSPR)*, December 2016.

- Lu Sun, Mineichi Kudo and Keigo Kimura. Multi-Label Classification with Meta-Label-Specific Features. *In Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*, December 2016.
- Lu Sun, Mineichi Kudo and Keigo Kimura. A Scalable Clustering-Based Local Multi-Label Classification Method. *In Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI)*, August 2016.
- Keigo Kimura, Mineichi Kudo and Lu Sun. Dimension Reduction Using Nonnegative Matrix Tri-Factorization in Multi-label Classification. *In Proceedings of the 21st International Conference on Parallel and Distributed Processing Techniques and Applications: Workshop on Mathematical Modeling and Problem Solving (PDPTA)*, July 2015.
- Lu Sun and Mineichi Kudo. Polytree-Augmented Classifier Chains for Multi-Label Classification. *In Proceedings of the 24th International Joint Conference of Artificial Intelligence (IJCAI)*, July 2015.

Contents

Summary

In recent years we have witnessed an explosively increasing of web-related applications in our daily lives, where the scale of data and information has grown dramatically. To deal with such a huge amount of data, machine learning becomes a crucial way to help human beings to be free from the massive tasks, such as classification, pattern recognition and prediction. As one of the fundamental tasks, classification has attracted a lot of attentions from researchers, and been specifically developed in various settings, such as binary classification and multi-class classification, in order to meet the distinct requirements of real-world applications. In this thesis, we concentrate our research on a special case of classification problems, Multi-Label Classification (MLC).

Different form the traditional single-label classification, where an instance is related with one class label, MLC aims to solve the multi-label problems, where an instance probably belongs to multiple labels. Such a generalization significantly increases the difficulty of achieving a desirable classification accuracy at a tractable time cost. Despite of this difficulty, as an appealing and challenging supervised learning problem, MLC has a wide range of real-world applications, such as text categorization, semantic image classification, bioinformatics analysis, music emotion detection and video annotation.

In general, there are two major concerns in the MLC problems. First, label correlations are strong and ubiquitous in various multi-label datasets. For example, in semantic image annotation, the labels "lake" and "reflection" share a strong correlation since they typically appear together. Thus, it is important and crucial to capture such label correlations in order to achieve a desirable classification performance. Second, as the rapid increase of web-related applications, more and more datasets emerge in high-dimensionality, whose number of instances, features and labels are far from the regular scale. For example, there are millions of videos in the video-sharing website Youtube, while each one can be tagged by some of millions of candidate categories. Such high-dimensionality of multi-label data significantly increases the time and space complexity in learning, and degrades the classification performance due to the curse of dimensionality. To cope with these two concerns, various MLC methods have been proposed in recent years, with remarkable success in a number

of applications. However, further improvement in terms of time complexity and classification accuracy is recently required.

The research objective of this thesis is to improve the performance of MLC by capturing label correlations and reducing dimensionality. According to the objective, the thesis is separated into two major parts: Part I Multi-Label Classification and Part II Multi-Label Dimension Reduction.

In part I, we focus on solving the MLC problems in terms of label correlation modeling and label-specific feature selection. Motivated by the Classifier Chains (CC) method, we propose the Polytree-Augmented Classifier Chains (PACC) in order to save label correlations in the polytree-like graphical model in sense that the limitations of error propagation and poorly ordered chain in CC can be overcame in PACC. To further improve its performance, a two-stage feature selection approach is developed by removing irrelevant and redundant features for each label. In addition, based on conditional likelihood maximization, we reconsider both label correlation modeling and feature selection via a unified framework, Optimized Classifier Chains (OCC). Under this framework, we show that existing CC-based methods and several feature selection approaches are special cases of the proposed method.

In Part II, our goal is to improve the classification performance of a multi-label classifier by decreasing the problem size. To reduce the dimensionality of features, we conduct Feature Space Dimension Reduction (FS-DR) by proposing two approaches, MLC with Meta-Label-Specific Features (MLSF) and Robust sEmi-supervised multi-lAbel DimEnsion Reduction (READER) via empirical risk minimization. Benefited from the $\ell_{2,1}$-norm loss and regularization, READER performs feature selection in a robust manner with label embedding (label correlation modeling) and manifold learning (semi-supervised learning). To avoid the problem of imperfect label information, we conduct Label Space Dimension Reduction (LS-DR) by extending READER to apply nonlinear Label Embedding (READER-LE) with a linear approximation. Furthermore, in order to utilize distributed computing, for the first time we introduce Instance Space Decomposition (ISD) and propose the Clustering-based Local MLC (CLMLC) method to evaluate its efficiency. Different with existing ISD methods, CLMLC conducts the feature-guided ISD in a feature subspace rather than the original feature space, and builds cluster-specific local models.

According to extensive empirical evidences reported in this thesis, the proposed methods successfully address the two major concerns in MLC, and achieve competitive classification performance compared with the state-of-the-art methods. Therefore, it is hopeful for researchers in the field of MLC to build their MLC systems and develop novel MLC methods based on the research work in this thesis.

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Background

As an instinct of human beings, classification has always been considered as the fundamental task of machine learning and pattern recognition. Collecting data and associating each data point with a reasonable class or label helps us to better utilize the information on hand, analyze the various patterns as well as phenomenons in the nature and make decisions on our future actions. As the boom of information technology, the scale of data and information we face with in daily life has increased dramatically in recent years. Therefore, artificial intelligence and machine learning have become a hot topic on aiding human beings to better deal with such a huge amount of information.

According to whether using the associated class information or not, the applications of machine learning can be cast into two main categories: supervised learning and unsupervised learning. Although the unsupervised learning has a wider range of applications than supervised learning, in this thesis we concentrate our research on the supervised learning. One standard and classical task of supervised learning is binary classification, which aims to find the best class (0 or 1) to an unseen test instance by the classifier learned on a training dataset. We can find numerous applications of binary classification in the real world. For example, there are two statuses of a switch, on and off; For coin tossing, we shall have two results, head and tail; In an exam, each student probably passes or loses it. To deal with these problems, a variety of methods have been developed for binary classification, such as decision trees [68], Bayesian networks [36], support vector machines [87], neural networks [35] and logistic regression [64]. These methods provide a theoretical fundamental for the solutions to the more complicated supervised learning tasks.

However, there are still a large number of applications where it is difficult to directly apply binary classification. For example, in the iris flower dataset intro-

Figure 1.1: A multi-label example from semantic image annotation (ocean ✗, ship ✗, sky ✓, airplane ✗, building ✓, tree ✓, people ✗, car ✗, lake ✓, reflection ✓).

duced by Ronald Fisher in [27], each iris flower instance belongs to one of three related species: iris setosa, iris virginica and iris versicolor, and is measured by the length and the width of its sepal and petal; In the MINST dataset [50], which collects 70,000 handwritten digit images, each image is probably assigned with a number taken from 0 to 9. To distinguish with binary classification, the machine learning technologies aiming to cope with the problems where each instance is relevant to one class taken from multiple candidate classes, are termed as multi-class classification. Various methods have been specifically designed for multi-class classification, such as k-nearest neighbors [29], multilayer perceptron [75], naive Bayes [76], multi-class SVM [22] and extreme learning machines [37]. In fact, multi-class classification is more general than binary classification, since the latter is a special case where the number of classes is limited as two.

The application of multi-class classification is limited due to its assumption that each instance belongs to only one class, which violates the problem setting of many real-world applications, such as text categorization, semantic image annotation and bioinformatics analysis. An apparent example is that a single image probably relates with several semantic objects simultaneously, like "sky", "lake", "reflection" and "trees". Therefore, it is nature to introduce a more general machine learning technique than multi-class classification. In recent years, Multi-Label Classification (MLC) [8, 92, 69, 109] has risen as an appealing and challenging supervised learning problem, where multiple class labels, rather than a single label in multi-class classification, are associated with an unseen test instance. Such a generalization greatly increases the difficulty of achieving a desirable classification accuracy at a tractable time cost. Fig. 1.1 shows a multi-label example for semantic image annotation, where five of ten candidate labels are relevant with the example image. In Fig. 1.2, the visualization in two-dimensional feature space of the Scene dataset with 294 features and 6 labels is showed, where the original instances are projected into the feature subspace by

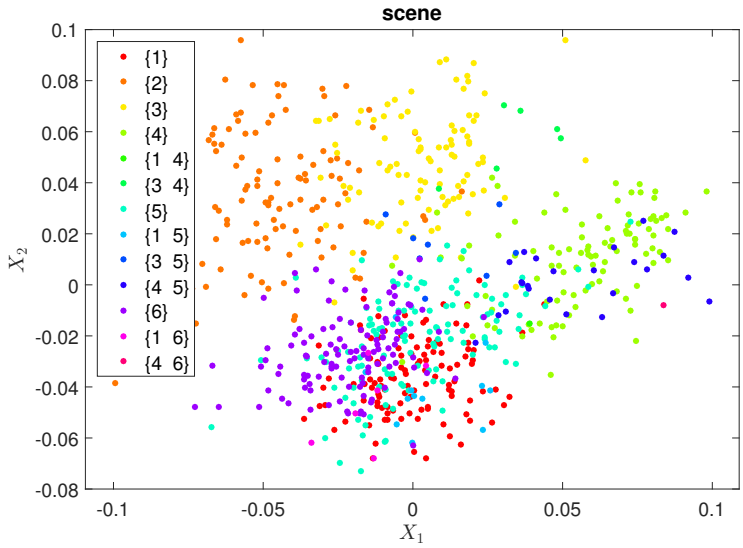


Figure 1.2: Visualization of the Scene dataset with 294 features and 6 labels. Different color indicates different label set associated with a data point.

the feature extraction approach implemented in CLMLC [85]. As shown in Fig. 1.2, a large number of instances in the Scene dataset are associated with more than one label.

The existing MLC methods can be cast into two strategies: problem transformation and algorithm adaptation [92]. A convenient and straightforward way for MLC is to conduct problem transformation in which a MLC problem is transformed into one or more single label classification problems. Problem transformation mainly comprises three kinds of methods: binary relevance (BR) [8], pairwise (PW) [30] and label combination (LC) [94]. BR simply trains a binary classifier for each label, ignoring the label correlation, which is apprantly crucial to make accurate prediction for MLC. PW and LC are designed to capture label dependence directly. PW learns a classifier for each pair of labels, and LC transforms MLC into the possible largest single-label classification problem by treating each possible label combination as a meta-label. At the expense of their high ability on modeling label correlations, the complexity increases quadratically and exponentially with the number of labels for PW and LC, respectively, thus they typically become impracticable even for a small number of labels. In the second strategy, algorithm adaptation, multi-label problems are solved by modifying conventional machine learning algorithms, such as support vector machines [24], k-nearest neighbors [112], adaboost [73], neural networks [111], decision trees [17] and probabilistic graphical models [31, 67, 32, 2]. They achieved competitive performances to those of problem transformation based methods. However, they have several limitations, such as difficulty of

choosing parameters, high complexity on the prediction phase, and sensitivity on the statistics of data. Due to the simplicity and efficiency of the problem transformation strategy, we shall focus on discuss and develop MLC methods following this strategy.

1.2 Concerns and Motivations

Due to the intrinsic properties of multi-label datasets, it is a non-trivial thing to directly conduct traditional multi-class classification methods on the multi-label data. In general, we have four major concerns in MLC:

1. Labels in multi-label datasets are typically correlated and dependent with each other, thus it is important to model label correlations for MLC model building. One simple example is that, in Fig. 1.1, the concepts "lake" and "reflection" share a strong correlation, therefore a successful MLC classifier shall model such correlation;

2. The existence of irrelevant and redundant features increases the computational complexity in both learning and prediction, and moreover reduces the generalization ability of the classifier built on instances due to the curse of dimensionality;

3. The existence of noisy labels (outliers) and incomplete labels in multi-label data should be considered. Such noisy outliers, usually resulting from the mistakes in the label annotation by human beings, would misguide the learning algorithm;

4. A large proportion of training data is unlabeled in various real-world applications. It is intractable to annotate each data point with multiple labels from a huge number of instances and candidate labels.

In this thesis, we attempt to develop MLC methods to address these problems. According to the distinct objectives of proposed methods, the thesis is separated into two major parts: multi-label classification and multi-label dimension reduction. In the first part, we mainly focus on solve the MLC problems via label correlation modeling and multi-label feature selection. Regarding to the first concern, we follow one promising MLC method, Classifier Chains (CC) [69], to propose a Polytree-Augmented Classifier Chains (PACC) [81]. In PACC, label correlations are modeled by a special probabilistic graphical model, polytree, where the problems of error propagation and poorly ordered chain in CC can be successfully avoided. As for the second concern, label-specific features are selected for PACC by a two-stage feature selection approach, based on which we propose PACC with Label-Dependent Features (PACC-LDF) [82]. Moreover, to select both relevant parent labels and useful label-specific features in CC,

we optimize traditional CC via conditional likelihood maximization, and propose Optimized CC (OCC) [83]. OCC enables to capture label correlations and useful features by model selection and multi-label feature selection, respectively.

In the second part, Multi-Label Dimension Reduction (ML-DR), we aim to improve the classification performance and reduce the computational complexity by decreasing the problem size of MLC in instances, features and labels. To decrease the dimensionality of features, we conduct Feature Space Dimension Reduction (FS-DR) by proposing two ML-DR methods, MLC with Meta-Label-Specific Features (MLSF) [84] and Robust sEmi-supervised multi-lAbel DimEnsion Reduction (READER) [86]. Both MLSF and READER are developed in order to perform multi-label feature selection by saving label correlations. Note that, by introducing manifold learning, READER enables to utilize a large amount of unlabeled data to improve its performance on selecting discriminative features. To avoid the problem of imperfect label information, we conduct Label Space Dimension Reduction (LS-DR) by extending READER to apply nonlinear Label Embedding (READER-LE). Furthermore, in order to utilize distributed computing, we introduce a novel category for ML-DR, termed as Instance Space Decomposition (ISD), and propose a Clustering-based Local MLC (CLMLC) [85] method to verify its effectiveness.

1.3 Notation

The notations used in the thesis are summarized as follows.

- Feature space: $\mathcal{X} = \mathbb{R}^M$.
- Label space: $\mathcal{Y} = \{0, 1\}^L$.
- Multi-label classifier: $h : \mathcal{X} \mapsto \mathcal{Y}$.
- Random feature vector: $\boldsymbol{X} = (X_1, ..., X_M) \in \mathcal{X}$.
- Random label vector: $\boldsymbol{Y} = (Y_1, ..., Y_L) \in \mathcal{Y}$.
- Data observation: $\mathbf{x} \in \mathbb{R}^M$, a realization of $\boldsymbol{X}$.
- Labelset observation: $\mathbf{y} \in \{0, 1\}^L$, a realization of $\boldsymbol{Y}$.
- Training set: $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, where $y_{ij} = 1$ indicates the relevance of the jth label with the ith instance, and $y_{ij} = 0$ otherwise, $\forall i, j$.
- Feature matrix: $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^\intercal \in \mathcal{R}^{N \times M}$.
- Label matrix: $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_N]^\intercal \in \{0, 1\}^{N \times L}$.
- Test instance: $\hat{\mathbf{x}}$, whose prediction is $\hat{\mathbf{y}} \leftarrow h(\hat{\mathbf{x}})$.

- Loss function: $loss(\cdot)$, e.g, $loss(\mathbf{y}, \hat{\mathbf{y}})$ is the loss between $\mathbf{y}$ and $\hat{\mathbf{y}}$.
- Regularization: $\Omega(\cdot)$; e.g. $\Omega(h)$ denotes the regularization term on h.
- Probability distribution: p; e.g., $p(Y|X)$ denotes the probability distribution of random variable Y conditioned on random variable X.
- Expectation: $\mathbb{E}$; e.g., $\mathbb{E}_{\boldsymbol{XY}} loss(\boldsymbol{Y}, h(\boldsymbol{X}))$ denotes the expected loss of training data over the joint distribution $p(\boldsymbol{X}, \boldsymbol{Y})$.
- Entropy: $H(\cdot)$; e.g., $H(X) = -\mathbb{E}_X \log p(X)$.
- Conditional entropy: $H(\cdot|\cdot)$; e.g., $H(X|Y) = -\mathbb{E}_{XY} \log p(X|Y)$.
- Mutual information: $I(\cdot;\cdot)$; e.g., $I(X;Y) = H(X) - H(X|Y)$.
- Cardinality: $|\cdot|$, which meansures the cardinality; e.g, $|\mathbf{y}| = \sum_{j=1}^{L} y_j$.
- Indicator function: $\mathbb{1}_{(\cdot)}$; e.g. $\mathbb{1}_{\mathbf{y}=\hat{\mathbf{y}}}$. equals to 1 if $\mathbf{y} = \hat{\mathbf{y}}$, and 0 otherwise.
- Probabilistic classifier: $f : \mathcal{X} \mapsto [0, 1]$.

Without loss of generality, several matrix operations are defined for arbitrary matrices $\mathbf{Z} \in \mathbb{R}^{n \times m}$ and $\mathbf{G} \in \mathbb{R}^{n \times q}$.

- Row vector in the ith row of $\mathbf{Z}$: $\mathbf{Z}_{i\cdot}$.
- Column vector in the jth column of $\mathbf{Z}$: $\mathbf{Z}_{\cdot j}$.
- Element in the ith row and jth column: $\mathbf{Z}_{ij}$.
- ℓ_1-norm: $\|Z_{\cdot j}\|_1 = \sum_{i=1}^{n} |\mathbf{Z}_{ij}|$.
- ℓ_2-norm: $\|Z_{\cdot j}\|_2 = \sqrt{\sum_{i=1}^{n} \mathbf{Z}_{ij}^2}$.
- Frobenius norm: $\|Z\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{m} \mathbf{Z}_{ij}^2} = \sqrt{\sum_{i=1}^{n} \|Z_{i\cdot}\|_2^2}$.
- $\ell_{2,1}$-norm: $\|Z\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{m} \mathbf{Z}_{ij}^2} = \sum_{i=1}^{n} \|Z_{i\cdot}\|_2$.
- Trace: $\mathrm{Tr}(\mathbf{Z}^\intercal \mathbf{Z}) = \sum_{j=1}^{m} (\mathbf{Z}^\intercal \mathbf{Z})_{jj}$.
- Identity matrix: $\mathbf{I}$; e.g., $\mathbf{I}_n = \mathrm{diag}(1, ..., 1)$ with n diagonal elements.
- Centering matrix: $\mathbf{C}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\intercal$, where $\mathbf{1} = [1, ..., 1]^\intercal$ is a n-dimensional vector; e.g., $\mathbf{C}_n\mathbf{Z}$ centers $\mathbf{Z}$ in columns.
- Scatter/Covariance matrix between $\mathbf{G}$ and $\mathbf{Z}$: $\mathbf{S}_{GZ} = \mathbf{G}^\intercal \mathbf{C}_n \mathbf{Z}$.
- Round-based decoding: round$(\cdot)$; e.g. round$([-0.2, 0.8, 1.6]) = [0, 1, 2]$.

1.4 Thesis Organization

According to the concerns and motivations mentioned previously, the remainder of this thesis consists of eight chapters, which can be cast into three parts: Part I Multi-Label Classification (MLC) (Chapters 2 to 4), Part II Multi-Label Dimension Reduction (ML-DR) (Chapters 5 to 8) and Part III Conclusion (Chapter 9). In Part I, we give the mathematical definition of MLC, and then focus on solving MLC by capturing label correlations and mining label-specific features from a probabilistic framework. Part II first introduces the objective and related work of ML-DR, and then concentrates on ML-DR from three viewpoints: Feature Space Dimension Reduction (FS-DR), Label Space Dimension Reduction (LS-DR) and Instance Space Decomposition (ISD). Both FS-DR and LS-DR are conducted via empirical risk minimization, while ISD is introduced by decomposing one original dataset into several smaller-scale datasets via clustering.

Specifically, the thesis is organized as follows. In Part I, Chapter 2 presents the problem of MLC by introducing the statistics of popular benchmark multi-label datasets as well as evaluation metrics, and discusses the related work on CC-based methods, which motivates the following research work in Part I. Chapter 3 reconsiders CC from a novel probabilistic graphical model, the polytree structure, avoiding CC's limitations of error propagation and poorly ordered chain, and develops a two-stage feature selection framework on mining label-dependent features in order to improve its performance. Chapter 4 generalizes previous CC-based methods and several popular information theoretic feature selection approaches via conditional likelihood maximization, and provides theoretical analysis and extensive empirical evidence to support our propositions. In Part II, Chapter 5 makes a brief introduction on ML-DR and categorizes the existing ML-DR methods into three major strategies. Chapter 6 focuses on FS-DR from the viewpoint of empirical risk minimization and proposes two novel methods, the supervised MLSF and semi-supervised READER. Chapter 7 illustrates the framework of LS-DR and extends READER to be able to conduct LS-DR through a linear approximation on the nonlinear label embedding. Chapter 8 presents the details of ISD and develops a clustering-based local method to demonstrate its efficiency. Finally, in Part III, Chapter 9 concludes this thesis and discusses the future work motivated by the thesis.

Part I

Multi-Label Classification

Chapter 2

Introduction

2.1 Multi-Label Classification

Unlike traditional single label classification problems where an instance is associated with a single-label, multi-label classification (MLC) attempts to allocate multiple labels to any input unseen instance by a multi-label classifier learned from a training set. Obviously, such a generalization greatly raises the difficulty of obtaining a desirable prediction accuracy at a tractable complexity. Nowadays, MLC has drawn a lot of attentions in a wide range of real world applications, such as text categorization [105], semantic image annotation [8] and bioinformatics analysis [112]. For example, a news article probably relates to multiple topics, like "economic", "politics", "technology", etc; one image is possibly relevant to a set of semantic concepts, like "sky","lake", "reflection", etc; maybe a gene is associated with several functional classes, like "metabolism", "energy", "cellular biogenesis", etc. Fig. 2.1[1] shows a multi-label example, where Fig. 2.1(a) belongs to labels "fish" and "sea" and Fig. 2.1(b) is assigned with labels "windsock" and "sky." Note that both objects are *fish*, but the contexts (backgrounds), in other words, the label dependencies, distinguish them clearly. To capture label correlations has been shown that it is crucial to make accurate prediction for MLC in a variety of papers. For example, it is quite difficult to distinguish the labels "fish" and "windsock" in Fig. 2.1 from the visual features, unless we consider the label correlations with "sea" and "sky."

2.1.1 Multi-Label Datasets and Statistics

The statistics of popular benchmark multi-label datasets in Mulan [95] and Meka [71] are summarized in Table 2.1, where we report only the first two subsets of the Rcv1 and Corel16k datasets, since the subsets share similar statistics. In

[1] http://www.yunphoto.net

Table 2.1: The statistics of benchmark multi-label datasets. "LCard", "LDen" and "LDist" denote the label cardinality, label density and the number of distinct label combinations, respectively.

Dataset	L	M	N	LCard	LDen	LDist	Domain
Emotions	6	72	593	1.869	0.311	27	music
Scene	6	294	2407	1.074	0.179	15	image
Flags	7	19	194	3.392	0.485	54	image
Yeast	14	103	2417	4.237	0.303	198	biology
Birds	19	260	645	1.014	0.053	133	audio
Tmc2007	22	500	28596	2.158	0.098	1341	text
Mirfilckr	24	150	25000	3.716	0.155	2880	image
Genbase	27	1186	662	1.252	0.046	32	biology
Medical	45	1449	978	1.245	0.028	94	text
Enron	53	1001	1702	3.378	0.064	753	text
Language	75	1004	1460	1.180	0.016	286	text
Rcv1s1	101	944	6000	2.880	0.029	837	text
Rcv1s2	101	944	6000	2.634	0.026	800	text
Mediamill	101	120	43907	4.376	0.043	6555	video
Bibtex	159	1836	7395	2.402	0.015	1654	text
Corel16k1	153	500	13766	2.859	0.019	1791	image
Corel16k2	164	500	13761	2.882	0.018	1782	image
CAL500	174	68	502	26.044	0.150	502	music
Bookmarks	208	2150	87856	2.028	0.010	18716	text
Corel5k	374	499	5000	3.522	0.009	1453	image
Delicious	983	500	16105	19.020	0.019	3937	text

(a) "fish","sea" (b) "windsock","sky"

Figure 2.1: A multi-label example to show label dependency for inference. (a) A fish in the sea; (b) carp shaped windsocks (koinobori) in the sky.

Tabel 2.1, "LCard", "LDen" and "LDist" denote the label cardinality, label density and the number of distinct label combinations, respectively. "Domain" shows the source of each dataset. Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$, where $\mathbf{x} \in \mathbb{R}^M$ and $\mathbf{y} \in \{0,1\}^L$, the statistics are defined as follows,

- **LCard** $\coloneqq \frac{1}{N}\sum_{i=1}^{N} |\mathbf{y}_i|$: the label cardinality measures the average number of labels per instance,
- **LDen** $\coloneqq \frac{1}{NL}\sum_{i=1}^{N} |\mathbf{y}_i|$: the label density equals to the label cardinality normalized by the number of labels,
- **LDist** $\coloneqq |\{\mathbf{y}|(\mathbf{x},\mathbf{y}) \in \mathcal{D}\}|$: the number of distinct label sets in the dataset $\mathcal{D}$.

2.1.2 Multi-Label Evaluation Metrics

The existing multi-label evaluation metrics can be separated into two groups: instance-based metrics and label-based metrics [92]. Given a test data set $\mathcal{T} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N_t}$, the evaluation metrics are given as follows:

- Instance-based metrics
  - **Exact-Match** $\coloneqq \frac{1}{N_t}\sum_{i=1}^{N_t} \mathbb{1}_{\hat{\mathbf{y}}_i = \mathbf{y}_i}$,
  - **Hamming-Score** $\coloneqq \frac{1}{N_t L}\sum_{i=1}^{N_t}\sum_{j=1}^{L} \mathbb{1}_{\hat{y}_{ij} = y_{ij}}$,
  - **Accuracy** $\coloneqq \frac{1}{N_t}\sum_{i=1}^{N_t} \frac{\langle \hat{\mathbf{y}}_i, \mathbf{y}_i \rangle}{|\hat{\mathbf{y}}_i| + |\mathbf{y}_i| - \langle \hat{\mathbf{y}}_i, \mathbf{y}_i \rangle}$,
- Label-based metrics:
  - **Macro-F1** $\coloneqq \frac{1}{L}\sum_{j=1}^{L} \frac{2\sum_{i=1}^{N_t} \hat{y}_{ij} \times y_{ij}}{\sum_{i=1}^{N_t}(\hat{y}_{ij} + y_{ij})}$,
  - **Micro-F1** $\coloneqq \frac{2\sum_{i=1}^{N_t} \langle \hat{\mathbf{y}}_i, \mathbf{y}_i \rangle}{\sum_{i=1}^{N_t}(|\hat{\mathbf{y}}_i| + |\mathbf{y}_i|)}$,

Among the metrics, Exact-Match is the most stringent measure, especially in the case with a large number of labels. It does not evaluate partial match of labels. Despite such a limitation, according to the definition, it is a good measure to know how well label correlations are modeled. Hamming-Score emphasizes on the prediction accuracy on label-instance pairs, able to evaluate the performance on each single label. Accuracy is useful to know the performance of classifier in terms of both positive and negative prediction ability. Unlike Exact-Match, either Macro-F1 or Micro-F1 is able to take the partial match into account. In addition, as stated in [89], Macro-F1 is more sensitive to the performance of rare categories (the labels in minority), while Micro-F1 is affected more by the major categories (the labels in majority). Hence, joint use of Macro-F1 and Micro-F1 should be a good supplement for the instance-based evaluation metrics to evaluate the performances of MLC methods.

2.1.3 Problems and Challenges

In Table 2.2, we show a specific example on the multi-label Scene dataset. Hence, our objective is to find a projection h from the features $\mathbf{x} \in \mathcal{X} = \mathbb{R}^M$ to the labels $\mathbf{y} \in \mathcal{Y} = \{0,1\}^L$ by minimizing a specific loss function on the training instances. Then, given a test instance $\hat{\mathbf{x}}$, we can obtain the prediction by $\hat{\mathbf{y}} \leftarrow h(\hat{\mathbf{x}})$.

Now we can give the mathematical definition on MLC, which is the fundamental formulation for the rest of this thesis. The task of MLC is to find an optimal classifier $h : \mathbb{R}^M \mapsto \{0,1\}^L$, which assigns a label vector $\hat{\mathbf{y}} = h(\mathbf{x})$ to each instance $\mathbf{x}$ such that h minimizes a loss function between $\hat{\mathbf{y}}$ and $\mathbf{y}$. For a loss function $loss(\cdot)$, the optimal classifier h^* is

$$h^* = \arg\min_h \mathbb{E}_{\boldsymbol{XY}} loss(\boldsymbol{Y}, h(\boldsymbol{X})). \tag{2.1}$$

Specifically, given the subset 0-1 loss $loss_s(\mathbf{y}, \hat{\mathbf{y}}) = \mathbb{1}_{\mathbf{y} \neq \hat{\mathbf{y}}}$, where $\mathbb{1}_{(\cdot)}$ denotes the indicator function, (2.1) can be rewritten in a pointwise way,

$$\hat{\mathbf{y}} = h^*(\mathbf{x}) = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}). \tag{2.2}$$

Here we use $p(\boldsymbol{Y}|\boldsymbol{X})$ to represent the conditional probability distribution of label variables $\boldsymbol{Y}$ given feature variables $\boldsymbol{X}$.

Due to the intrinsic properties of multi-label datasets, there are several challenges in the front of researchers in the field of MLC. We summarize the major challenges in the following.

Existence of label correlations

It has been shown in several papers [69, 32, 38] that modeling label correlations helps to improve the classification performance. In fact, the class labels are typically correlated and dependent with other labels. There are two types of label

Table 2.2: Instances from the Scene set, with 294 features and 6 labels.

Feature vector $\mathbf{x}$				Label vector $\mathbf{y}$						Label set $\mathcal{L}$
x_1	x_2	...	x_{294}	y_1	y_2	y_3	y_4	y_5	y_6	$\{\lambda_1, ..., \lambda_6\}$
0.65	0.67	...	0.03	**1**	0	0	0	**1**	0	$\{\lambda_1, \lambda_5\}$
0.18	0.43	...	0.00	0	**1**	0	0	0	0	$\{\lambda_2\}$
0.82	0.65	...	0.11	0	0	**1**	**1**	0	0	$\{\lambda_3, \lambda_4\}$
0.48	0.43	...	0.01	0	0	0	0	**1**	0	$\{\lambda_5\}$
0.88	0.90	...	0.22	0	0	0	0	0	**1**	$\{\lambda_6\}$
0.39	0.43	...	0.11	**1**	0	0	0	0	**1**	$\{\lambda_1, \lambda_6\}$
A test instance $\mathbf{x}$				The prediction $\hat{\mathbf{y}}$						Label set $\mathcal{L}$
x_1	x_2	...	x_{294}	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$	$\hat{y}_4$	$\hat{y}_5$	$\hat{y}_6$	$\{\lambda_1, ..., \lambda_6\}$
0.25	0.50	...	0.09	?	?	?	?	?	?	?

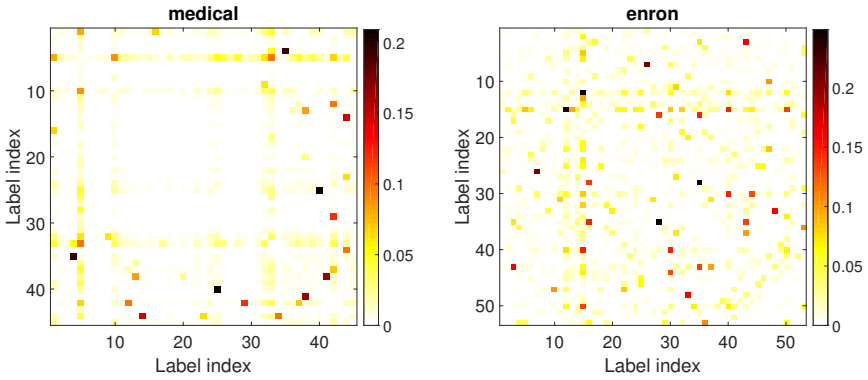


Figure 2.2: Visualization of label correlations on the Medical ($L = 45$) and Enron ($L = 53$) datasets by mutual information.

correlations for MLC, *marginal* and *conditional* dependence. According to the structure of a Bayesian network for $p(\boldsymbol{X}, \boldsymbol{Y})$, we have a marginal distribution and a conditional distribution as

$$p(\boldsymbol{Y}) = \prod_{j=1}^{L} p(Y_j | \boldsymbol{Pa}_j), \quad p(\boldsymbol{Y} | \boldsymbol{X}) = \prod_{j=1}^{L} p(Y_j | \boldsymbol{Pa}_j, \boldsymbol{X}), \tag{2.3}$$

where $\boldsymbol{Pa}(Y_j)$ represents the parent label set of Y_j. Then the following definition on label dependence is induced:

Definition 1. *Label random vector $\boldsymbol{Y}$ is called marginally (or conditionally) independent if $\forall Y_j : \boldsymbol{pa}(Y_j) = \emptyset$ in (2.3).*

Fig. 2.2 shows the mutual information matrices of labels on the Medical and Enron datasets, where the warmer the color is, the stronger the label-pair correlation. In the same way demonstrated in Fig. 2.2, we can see that label-pair correlation is prevalent in many datasets from the values of mutual information $I(Y_j; Y_k)$ for any pair of Y_j and Y_k.

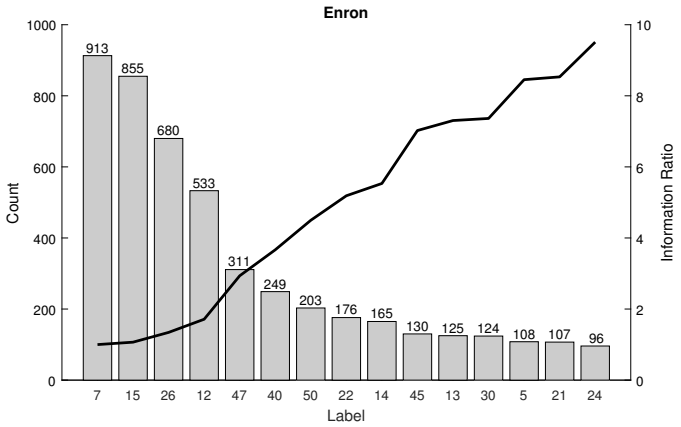


Figure 2.3: The label imbalance problem in the Enron dataset, where 15 most frequent labels are reported.

High-dimensional real-world datasets

The existence of irrelevant and redundant features for classification increases the computational complexity in both learning and prediction, and moreover often reduces the generalization ability of the classifiers designed from instances due to the curse of dimensionality. Irrelevant and redundant features are two distinct concepts, an irrelevant feature has no discriminative information, while a redundant feature shares the same discriminative information with other features. Removal of these features, therefore, does not lose the discriminative information. Rather, removing irrelevant and redundant features simplifies the learning phase and prevents overfitting.

In addition, MLC often confronts with large-scale datasets, where either of the number of labels L, attributes M and instances N might be very large. In such a case, the time complexity will become an important aspect for evaluating an MLC algorithm, sometimes more important than classification accuracy for real-world applications.

Label imbalance

Multi-label datasets are typically imbalanced, i.e., the number of instances associated with each label is often unequal. Fig 2.3 shows the label imbalance problem in the Enron dataset, where 15 most frequent labels are reported. Note that the value of imformation ratio (defined by (3.10) in Chapter 3) will increase once the imbalance problem becomes severe. In addition, the ratio of positive instances against the negative ones may be low for some labels. The imbalance problem usually harms the performance of the learned classifier from two points of view. On one hand, if we aim at minimizing hamming loss or ranking loss,

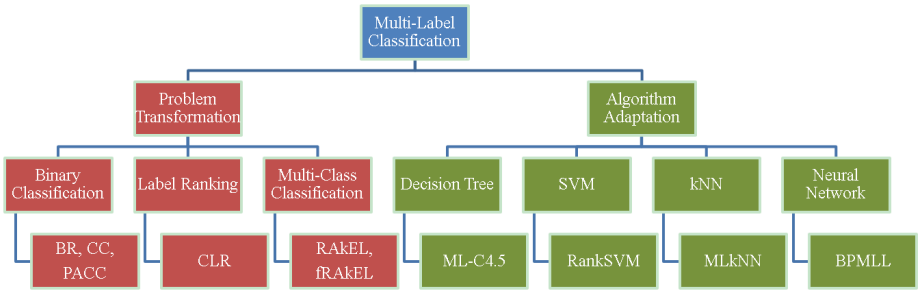


Figure 2.4: Two strategies of existing MLC methods.

we tend to ignore minority labels. On the other hand, a classifier for minority classes is difficult to design.

A large amount of unlabeled instances

In the real-world applications of MLC, it is reasonable that a large part of training instances are unlabeled. Because it is typically too expensive or even intractable to annotate each data point with multiple labels from a huge number of number of instances and candidate labels. Hence, it is necessary to design a MLC method which can handle such cases, and improve its classification performance by utilizing the unlabeled training data.

2.2 Two Strategies for Multi-Label Classification

The existing MLC methods fall into two broad strategies [92]:

- Problem Transformation
  - Transform an MLC problem into one multi-class or a set of binary problems
  - Transform to Binary Classification, Label Ranking or Multi-Class Classification
- Algorithm Adaptation
  - Adapt traditional machine learning algorithms in the MLC setting
  - Include ML-C4.5 [17], RankSVM [24], MLkNN [112], BPMLL [111], etc

Fig 2.4 summarizes the representative MLC methods following the two strategies. As a convenient and straightforward way for MLC, problem transformation

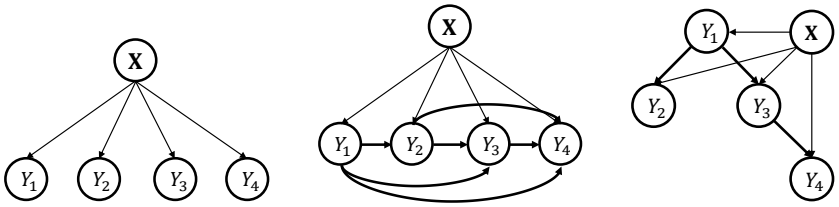
strategy transforms an MLC problem into one or a set of single-label classification problems, and learns one or a family of classifiers for modeling the single-label memberships. Most of popular baseline MLC methods, such as Binary Relevance (BR) [8], Calibrated Label Ranking (CLR) [30], and Label Powerset (LP) [94], belong to this strategy. Algorithm adaptation strategy induces conventional machine learning algorithms in the multi-label settings. Various MLC methods adopting one of the above two strategies have been developed and succeeded in dealing with multi-label problems.

2.2.1 Problem Transformation

Problem transformation transforms a MLC problem into one or set of single-labee classification problem. Currently, there are three ways for problem transformation: Binary Classification (BC), Label Ranking (LR) and Multi-Class Classification (MCC). BC transforms one MLC problem into a set of binary classification problem. The representative methods are Binary Relevance (BR) [8] and Classifier Chains (CC) [69]. BR simply trains a binary classifier for each label, totally ignoring label correlations. CC models label correlations by augmenting the feature space with parent labels following a randomly selected chain of labels. PACC [81] uses the polytree structure to model the reasonable conditional dependence between labels over features, preventing from problems of CC on poorly ordered chain and error propagation. LR and MCC are designed to model label dependence directly. LR builds a classifier for each pair of labels, while MCC transforms MLC into the possible largest single-label classification problem by treating each possible label combination as a new class label. The complexity increases quadratically and exponentially with the number of labels for LR and MCC, respectively, thus they typically become impracticable even for a small number of labels. To reduce the time complexity of MCC, RAndom k-labELsets (RAkEL) [94] randomly samples some label subsets with relatively small size from the labels, and apply multi-class classification on each label subset. In the proposed fast RAkEL (fRAkEL) [44], a two-stage classification strategy is employed to accelerate RAkEL by reducing the number of instances used in each local model.

2.2.2 Algorithm Adaptation

In the second strategy, algorithm adaptation, multi-label problems are solved by modifying conventional machine learning algorithms, such as support vector machines [24], k-nearest neighbors [112], adaboost [73], neural networks [111], decision trees [17] and probabilistic graphical models [31, 67, 32, 2]. They achieved competitive performances to those of problem transformation based methods. However, they have several limitations, such as difficulty of choosing parame-



(a) BR (b) CC and PCC (c) BCC

Figure 2.5: Probabilistic graphical models of BR and CC based methods.

ters, high complexity on the prediction phase, and sensitivity on the statistics of data.

2.3 Modeling Label Correlations by Classifier Chains

2.3.1 Classifier Chains based Methods

Binary Relevance (BR)

As the most simple and straightforward MLC method, BR [8] trains a multi-label classifier h comprised of L binary classifiers $h_1, ..., h_L$, where each h_j predicts $\hat{y}_j \in \{0, 1\}$, forming a vector $\hat{\mathbf{y}} \in \{0, 1\}^L$. Fig. 2.5(a) shows the probabilistic graphical model of BR. Here, the model

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{j=1}^{L} p(Y_j|\boldsymbol{X}) \tag{2.4}$$

means that the class labels are mututally independent.

Classifier Chains (CC)

Classifier chains (CC) [69] models label correlations in a randomly ordered chain based on (2.5).

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{j=1}^{L} p(Y_j|\boldsymbol{Pa}_j, \boldsymbol{X}). \tag{2.5}$$

Here $\boldsymbol{Pa}_j$ represents the parent labels for Y_j. Obviously, $|\boldsymbol{Pa}_j| = q$, where q is the number of labels prior to Y_j following the chain order.

In the training phase, according to a predefined chain order, it bulids L binary classifiers $h_1, ..., h_L$ such that each classifier predicts correctly the value of y_j by referring to $\mathbf{pa}_j$ in addition to $\mathbf{x}$. Here $\mathbf{pa}_j$ denotes a realization of

$\boldsymbol{Pa}_j$. In the testing phase, it predicts the value of y_j in a greedy manner:

$$\hat{y}_j = \arg\max_{y_j} p(y_j|\hat{\mathbf{pa}}_j, \mathbf{x}), \ \forall j. \tag{2.6}$$

The final prediction is the collection of the results, $\hat{\mathbf{y}}$. The computational complexity of CC is $\mathcal{O}(L \times T(M, N))$, where $T(M, N)$ is the complexity of constructing a learner for M attributes and N instances. Its complexity is identical with BR's, if a linear baseline learner is used. Fig. 2.5(b) shows the probabilistic graphical model of CC following the order of $Y_1 \to Y_2 \to \cdots \to Y_L$.

CC suffers from two risks: if the chain is wrongly ordered in the training phase, then the prediction accuracy can be degraded, and if the previous prediction of labels was wrong in the testing phase, then such a mistake can be propagated to the succeeding prediction.

Probabilistic Classifier Chains (PCC)

In the light of risk minimization and Bayes optimal prediction, probabilistic classifier chains (PCC) [19] is proposed. PCC approximates the joint distribution of labels, providing better estimates than CC at the cost of higher computational complexity.

The conditional probability of the label vector $\boldsymbol{Y}$ given the feature vector $\boldsymbol{X}$ is the same as CC. Accordingly, PCC shares the model (2.5) and Fig. 2.5(b) with CC.

Unlike CC which predicts the output in a greedy manner by (2.6), PCC examines all the 2^L paths in an exhaustive manner:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}). \tag{2.7}$$

PCC is a better method in accuracy, but the exponential cost limits its application even for a moderate number of labels, typically no more than 15.

Bayesian Classifier Chains (BCC)

BCC [107] introduces a Bayesian network to find a reasonable connection between labels before building classifier chains. Specifically, BCC constructs a Bayesian network by deriving a maximum-cost spanning tree from marginal label dependence.

The learning phase of BCC consists of two stages: learning of a tree-structured Bayesian network and building of L binary classifiers following a chain. The chain is determined by the directed tree, which is established by randomly choosing a label as its root and by assigning directions to the remaining edges.

BCC also shares the same model (2.5) with CC and PCC. Note that because of the tree structure, $|\boldsymbol{Pa}_j| \leq 1$ in BCC unlike $|\boldsymbol{Pa}_j| = q$ in CC and PCC, limiting its ability on modeling label dependence. Fig. 2.5(c) shows an example of the probabilistic graphical model of BCC with five labels.

Chapter 3

Polytree-Augmented Classifier Chains

3.1 Polytree-Augmented Classifier Chains (PACC)

We propose a novel polytree-augmented classifier chains (PACC) as a compromise between the expression ability and the efficiency. A *polytree* (Fig. 3.1) is a directed acyclic graph whose underlying undirected graph is a tree but a node can have multiple parents [72]. That is, it is more flexible than trees. A *causal basin*, as shown in Fig. 3.1(b), is a subgraph which starts with a multi-parent node and continues following a causal flow to include all the descendants and their direct parents.

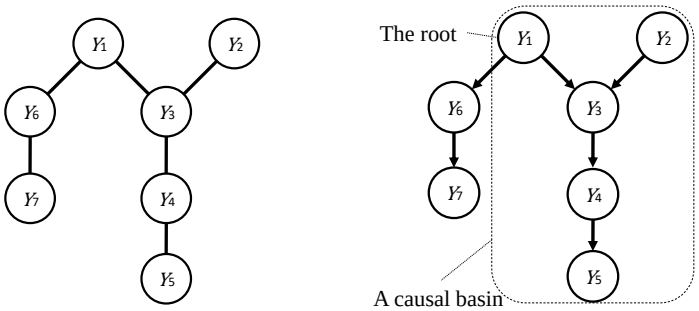
3.1.1 Structure Learning from a Probabilistic Framework

In PACC, the conditional label dependence is obtained by approximating the true distribution $p(\boldsymbol{Y}|\boldsymbol{X})$ by another distribution. According to Chou-liu's proof [16] and our previous work [81], we can have its feature-conditioned version.

Theorem 1. *To approximate a conditional distribution $p(\boldsymbol{Y}|\boldsymbol{X})$, the optimal Bayesian network B^* in K-L divergence is obtained if the sum of conditional mutual information between each variable of $\boldsymbol{Y}$ and its parent variables given the observation of $\boldsymbol{X}$ is maximized.*

Proof. The optimization problem of (4.13) can be developed as follows:

$$\begin{aligned} B^* &= \arg\min_B \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p(\boldsymbol{Y}|\boldsymbol{X})}{p_B(\boldsymbol{Y}|\boldsymbol{X})} \right\} \\ &= \arg\max_B \mathbb{E}_{\boldsymbol{XY}} \left\{ \log p_B(\boldsymbol{Y}|\boldsymbol{X}) \right\} + H(\boldsymbol{Y}|\boldsymbol{X}), \end{aligned}$$



(a) A tree skeleton (b) A polytree

Figure 3.1: A polytree with a causal basin.

$H(\boldsymbol{Y}|\boldsymbol{X})$ can be omitted due to its independence with B, thus we have

$$\begin{aligned} B^* &= \arg\max_B \mathbb{E}_{\boldsymbol{XY}} \{\log p_B(\boldsymbol{Y}|\boldsymbol{X})\} \\ &= \arg\max_B \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \prod_{j=1}^{L} p(Y_j|\boldsymbol{Pa}_j, \boldsymbol{X}) \right\} \\ &= \arg\max_B \sum_{j=1}^{L} \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p(Y_j, \boldsymbol{Pa}_j|\boldsymbol{X})}{p(Y_j|\boldsymbol{X})p(\boldsymbol{Pa}_j|\boldsymbol{X})} \cdot p(Y_j|\boldsymbol{X}) \right\} \\ &= \arg\max_B \sum_{j=1}^{L} I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}) - \sum_{j=1}^{L} H(Y_j|\boldsymbol{X}). \end{aligned}$$

Since $\sum_{j=1}^{L} H(Y_j|\boldsymbol{X})$ is independent of B, we reach our conclusion:

$$B^* = \arg\min_B D_{KL}(p||p_B) = \arg\max_B \sum_{j=1}^{L} I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}). \tag{3.1}$$

□

Theorem 1 shows

$$\min_B D_{KL}(p(\boldsymbol{Y}|\boldsymbol{X})||p_B(\boldsymbol{Y}|\boldsymbol{X})) = \max_B \sum_{j=1}^{L} I_p(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}). \tag{3.2}$$

That is, we should construct B so as to maximize the mutual information between a child and its parents. However, in practice, we do not know the true $p(\boldsymbol{Y}|\boldsymbol{X})$. Therefore we use the empirical distribution $\hat{p}(\boldsymbol{Y}|\boldsymbol{X})$ instead. Unfortunately, learning of the optimal B^* is NP-hard in general, we limit our hypothesis B to the ones satisfying $|\boldsymbol{Pa}_j| \leq 1$ so as to $\boldsymbol{Pa}_j = Y_k$ for some $k \in \{1, 2, ..., L\}$ or null, indicating the tree skeleton is to be built. In practice, we carry out Chou-liu's algorithm [16] to obtain the maximum-cost spanning tree (Fig. 3.1(a)), maximizing the weight sum, with edge weights $I_{\hat{p}}(Y_i; \boldsymbol{Pa}_j|\boldsymbol{X})$.

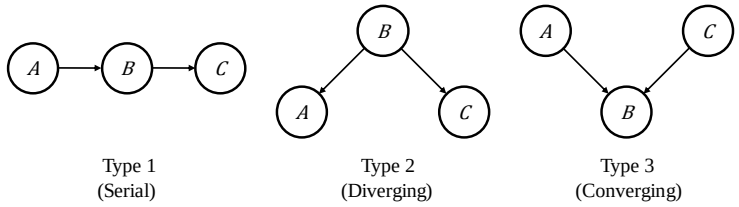


Figure 3.2: Three basic types of adjacent triplets A, B, C.

3.1.2 Construction of PACC

It is quite difficult to estimate conditional probability $P(\boldsymbol{Y}|\boldsymbol{X})$, when $\boldsymbol{X}$ is continuous. Recently some methods [18, 107, 110] have been proposed to solve this problem. In BCC [107], as an approximation of conditional probability, marginal probability of labels $\boldsymbol{Y}$ is obtained by simply counting the frequency of occurrence. Similar with [18], LEAD [110] directly obtains conditional dependence by estimating the degree of dependency of errors in multivariate regression models.

In [81], we used a more general approach to estimate the conditional probability. The data set $\mathcal{D}$ is splitted into two sets: a training set $\mathcal{D}_t$ and a hold-out set $\mathcal{D}_h$. Probabilistic classifiers, outputing the probability of each label, are learned from $\mathcal{D}_t$ to represent conditional probability of labels, and the probability is calculated based on the output of the learned classifiers over $\mathcal{D}_h$. First, three probabilistic classifiers f_j, f_k and $f_{j|k}$ are learned on $\mathcal{D}_t$ to approximate conditional probabilities $\hat{p}(y_j = 1|\mathbf{x})$, $\hat{p}(y_k = 1|\mathbf{x})$ and $\hat{p}(y_j = 1|y_k, \mathbf{x})$, respectively. Then corresponding probabilities are computed by conducting f_j, f_k and $f_{j|k}$ on $\mathcal{D}_h$. Last, $I_{\hat{p}}(Y_j; Y_k|\boldsymbol{X})$ is estimated by

$$I_{\hat{p}}(Y_j; Y_k|\boldsymbol{X}) = \frac{1}{|\mathcal{D}_h|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}_h} \mathbb{E}_{\hat{p}(y_j|y_k,\mathbf{x})} \mathbb{E}_{\hat{p}(y_k|\mathbf{x})} \log \frac{\hat{p}(y_j|y_k, \mathbf{x})}{\hat{p}(y_j|\mathbf{x})}. \tag{3.3}$$

After obtaining the skeleton of the polytree, our next task is to assign directions to its edges, that is, an ordering of the nodes to complete the polytree. First we assign some or all directions to the skeleton by finding causal basins. This is implemented by finding multi-parent nodes and the corresponding directionality. The detailed procedure is as follows. Fig. 3.2 shows three possible graphical models over triplets A, B and C. Here Types 1 and 2 are indistinguishable because they share the same joint distribution, while Type 3 is different from Types 1 and 2. In Type 3, A and C are marginally independent, so that we have

$$I(A; C) = \sum_{a,c} p(a, c) \log \frac{p(a, c)}{p(a)p(c)} = 0. \tag{3.4}$$

In this case, B is a multi-parent node. More generally, we can do Zero-Mutual Information (Zero-MI) testing for a triplet, Y_j with its two neighbors Y_a and Y_b:

if $I(Y_a; Y_b) = 0$, then Y_a and Y_b are parents of Y_j, and Y_j becomes a multi-parent node. The other non-parent neighbors will be treated as Y_j's child nodes. By performing the Zero-MI testing for every pair of Y_j's direct neighbors, $\boldsymbol{Pa}_j$ and a causal flow outside Y_j is determined, by which a causal basin will be found. In PACC, $\boldsymbol{Pa}_j$ can be more than one node, so that the model is more flexible than that of BCC using a tree.

In order to build a classifier chain by the learned directions, we rank the labels to form a chain and then train a classifier for every label following the chain. The ranking strategy is simple: the parents should be ranked higher than their descendants, and the parents sharing the same child should be ranked in the same level. Hence, learning of a label is not performed until the labels with higher ranks, including its parents, have been learned. That is, a kind of lazy decision is made. In PACC, we choose logistic regression with ℓ_2 regularization as the baseline classifier. Therefore, a set of L logistic regressiors $\mathbf{f} = \{f_j\}_{j=1}^{L}$ is learned, each of which is trained by treating the union of $\mathbf{x}$ and $\mathbf{pa}_j$ as new augmented attributes $\tilde{\mathbf{x}} = \mathbf{x} \cup \mathbf{pa}_j$, shown as follows:

$$f_j(\tilde{\mathbf{x}}, \boldsymbol{\theta}_j) = p(y_j = 1|\tilde{\mathbf{x}}, \boldsymbol{\theta}_j) = \frac{1}{1 + e^{-\boldsymbol{\theta}_j^T \tilde{\mathbf{x}}}}, \quad j = 1, ..., L, \tag{3.5}$$

where $\boldsymbol{\theta}_j$ is the model parameters for Y_j, which could be learned by maximizing the regularized log-likelihood given the training set:

$$\max_{\boldsymbol{\theta}_j} \sum_{i=1}^{N} \log p(y_{ij}|\tilde{\mathbf{x}}_i, \boldsymbol{\theta}_j) - \frac{\lambda}{2} \|\boldsymbol{\theta}_j\|_2^2, \tag{3.6}$$

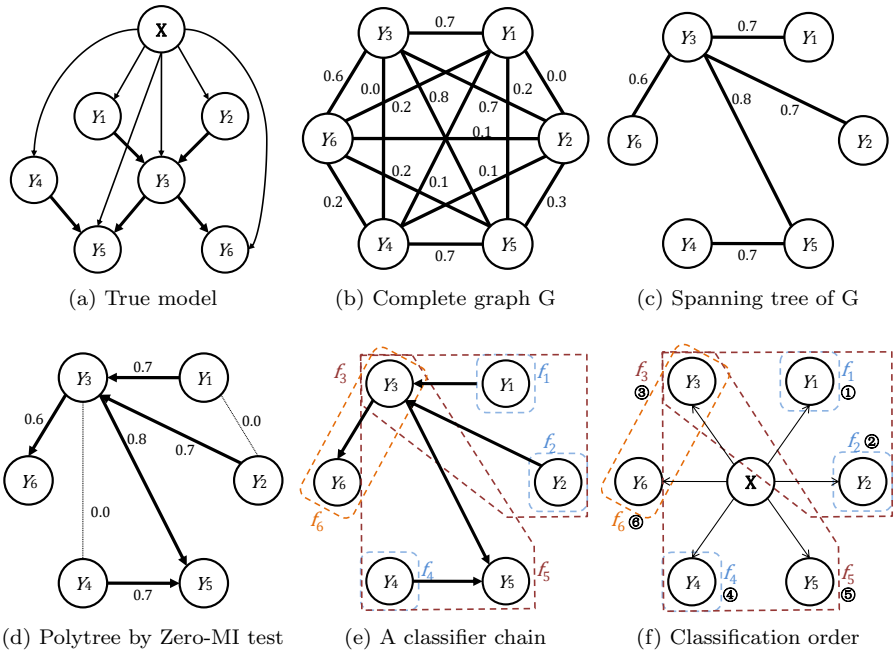
where λ is a trade-off coefficient to avoid overfitting by generating sparse parameters $\boldsymbol{\theta}_j$. Then traditional convex optimization techniques, such as Quasi-Newton method with BFGS iteration [54], can be used to learn the parameters.

Classification

Exact inference in the prediction phase is NP-hard in directed acyclic graphs. However, in polytrees, using the max-sum algorithm [63], we can make exact/exhaustive inference in a reasonable time by bounding the indegree of nodes.

Two phases are performed in order. The first phase, we begin at the root(s) and propagates testing downward to the leaves. The conditional probability table for each node is calculated on the basis of its local graphical structure. In the second phase, message propagation starts upward from the leaves to the root(s). In each node Y_j, we collect all the incoming messages and finding the local maximum with its value $\hat{y}_j$. In this way, we have the *Maximum a Posteriori* (MAP) estimate $\hat{\mathbf{y}} = (\hat{y}_1, ..., \hat{y}_L)$ such as

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \in \mathcal{Y}} \prod_{j=1}^{L} f_j(\mathbf{x}, \hat{\mathbf{pa}}_j) = \arg\max_{1,...,y_r,y_l,...,y_L} \Big[f_r(\mathbf{x}) \big[\cdots f_l(\hat{\mathbf{pa}}_l, \mathbf{x}) \big] \Big], \tag{3.7}$$



(a) True model (b) Complete graph G (c) Spanning tree of G

(d) Polytree by Zero-MI test (e) A classifier chain (f) Classification order

Figure 3.3: Learning (b-e) and prediction (f) phases of PACC. The true but hidden graphical model (a) is learned from data. (b) Construct a complete graph G with edges weighted by the mutual information **MI**. (c) Construct a spanning tree in G. (d) Make directions by Zero-MI testing. (e) Train six probabilistic classifiers f_1-f_6. (f) Prediction is made in the order of circled numbers.

where Y_l represents a leaf label and Y_r is a root label, respectively.

An example of learning and prediction in PACC is shown in Fig. 3.3. The pseudo code of PACC is depicted in Algorithm 1.

3.2 Label-Dependent Features (LDF) Selection

A two-stage feature selection approach consisting of classifier-independent *filter* and classifier-dependent *wrapper* has been recommended [47] to gain a good trade-off between classification performance and computation time. Motivated by this study, we developed a two-stage feature selection approach for CC methods based on a simple filter algorithm to find label-dependent, equivalently class-dependent features [3] and save label correlations during feature selection. In this way, we expect the proposed approach to improve classification performance and reduce the computational complexity in both learning and prediction phases. According to whether features are evaluated individually or not, the

Algorithm 1 The algorithm of PACC

Input: $\mathcal{D}$: training set, $\mathcal{T}$: test set, $\mathbf{f} = \{f_j\}_{j=1}^{L}$: multi-label probabilistic classifier

Output: $\hat{\mathbf{y}}$: prediction on a test instance $\hat{\mathbf{x}}$, $\hat{\mathbf{x}} \in \mathcal{T}$

1: Transform $\mathcal{D}$ into $\{\mathcal{D}_j\}_{j=1}^{L}$, where $\mathcal{D}_j = \{\mathbf{x}_i, y_{ij}\}_{i=1}^{N}$;
2: Calculate the mutual information matrix $\mathbf{MI} = (I_{jk})_{L\times L}$ by (3.3);
3: Construct a polytree $B = \{\boldsymbol{Pa}_j\}_{j=1}^{L}$ on $\mathbf{MI}$, and form the **chain**;
4: Transform $\{\mathcal{D}_j\}_{j=1}^{L}$ into $\{\mathcal{D}_j^{+}\}_{j=1}^{L}$ based on B, where $\mathcal{D}_j^{+} = \{\mathbf{x}_i \cup \mathbf{pa}_{ij}, y_{ij}\}_{i=1}^{N}$;
5: **for** $j \in$ **chain do**
6: Learn a probabilistic classifier f_j on $\mathcal{D}_j^{+}$ according to (3.5) and (3.6);
7: **for** $\hat{\mathbf{x}} \in \mathcal{T}$ **do**
8: Return $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}} \prod_{j=1}^{L} f_j(\hat{\mathbf{x}}, \hat{\mathbf{pa}}_j)$ according to (3.7);

existing filter algorithms can be categorized into two groups: feature weighting algorithms and subset search algorithms [106]. Feature weighting algorithms evaluate the weights of features individually, and rank them by the relevance to the target class. It is quite efficient to remove irrelevant features, but totally ignores the correlations between features. On the other hand, redundant features that are strongly correlated with others also harm the performance of the learning algorithm [46]. Subset search algorithms aim to overcome such limitations, and still maintain a reasonable time complexity compared with the wrapper algorithm. The subset search algorithm searches through candidate feature subsets guided by a certain evaluation measure that captures the goodness of each subset [55]. In this study, we propose a two-stage approach by using both feature weighting and subset search in order to select label-dependent features.

3.2.1 LDF Weighting

In the first stage, we develop a novel Multi-Label Information Gain (MLIG) algorithm based on feature weighting to efficiently remove irrelevant features for each label. IG has been frequently used as an evaluation criterion for feature weighting in various machine learning tasks [103]. Given a label variable Y_j and a feature variable X_k, IG measures the amount of the entropy of Y_j reduced by knowing X_k:

$$\begin{aligned} I(Y_j; X_K) &= H(Y_j) - H(Y_j|X_k) \\ &= -\sum_{y_j\in\{0,1\}} P(y_j)\log P(y_j) + \sum_{x_k\in\mathcal{V}_k} P(x_k) \sum_{y_j\in\{0,1\}} P(y_j|x_k)\log P(y_j|x_k), \end{aligned} \tag{3.8}$$

where $\mathcal{V}_k$ denotes the value space of the feature variable X_k. In practice, the numeric features should be discretized beforehand for computational efficiency. For the multi-label datasets, a straightforward way to apply IG is to rank all the features for each label according to (3.8), and then select the top-ranked features to feed the postprocess.

However, it is nontrivial to choose an appropriate threshold for filtering out irrelevant features. In addition, in the MLC setting, it is unreasonable to set the same threshold for all labels due to the label imbalance problem. For the labels with higher imbalance ratio, the number of positive instances may be insufficient for building an accurate classifier, in which case a smaller number of features should be chosen. To overcome the problem, in MLIG we set the percentage α_j of selected features for the label variable Y_j according to

$$\alpha_j = 2r \cdot \frac{e^{\beta_j} - 1}{e^{\beta_j} + 1} + r, \quad \beta_j = \frac{\overline{\mathrm{IR}}(\mathcal{D})}{\mathrm{IR}(Y_j) \cdot (\mathrm{CVIR}(\mathcal{D}) + 1)}, \tag{3.9}$$

where r is a factor controlling the range of α_j so that $\alpha_j \in [r, 3r]$. To depict the imbalance level of a dataset $\mathcal{D}$, the mean of the Imbalance Ratio (IR) and the Coefficient of Variation of IR (CVIR) [13] are defined as follows,

$$\overline{\mathrm{IR}}(\mathcal{D}) = \frac{1}{L}\sum_{j=1}^{L} \mathrm{IR}(Y_j), \quad \mathrm{CVIR}(\mathcal{D}) = \frac{1}{\overline{\mathrm{IR}}(\mathcal{D})}\sqrt{\sum_{j=1}^{L} \frac{(\mathrm{IR}(Y_j) - \overline{\mathrm{IR}}(\mathcal{D}))^2}{L-1}}, \tag{3.10}$$

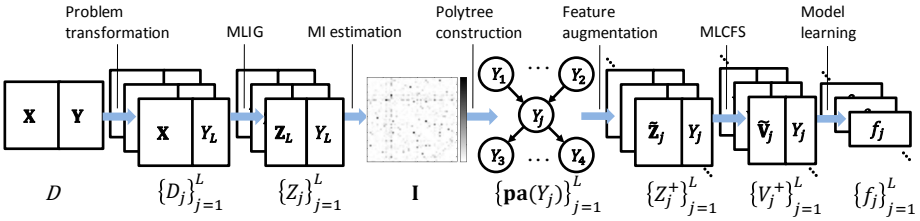
where $\mathrm{IR}(Y_j) = \max_k \sum_{i=1}^{N} \mathbb{1}_{y_{ik}=1} / \sum_{i=1}^{N} \mathbb{1}_{y_{ij}=1}$. We can see that multi-label datasets are highly imbalanced from the values of these two measures in [13]. According to (3.9), we can see that the value of α_j is close to $3r$ for the majority labels in well-balanced datasets, and α_j becomes r for the minority labels in highly imbalanced datasets. As a result, a smaller number of features is selected for each minority label in an imbalanced dataset, and vice versa.

In this way, MLIG first calculates a feature-label information gain matrix according to (3.8), then rank the features for each label and select most relevant label-dependent features up to $m_j = \alpha_j M$, $j = 1, ..., L$. Finally, we transform the original data $\mathcal{D}_j = \{(\mathbf{x}_i, y_{ij})\}_{i=1}^{N}$ into $\mathcal{Z}_j = \{(\mathbf{z}_{ij}, y_{ij})\}_{i=1}^{N}$, $\mathbf{z}_j \in \mathbb{R}^{m_j}$.

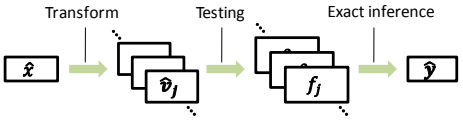
3.2.2 LDF Subset Selection

Although the MLIG approach works for feature selection to some extent, it is unable to eliminate redundant features. Thus, we consider a feature subset selection algorithm to find a more compact feature subset by incorporating the label dependency modeled by the polytree structure.

In the second stage, we extend the Correlation-based Feature Selection (CFS) [33], one of the subset search algorithms. We apply CFS after construction of the polytree $B = \{\boldsymbol{Pa}_j\}_{j=1}^{L}$. In the proposed Multi-Label CFS



(a) Learning phase of PACC-LDF



(b) Prediction phase of PACC-LDF

Figure 3.4: Flow chart of the proposed PACC-LDF method. The learning phase (a) consists of seven steps: problem transformation, MLIG, MI estimation, polytree construction, feature augmentation, MLCFS and model learning. The prediction phase (b) consists of three steps: instance transform, testing and exact inference.

(MLCFS), we conduct the traditional CFS individually for each label, taking label correlations modeled by B into account. More specifically, given a label variable Y_j with its dataset $\mathcal{Z}_j^+ = \{\tilde{\mathbf{z}}_{ij}, y_{ij}\}_{i=1}^N$, where $\tilde{\mathbf{z}}_j = \mathbf{z}_j \cup \mathbf{pa}_j$. The merit of a feature subset S_j in size of $\tilde{n}_j$ ($\tilde{n}_j = n_j + |\boldsymbol{Pa}_j|$) features is evaluated as

$$\text{Merit}(S_j) = \frac{\tilde{n}_j \rho_{Y_j \tilde{Z}}}{\sqrt{\tilde{n}_j + \tilde{n}_j(\tilde{n}_j - 1)\rho_{\tilde{Z}\tilde{Z}}}}, \tag{3.11}$$

where the mean correlations $\rho_{Y_j \tilde{Z}}$ and $\rho_{\tilde{Z}\tilde{Z}}$ are calculated according to

$$\rho_{Y_j \tilde{Z}} = \frac{2}{\tilde{n}_j} \sum_{k=1}^{\tilde{n}_j} \frac{I(\tilde{Y}_j; \tilde{Z}_k)}{H(\tilde{Y}_j) + H(\tilde{Z}_k)}, \quad \rho_{\tilde{Z}\tilde{Z}} = \frac{4}{\tilde{n}_j(\tilde{n}_j - 1)} \sum_{\substack{k,l=1 \\ k \neq l}}^{\tilde{n}_j} \frac{I(\tilde{Z}_k; \tilde{Z}_l)}{H(\tilde{Z}_k) + H(\tilde{Z}_l)}. \tag{3.12}$$

MLCFS first calculates a feature-feature and feature-label correlation matrix, and then employs a heuristic search algorithm, such as Best First [46], with the start set $\boldsymbol{Pa}_j$ to search the feature subset of space Y_j by maximizing (3.11) in reasonable time. In this way, the dimensionality of the feature space is reduced from m_j to n_j, typically $n_j \ll m_j$. We transform the data $\mathcal{Z}_j^+ = \{\tilde{\mathbf{z}}_{ij}, y_{ij}\}_{i=1}^N$ into $\mathcal{V}_j^+ = \{\tilde{\mathbf{v}}_{ij}, y_{ij}\}_{i=1}^N$, where $\tilde{\mathbf{v}}_j = (\mathbf{v}_j, \mathbf{pa}_j)$, $\mathbf{v}_j \in \mathbb{R}^{n_j}$. Finally, $\mathcal{V}_j^+$ is used to learn the probabilistic classifier f_j.

Algorithm 2 gives the pseudo code of PACC with Label-Dependent Features, named PACC-LDF. PACC-LDF first performs problem transformation in Step 1, follows MLIG to remove irrelevant features and transform the training set $\{\mathcal{D}_j\}$ into $\{\mathcal{Z}_j\}$ from Steps 2 to 4. Then a polytree B is built on $\{\mathcal{Z}_j\}$ from

Steps 5 to 6, and $\{\mathcal{Z}_j\}$ is transformed into $\{\mathcal{Z}_j^+\}$ in Step 7 based on B. After that, MLCFS is performed on $\{\mathcal{Z}_j^+\}$, and $\{\mathcal{Z}_j^+\}$ is further transformed into $\{\mathcal{V}_j^+\}$ in Steps 8 to 10. Based on the set $\{\mathcal{V}_j^+\}$ with label-dependent features, a multi-label probabilistic classifier $\{f_j\}$ is learned at Step 11. Finally, a test set $\mathcal{T}$ is transformed into the same lower-dimensional feature space, and given the predicted label subsets in Steps 12 to 15. Fig. 3.4 shows the framework of PACC-LDF.

Algorithm 2 The algorithm of PACC-LDF

Input: $\mathcal{D}$: training set, $\mathcal{T}$: test set, $\mathbf{f} = \{f_j\}_{j=1}^L$: multi-label probabilistic classifier

Output: $\hat{\mathbf{y}}$: prediction on a test instance $\hat{\mathbf{x}}$, $\hat{\mathbf{x}} \in \mathcal{T}$

1: Transform $\mathcal{D}$ into $\{\mathcal{D}_j\}_{j=1}^L$, where $\mathcal{D}_j = \{\mathbf{x}_i, y_{ij}\}_{i=1}^N$;
2: **for** $j = 1$ to L **do**
3: Perform MLIG on $\mathcal{D}_j$ according to (3.8), i.e., $g'_j : \mathbf{x}|_{M\times 1} \mapsto \mathbf{z}_j|_{m_j\times 1}$;
4: Transform $\mathcal{D}_j$ into $\mathcal{Z}_j = \{(\mathbf{z}_{ij}, y_{ij})\}_{i=1}^N$, where $\mathbf{z}_j = g'_j(\mathbf{x})$;
5: Calculate the mutual information matrix $\mathbf{MI} = \{I_{jk}\}_{L\times L}$, where I_{jk} is computed according to (3.3) based on $\mathcal{Z}_j$ and $\mathcal{Z}_k$;
6: Construct a polytree $B = \{\boldsymbol{Pa}_j\}_{j=1}^L$ on **MI**, and form the **chain**;
7: Transform $\{\mathcal{Z}_j\}_{j=1}^L$ into $\{\mathcal{Z}_j^+\}_{j=1}^L$ based on B, where $\mathcal{Z}_j^+ = \{\mathbf{z}_{ij} \cup \mathbf{pa}_{ij}, y_{ij}\}_{i=1}^N$
8: **for** $j \in$ **chain do**
9: Conduct MLCFS on $\mathcal{Z}_j^+$ by Best First search with the start set $\boldsymbol{Pa}_j$ according to (3.11), i.e., $g''_j : \mathbf{z}_j|_{m_j\times 1} \mapsto \mathbf{v}_j|_{n_j\times 1}$;
10: Transform $\mathcal{Z}_j^+$ into $\mathcal{V}_j^+ = \{(\mathbf{v}_{ij} \cup \mathbf{pa}_j, y_{ij})\}_{i=1}^N$, where $\mathbf{v}_j = g''_j(\mathbf{z}_j)$;
11: Learn a probabilistic classifier f_j on $\mathcal{V}_j^+$ according to (3.5) and (3.6);
12: **for** $\hat{\mathbf{x}} \in \mathcal{T}$ **do**
13: **for** $j \in$ **chain do**
14: Transform $\hat{\mathbf{x}}_j$ into $\hat{\mathbf{v}}_j$, i.e., $\hat{\mathbf{v}}_j = (g''_j \circ g'_j)(\hat{\mathbf{x}}_j)$;
15: Return $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}} \prod_{j=1}^L f_j(\hat{\mathbf{v}}_j, \hat{\mathbf{pa}}_j)$ according to (3.7).

PACC-LDF can be considered a general version of PACC, since PACC-LDF selects label-dependent features during the model building of PACC to improve its performance and reduce time complexity. By applying only one stage of the proposed feature selection approach, we can have two variants of PACC-LDF: PACC with MLIG (PACC-MLIG) (with Steps 2–4 only) and PACC with MLCFS (PACC-MLCFS) (with Steps 8–10 only). Note that the two-stage feature selection can also be applied to other MLC methods. For instance, it could be directly incorporated with BR by removing Steps 5–7 from Algorithm 2, leading to BR-LDF. For CC-based methods, we can do this by changing the content of $\boldsymbol{Pa}_j$, producing CC-LDF and BCC-LDF.

Table 3.1: Statistics of six synthetic multi-label datasets.

Dataset	N	$M^{\dagger}$	L	LCard	LDen	LDist	IR($\mathcal{D}$)	CVIR($\mathcal{D}$)
Data40-10	500	40	10	1.260	0.126	57	2.235	0.400
Data80-10	500	80	10	1.182	0.118	53	1.466	0.400
Data120-20	1000	120	20	1.404	0.070	227	1.810	0.368
Data160-20	1000	160	20	1.378	0.069	219	1.698	0.354
Data400-60	2000	400	60	2.187	0.036	1302	1.444	0.214
Data800-60	2000	800	60	2.147	0.036	1302	1.478	0.251

† The ratio of relevant, irrelevant and redundant features is $2:1:1$.

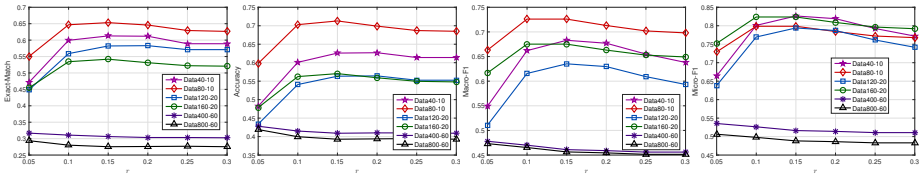
3.3 Experimental Results

3.3.1 Experimental Results on PACC-LDF

The methods used in the experiments were implemented based on Mulan [95] and Meka [71], and performed on six synthetic datasets and 12 benchmark datasets. To evaluate the classification performance, 5-fold and 3-fold cross validation was used for the eight regular-scale and four large-scale datasets, respectively. In the experiments, we chose logistic regression with ℓ_2 regularization as the baseline classifier, and set the constant value $\lambda = 0.1$ for the trade-off parameter λ in (3.6) for all MLC methods. To reduce the training cost, normalized mutual information instead of conditional mutual information (3.3) was calculated for large-scale datasets. The experiments were conducted on a computer configured with an Intel Quad-Core i7-4770 CPU at 3.4 GHz with 4G RAM.

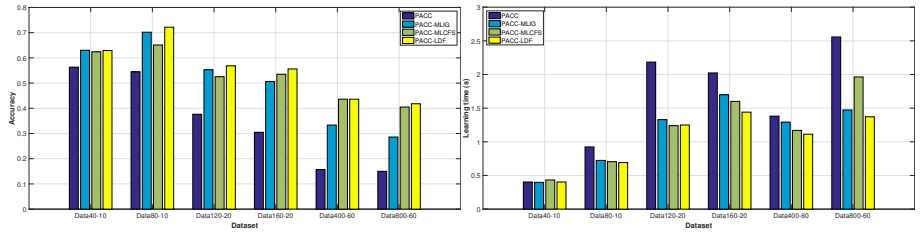
Synthetic datasets

In this section, we conduct experiments on six synthetic multi-label datasets to evaluate the performance of PACC with its three variants, PACC-MLIG, PACC-MLCFS, and PACC-LDF. In total, six synthetic datasets, including four regular-scale and two large-scale sets, were generated according to the method in [91]. In each dataset, instances were produced by randomly sampling from R hypercubes (labels) in the M-dimensional feature space, and thus the dataset is represented by Data$M$-R. The M-dimensional features consisted of three parts: relevant features, irrelevant features, and redundant features. The irrelevant features were randomly generated, and the redundant features were the copies of existing relevant features. In addition, to simulate real-world multi-label data, classification noise was added into these synthetic datasets, which flips the value of each label for an instance in a random manner with a probability



(a) Exact-Match (b) Accuracy (c) Macro-F1 (d) Micro-F1

Figure 3.5: The performance of PACC-LDF according to four evaluation metrics on six synthetic datasets by varying the value of r from 0.05 to 0.3 in steps of 0.05.



(a) Accuracy (b) Learning time

Figure 3.6: The performance of PACC and its three variants in Accuracy and Learning time (in seconds) on six synthetic datasets.

of 0.02. The statistics of the synthetic datasets are reported in Table 3.1.

First, we performed experiments on PACC-LDF by changing the value of factor r in (3.9), which controls the lower and upper bounds of α_j by $r \leq \alpha_j \leq 3r$ according to (3.9). Experimental results according to four evaluation metrics are shown in Fig. 3.5, from which we can reach two conclusions. (1) In the regular-sized datasets, the performance of PACC-LDF is worse for a small value of r, $r < 0.1$, but improves and becomes stable as r exceeds 0.15. (2) In the large-sized sets, PACC-LDF performs better when the value of r is small, and works slightly worse if r exceeds 0.15.

In Fig. 3.6, the performance of PACC, PACC-MLIG, PACC-MLCFS, and PACC-LDF in Accuracy and Learning time is reported. Note that we do not show the performance in other metrics here, since similar results and patterns can be observed. The proposed LDF and its variants significantly improve the performance of PACC. Specifically, PACC-LDF works best among the four methods, and achieves a performance improvement of at least 10% compared with the original PACC, indicating the effectiveness of the proposed two-stage feature selection approach. In terms of learning time, PACC-LDF consumed the least time on the last five datasets. In terms of testing time, all methods consumed a similar time on regular-scale datasets, but PACC-LDF cost the least

time on the two large-scale datasets. Therefore, the two-stage feature selection approach, LDF, rather than MLIG or MLCFS, is employed in the following experiments.

Real-world datasets

Next, we evaluate the performance of popular MLC methods on the 12 real-world benchmark multi-label datasets in Table 2.1. This part of the experiment is composed of three major parts. In the first part, we compare PACC with three CC-based methods, namely BR, CC, and BCC, to demonstrate the effectiveness of the polytree structure on capturing label correlations. In the second part, PACC-LDF is compared with three state-of-the-art MLC methods in terms of classification accuracy and execution time. In the last part, CC-based methods are compared with their LDF variants in a pairwise way to evaluate the performance of the two-stage feature selection approach. In addition, the comparing results of LDF with traditional feature selection algorithms are presented. The MLC methods used in this section are summarized as follows.

- CC-based methods were introduced in Section 3. In CC, the chain is established in a randomly determined order. In BCC, the normalized mutual information is used for marginal dependency estimation on each label pair, since the performance could be slightly improved without consuming extra processing time.

- Multi-Label k-Nearest Neighbors (MLkNN) [112] originates from the traditional k-nearest neighbors algorithm. For each test instance, according to the label assignments of its k-nearest neighbors in the training set, the prediction is made on the basis of the MAP principal. In the experiments, we set $k = 10$, by following the suggestion in the literature [112].

- RAndom k LabELsets (RAkEL) [94] is an ensemble variant of the Label Combination (LC) method. RAkEL transforms an MLC problem into a set of smaller MLC problems, by training m LC models using random k-subsets of the original label set. To make it executable in a limited time cost (24 h), RAkEL employed the C4.5 decision tree as its baseline single-label classifier for large-scale datasets. We set $k = 3$ and $m = 2L$ as recommended in [94].

- By building HOMER [93] on the basis of balanced k-means clustering, HOMER reduces the complexity of prediction and addresses the label imbalance problem. According to the experimental results in [93], the number k of clusters for building the hierarchical structure was set to 4. In addition, BR with ℓ_2 regularized logistic regression was used as its baseline multi-label classifier.

Table 3.2: Experimental results (mean±std) of comparing CC-based methods on 12 multi-label datasets in terms of four evaluation metrics.

Method	Exact-match											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
BR	.239±.069	.466±.024	.139±.015	**.476±.060**	.949±.030	.580±.046	.089±.017	.224±.032	.047±.003	.006±.001	.110±.005	.003±.002
CC	.260±.044	.546±.017	**.194±.016**	**.476±.056**	.974±.009	**.586±.040**	.099±.011	**.237±.023**	.111±.007	.012±.001	.120±.006	**.009±.001**
BCC	.239±.050	.494±.028	.136±.012	**.476±.064**	**.977±.007**	.575±.042	.095±.017	.234±.018	.059±.004	.007±.001	.108±.009	.003±.001
PACC	**.261±.030**	**.568±.019**	.125±.079	.474±.057	.974±.009	**.586±.033**	**.112±.025**	**.237±.025**	**.113±.008**	**.016±.003**	**.128±.002**	**.009±.004**
Method	**Accuracy**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
BR	.511±.049	.567±.020	**.512±.018**	.576±.055	.973±.017	**.698±.029**	.377±.015	**.315±.020**	.243±.003	**.117±.006**	.287±.001	.097±.003
CC	.516±.038	.608±.020	.505±.013	.579±.051	.986±.006	.672±.028	.380±.009	.275±.012	.289±.003	.093±.001	**.299±.003**	.116±.001
BCC	.502±.035	.582±.028	.494±.014	**.580±.062**	**.987±.006**	.656±.028	.369±.019	.273±.008	.261±.002	.073±.006	.288±.005	.096±.002
PACC	**.523±.033**	**.625±.018**	.400±.131	**.580±.055**	.986±.006	.668±.027	**.384±.017**	.277±.015	**.294±.011**	.101±.005	.295±.002	**.118±.009**
Method	**Macro-F1**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
BR	**.636±.043**	.652±.019	**.424±.014**	.335±.028	.725±.055	**.372±.022**	**.216±.026**	**.090±.016**	.232±.008	**.093±.007**	.297±.008	**.045±.003**
CC	.623±.035	.639±.021	.404±.019	.341±.013	.740±.050	.355±.025	.208±.028	.055±.009	**.240±.013**	.070±.002	**.308±.009**	.043±.004
BCC	.620±.029	.649±.019	.394±.007	.345±.028	**.742±.049**	.352±.024	.209±.024	.056±.009	**.240±.010**	.073±.005	.299±.009	.044±.004
PACC	.632±.034	**.657±.017**	.351±.064	**.349±.019**	.740±.050	.354±.023	.206±.020	.057±.007	**.240±.010**	.076±.001	.302±.002	.043±.004
Method	**Micro-F1**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
BR	**.650±.042**	.645±.019	**.642±.015**	.441±.048	.977±.015	**.761±.020**	**.497±.010**	**.272±.015**	.344±.003	**.194±.008**	.376±.008	.164±.002
CC	.633±.034	.631±.022	.626±.009	.449±.030	.989±.004	.750±.015	**.497±.006**	.225±.024	**.369±.001**	.142±.002	.385±.007	**.178±.002**
BCC	.632±.027	.642±.021	.629±.012	.449±.048	.990±.004	.742±.017	.489±.015	.223±.027	.360±.002	.130±.008	.371±.008	.162±.004
PACC	.642±.032	**.647±.018**	.515±.146	**.454±.037**	.989±.004	.748±.015	.492±.009	.231±.022	**.369±.012**	.154±.001	**.389±.002**	.165±.010

The results of the CC-based methods are summarized in Table 3.2. In terms of instance-based evaluation metrics, Exact-Match and Accuracy, PACC was the best or competitive with the best methods on all experimental datasets, except the Yeast dataset. This is understandable because PACC is a subset 0-1 risk minimizer benefiting from its ability on the polytree structure as well as exact inference. In these metrics, CC is the second best, while BCC is the third in most cases. This is probably because BCC models only label pairwise correlations. Consistent with our theoretical analysis, BR obtains the worst result in Exact-Match because it ignores label correlations. It is also worth noting that BR works better than CC-related methods only on the Birds set, indicating weak label correlations in that set. In Macro-F1/Micro-F1, BR and BCC obtained competitive results with CC and PACC. This is likely because the label-based evaluation metrics place greater emphasize on the performance on the individual label. Indeed BR is actually the hamming-loss risk minimizer and BCC only models the most important label pairwise dependency.

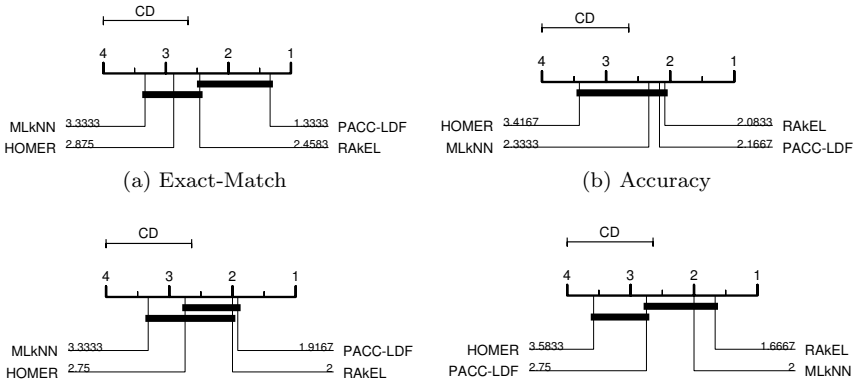
Next, PACC-LDF was compared with three popular MLC methods. The experimental results are shown in Table 3.3. From Table 3.3, we can see that PACC-LDF is the best in Exact-Match, and competitive with RAkEL and MLkNN in Accuracy and Macro-F1. In Accuracy and Micro-F1, RAkEL works best, followed by MLkNN and PACC-LDF. To compare the performance of multiple methods on multiple datasets, we conducted the Friedman test [21] aiming to reject the null hypothesis as equal performance among the comparing methods. Furthermore, since the null hypothesis is rejected by the Friedman test according to all the metrics (the values of statistic F_F of Exact-Match,

Table 3.3: Experimental results (mean±std) of comparing PACC-LDF with three state-of-the-art methods on 12 multi-label datasets in terms of four evaluation metrics.

Method	**Exact-match**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
ML*k*NN	.302±.035	**.638±.017**	**.192±.018**	.433±.028	.921±.020	.496±.049	.004±.004	.195±.030	.032±.006	.002±.000	.057±.008	.000±.000
RA*k*EL	.285±.040	.535±.025	.156±.016	.471±.067	.965±.023	.659±.044	.108±.025	.199±.023	.052±.007	.009±.002	.114±.007	.003±.001
HOMER	.223±.039	.476±.041	.136±.009	.457±.056	.950±.003	.544±.024	.091±.019	.214±.023	.054±.015	.011±.003	.062±.003	.003±.001
PACC-LDF	**.314±.046**	.595±.010	.125±.080	**.513±.047**	**.970±.012**	**.679±.019**	**.149±.018**	**.240±.030**	**.145±.013**	**.018±.004**	**.161±.008**	**.014±.004**
Method	**Accuracy**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
ML*k*NN	**.568±.033**	**.717±.014**	**.545±.019**	.542±.042	.964±.011	.632±.037	.369±.016	.255±.028	**.283±.009**	**.147±.005**	.188±.009	**.139±.005**
RA*k*EL	.550±.037	.617±.022	.515±.015	.576±.068	**.983±.011**	**.758±.035**	**.460±.023**	**.278±.019**	.277±.004	.107±.003	**.307±.007**	.083±.003
HOMER	.511±.034	.565±.033	.504±.016	.559±.056	.974±.015	.644±.014	.350±.024	.262±.018	.232±.019	.124±.002	.182±.004	.085±.005
PACC-LDF	.546±.033	.632±.007	.394±.146	**.584±.046**	**.983±.010**	**.758±.018**	.395±.017	.271±.018	.274±.012	.130±.020	.296±.012	.111±.011
Method	**Macro-F1**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
ML*k*NN	**.656±.025**	**.759±.013**	.426±.012	.300±.055	.619±.077	.241±.014	.126±.010	.052±.016	.183±.004	.051±.005	.178±.005	.026±.003
RA*k*EL	.654±.032	.673±.015	**.441±.012**	.343±.042	.736±.055	**.372±.043**	.176±.008	.063±.010	.202±.003	.048±.004	**.309±.004**	.024±.001
HOMER	.634±.029	.632±.025	.415±.013	.332±.041	.687±.049	.307±.028	**.178±.017**	.053±.017	.190±.002	.062±.004	.192±.002	.035±.001
PACC-LDF	.651±.039	.651±.011	.303±.054	**.358±.036**	**.740±.051**	.360±.019	.177±.012	**.068±.018**	**.209±.006**	.056±.005	.258±.015	**.040±.006**
Method	**Micro-F1**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
ML*k*NN	**.683±.027**	**.751±.011**	**.669±.015**	.416±.054	.957±.011	.697±.036	.531±.014	.186±.023	**.405±.006**	**.233±.008**	.296±.003	**.225±.007**
RA*k*EL	.665±.033	.665±.016	.642±.012	**.457±.063**	.986±.009	.812±.029	**.585±.009**	**.237±.009**	.389±.002	.181±.003	**.407±.007**	.151±.004
HOMER	.644±.028	.630±.027	.635±.017	.427±.044	.970±.019	.680±.012	.454±.019	.170±.033	.318±.012	.178±.002	.249±.004	.132±.006
PACC-LDF	.659±.033	.644±.008	.506±.161	.406±.051	**.988±.005**	**.805±.019**	.492±.018	.209±.037	.350±.011	.147±.023	.401±.016	.167±.015

Accuracy, Macro-F1 and Macro-F1 are 8.8995, 3.2960, 2.9237, and 8.5074, respectively, which are higher than the critical value of 2.8805 with a significance level of $\alpha = 0.05$); the Nemenyi test [60] was conducted for pairwise comparison in classification performance. According to [21], the performance of two methods is regarded as significantly different if their average ranks differ by at least the *critical difference* (CD). Figure 3.7 shows the CD diagrams for the four evaluation metrics at the 0.05 significance level. In each subfigure, the CD is given above the axis, where the averaged rank is marked. In Figure 3.7, algorithms that are not significantly different are connected by a thick line. The test result indicates that PACC-LDF performs significantly better than either ML*k*NN or HOMER in Exact-Match and Macro-F1. However, there was no significant difference between PACC-LDF and the other methods in Accuracy and Micro-F1.

Table 3.4 summarizes the learning and prediction times of eight comparing methods. Of all the methods, ML*k*NN needed the least training time due to its lazy strategy, while HOMER cost the least time in the prediction phase as it has sublinear time complexity with respect to the number of labels. RA*k*EL consumed the largest training time in all datasets except for the Medical dataset in spite of the fact that it employs a simple decision tree as its baseline classifier. The high complexity of RA*k*EL probably arises from its ensemble strategy and the LC models for modeling label correlations. For the CC-based methods, significant reduction in both learning and prediction times can be observed by employing LDF. Indeed, on average, 60% of features were removed in two



(a) Exact-Match (b) Accuracy

(c) Macro-F1 (d) Micro-F1

Figure 3.7: CD diagrams (0.05 significance level) of the four comparing methods according to four evaluation metrics. The performance of two methods is regarded as significantly different if their average ranks differ by at least the critical difference.

Table 3.4: Learning and prediction time (in seconds) of eight comparing methods on 12 datasets.

Method	**Learning time**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
ML*k*NN	0.680	12.594	5.506	1.148	3.412	**0.769**	**3.508**	**6.404**	110.673	356.076	**39.467**	**42.202**
RA*k*EL	3.541	62.611	40.115	117.877	9.650	49.955	693.616	559.051	3408.508	5954.040	7744.956	2035.276
HOMER	0.758	3.936	3.259	2.873	2.180	32.025	109.900	68.922	**78.163**	**95.862**	670.111	302.119
BR	0.264	2.623	2.774	2.468	2.726	157.696	118.021	161.566	217.546	523.935	1078.550	779.548
CC	**0.260**	**2.554**	2.580	2.346	**1.819**	167.315	145.068	175.858	188.427	273.912	1231.735	311.401
BCC	0.339	3.736	2.884	2.843	1.915	157.834	129.547	168.368	190.946	254.457	1068.095	211.065
PACC	0.406	3.742	2.935	2.791	1.969	164.628	128.109	175.168	191.379	271.635	1151.284	299.956
PACC-LDF	0.430	4.074	**2.315**	**1.043**	2.403	3.763	12.224	40.733	180.480	810.881	199.482	2020.880
Method	**Prediction time**											
	Emotions	Scene	Yeast	Birds	Genbase	Medical	Enron	Languagelog	Rcv1s1	Corel16k1	Bibtex	Corel5k
ML*k*NN	0.050	2.833	1.131	0.192	0.660	**0.072**	0.734	1.420	54.888	174.226	18.619	18.501
RA*k*EL	0.007	0.050	0.087	0.046	0.109	0.680	1.268	1.944	4.305	**7.201**	33.380	8.660
HOMER	**0.003**	0.022	**0.028**	**0.012**	0.049	0.167	0.643	**0.339**	**1.853**	16.772	21.503	**3.462**
BR	0.006	**0.017**	0.045	0.041	0.170	0.477	1.297	1.605	18.280	89.277	48.485	205.449
CC	0.007	0.024	0.041	0.036	0.172	0.477	1.325	2.143	30.853	92.732	50.062	223.149
BCC	0.007	0.032	0.055	0.041	0.180	0.482	1.265	2.278	27.654	74.457	54.983	174.232
PACC	0.009	0.036	0.052	0.042	0.239	0.490	1.246	2.104	29.328	98.745	29.034	195.372
PACC-LDF	0.007	0.031	0.043	0.020	**0.030**	0.140	**0.373**	0.759	3.024	27.121	**15.845**	73.235

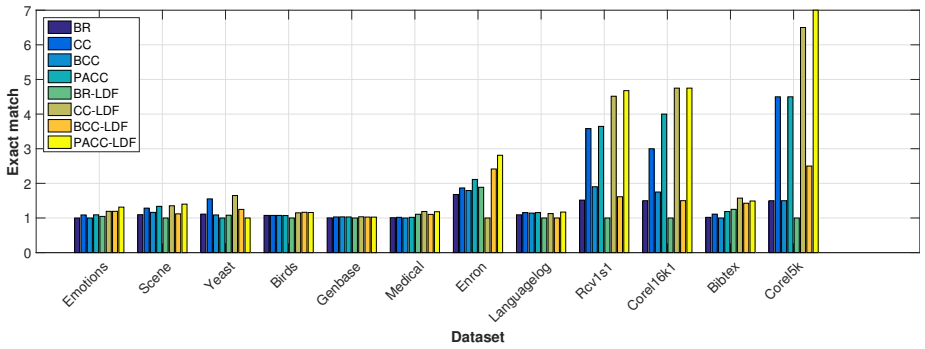


Figure 3.8: Comparison of CC-based methods with their LDF variants in Exact-Match. For each dataset, the values in Exact-Match have been normalized by dividing the lowest value in the dataset.

Table 3.5: Wilcoxon signed-ranks test with significance level $\alpha = 0.05$ for CC-based methods against their LDF variants according to four evaluation metrics (p_α-values are shown in parentheses); "win" denotes the existence of a significant difference.

Comparing methods	Exact-Match	Accuracy	Macro-F1	Micro-F1
BR-LDF vs. BR	**tie**[6.1e-1]	**tie**[6.4e-2]	**tie**[7.3e-1]	**win**[7.8e-3]
CC-LDF vs. CC	**win**[4.1e-2]	**tie**[6.4e-1]	**win**[8.3e-3]	**tie**[9.5e-1]
BCC-LDF vs. BCC	**win**[5.5e-2]	**tie**[3.3e-1]	**lose**[5.9e-3]	**tie**[2.3e-1]
PACC-LDF vs. PACC	**win**[4.9e-3]	**tie**[3.5e-1]	**tie**[1.8e-1]	**tie**[5.8e-1]

balanced datasets, Scene and Emotions, while at least 80% of features were eliminated in the other datasets, leading to a remarkable reduction in time complexity. However, PACC-LDF consumed more time in Corel16k1 and Corel5k than CC-based methods. This is likely because feature selection dominates the time complexity in these two datasets. In sum, PACC-LDF is a good choice for MLC when exact matching is expected and less execution time is demanded.

Results of feature selection

From Fig. 3.8, we can confirm the effectiveness of the proposed LDF selection approach. With respect to Exact-Match, the performance of CC-based methods was significantly improved in most of the datasets, especially in the large-scale datasets. For example, in the Corel5k dataset, PACC-LDF works over 40% better than PACC, and even 4 times better than BR, demonstrating the performance superiority of selecting LDF for such a large-scale dataset. According to Fig. 3.8, CC-based methods with LDF achieve an average performance improvement of 9.4% in Exact-Match, compared with the original

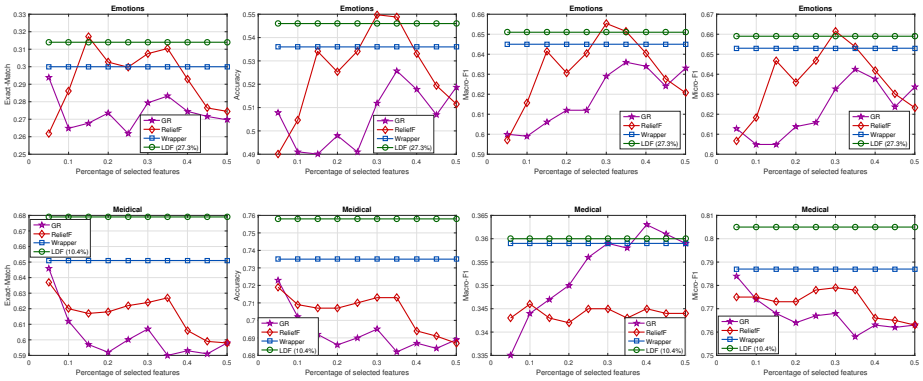


Figure 3.9: Comparison of LDF with three conventional feature selection algorithms on the Emotions (the top row) and Medical (the bottom row) datasets. The percentage of selected features increased from 0.05 to 0.5 in steps of 0.05. Note that Wrapper and LDF are independent of the percentage of features because Wrapper selects the feature subset that leads to the best performance, and LDF determines the number of label-specific features (on average, 27.3% for Emotions and 10.4% for Medical) by (3.9).

methods. The effectiveness of LDF is also confirmed by Table 3.5 with the results of the Wilcoxon signed-ranks test [21], which was conducted 16 times, each time on one CC-based method with its LDF counterpart. According to these results, all the LDF variants, except BR-LDF, outperform the original methods in Exact-Match, and obtain comparable results in the other evaluation metrics.

In addition, to demonstrate the effectiveness of the proposed LDF including the feature selection algorithm for LDF, we compared LDF with three feature selection approaches, Gain Ratio (GR) [40], ReliefF [45], and Wrapper [46], on the Emotions and Medical datasets. As a classifier, PACC with ℓ_2 regularized logistic regression was chosen. In these feature selection algorithms, backward greedy stepwise search is applied to find the relevant features for each label individually. To reduce the time cost of Wrapper, the top 50% (Emotions) and 10% (Medical) relevant features were selected by a filter algorithm [103], before applying the Wrapper algorithm [46]. The percentage of features increased from 0.05 to 0.5 in steps of 0.05. Fig. 3.9 shows the experimental results on the two datasets according to the four evaluation metrics. As shown in Fig. 3.9, LDF consistently works better than the other algorithms. Wrapper is the second best algorithm, and is competitive with LDF in Macro-F1. ReliefF performs better than GR in most cases, and is even comparable with LDF and Wrapper in some cases, but it is sensitive to the number of selected features. In terms of time complexity, ReliefF, GR, and LDF have similar time cost, while Wrapper needs hundreds more execution time than the other algorithms.

Chapter 4

Optimization of Classifier Chains via Conditional Likelihood Maximization

Although CC-based methods have achieved much success in various applications [69, 19, 107], further improvement in classification accuracy is still required. Here we seek the possibility to improve CC in terms of two aspects: label correlation modeling and multi-label feature selection. The intuition behind this idea is that all of the previously determined labels are not always necessary for decision on the current label (necessity of limiting label correlations), and irrelevant and redundant features are usually harmful for the performance of CC (necessity of feature selection). In this chapter, we propose a unified framework comprising of both label correlation modeling and multi-label feature selection via conditional likelihood maximization [83].

4.1 A Unified Framework for Multi-Label Classification

According to (2.2), the objective of MLC is actually to approximate the underlying conditional probability $p(\boldsymbol{Y}|\boldsymbol{X})$. In this paper, we assume that the underlying probability $p(\boldsymbol{Y}|\boldsymbol{X})$ can be approximated by a Bayesian network B of relatively simple structure with conditional probability $p_B(\boldsymbol{Y}|\boldsymbol{X})$, which optimally captures the label correlations. In addition, we further assume that $p_B(\boldsymbol{Y}|\boldsymbol{X})$ can be modeled by a subset of $\boldsymbol{X}$, i.e., the relevant features $\boldsymbol{X}_\theta \in \boldsymbol{X}$. A M-dimensional binary vector θ is adopted with 1 indicating the feature is selected and 0 otherwise. $\boldsymbol{X}_\theta/\boldsymbol{X}_{\bar{\theta}}$ denotes the selected/unselected feature sub-

set, and thus $\boldsymbol{X} = \{\boldsymbol{X}_\theta, \boldsymbol{X}_{\overline{\theta}}\}$. Hence we have $p_B(\boldsymbol{Y}|\boldsymbol{X}) = p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)$. Last, a predictive model $f(\boldsymbol{Y}|\boldsymbol{X}_\theta, \tau)$ is built to model $p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)$, where τ denotes the parameters of f used to predict $\mathbf{y}$. In this way, we find the optimal Bayesian network B, identify the relevant feature subset $\boldsymbol{X}_\theta$, and then build the predictive model f.

Given a sample of N observations $\mathcal{D} = \{(\mathbf{x}^i, \mathbf{y}^i)|i = 1, ..., N\}$ drawn from an underlying i.i.d. process $p : \boldsymbol{X} \to \boldsymbol{Y}$, the conditional likelihood $\mathcal{L}$ of the observations $\mathcal{D}$ given parameters $\{B, \theta, \tau\}$ becomes

$$\mathcal{L}(B, \theta, \tau|\mathcal{D}) = \prod_{i=1}^{N} f(\mathbf{y}^i|\mathbf{x}^i_\theta, \tau). \tag{4.1}$$

Therefore, our objective becomes

$$\{B^*, \theta^*, \tau^*\} = \arg\max_{B,\theta,\tau} \mathcal{L}(B, \theta, \tau|\mathcal{D}) = \arg\max_{B,\theta,\tau} \prod_{i=1}^{N} f(\mathbf{y}^i|\mathbf{x}^i_\theta, \tau). \tag{4.2}$$

For convenience, we use the average *log*-likelihood ℓ instead of $\mathcal{L}$:

$$\ell = \frac{1}{N} \sum_{i=1}^{N} \log f(\mathbf{y}^i|\mathbf{x}^i_\theta, \tau). \tag{4.3}$$

By taking $p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)$ into consideration, (4.3) is rewritten as

$$\ell = \frac{1}{N} \sum_{i=1}^{N} \log \left[\frac{f(\mathbf{y}^i|\mathbf{x}^i_\theta, \tau)}{p_B(\mathbf{y}^i|\mathbf{x}^i_\theta)} \cdot \frac{p_B(\mathbf{y}^i|\mathbf{x}^i_\theta)}{p_B(\mathbf{y}^i|\mathbf{x}^i)} \cdot \frac{p_B(\mathbf{y}^i|\mathbf{x}^i)}{p(\mathbf{y}^i|\mathbf{x}^i)} \cdot p(\mathbf{y}^i|\mathbf{x}^i) \right] \tag{4.4}$$

By the law of large numbers, ℓ approaches in probability the expected version

$$\mathbb{E}_{\boldsymbol{XY}} \left\{ \log \left[\frac{f(\boldsymbol{Y}|\boldsymbol{X}_\theta, \tau)}{p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)} \cdot \frac{p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)}{p_B(\boldsymbol{Y}|\boldsymbol{X})} \cdot \frac{p_B(\boldsymbol{Y}|\boldsymbol{X})}{p(\boldsymbol{Y}|\boldsymbol{X})} \cdot p(\boldsymbol{Y}|\boldsymbol{X}) \right] \right\} \tag{4.5}$$

We negate the above formula to minimize

$$\begin{aligned} -\ell \approx\ & \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)}{f(\boldsymbol{Y}|\boldsymbol{X}_\theta, \tau)} \right\} + \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p_B(\boldsymbol{Y}|\boldsymbol{X})}{p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)} \right\} \\ & + \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p(\boldsymbol{Y}|\boldsymbol{X})}{p_B(\boldsymbol{Y}|\boldsymbol{X})} \right\} - \mathbb{E}_{\boldsymbol{XY}} \left\{ \log p(\boldsymbol{Y}|\boldsymbol{X}) \right\} \end{aligned} \tag{4.6}$$

Since $\boldsymbol{X} = \{\boldsymbol{X}_\theta, \boldsymbol{X}_{\overline{\theta}}\}$, the second term can be developed as follows,

$$\begin{aligned} \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p_B(\boldsymbol{Y}|\boldsymbol{X})}{p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)} \right\} &= \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta, \boldsymbol{X}_{\overline{\theta}})}{p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)} \right\}, \\ &= \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p_B(\boldsymbol{X}_{\overline{\theta}}, \boldsymbol{Y}|\boldsymbol{X}_\theta)}{p_B(\boldsymbol{X}_{\overline{\theta}}|\boldsymbol{X}_\theta) p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)} \right\}, \\ &= I_{p_B}(\boldsymbol{X}_{\overline{\theta}}; \boldsymbol{Y}|\boldsymbol{X}_\theta), \end{aligned} \tag{4.7}$$

where $I(\boldsymbol{X};\boldsymbol{Y}|\boldsymbol{Z})$ denotes the mutual information between $\boldsymbol{X}$ and $\boldsymbol{Y}$ conditioned on $\boldsymbol{Z}$,

$$I(\boldsymbol{X};\boldsymbol{Y}|\boldsymbol{Z}) = \sum_{\mathbf{xyz}} p(\mathbf{x},\mathbf{y},\mathbf{z}) \log \frac{p(\mathbf{x},\mathbf{y}|\mathbf{z})}{p(\mathbf{x}|\mathbf{z})p(\mathbf{y}|\mathbf{z})} = \mathbb{E}_{\boldsymbol{XYZ}} \log \frac{p(\boldsymbol{X},\boldsymbol{Y}|\boldsymbol{Z})}{p(\boldsymbol{X}|\boldsymbol{Z})p(\boldsymbol{Y}|\boldsymbol{Z})}. \tag{4.8}$$

Using the K-L divergence [48]:

$$D_{KL}(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} = \mathbb{E}_{\boldsymbol{X}} \left\{ \log \frac{p(\boldsymbol{X})}{q(\boldsymbol{X})} \right\}, \tag{4.9}$$

(4.6) is finally rewritten as

$$-\ell \approx D_{KL}(p_B||f) + I_{p_B}(\boldsymbol{X}_{\overline{\theta}};\boldsymbol{Y}|\boldsymbol{X}_\theta) + D_{KL}(p||p_B) + H(\boldsymbol{Y}|\boldsymbol{X}) \tag{4.10}$$

From (4.10), our objective function can be decomposed into four different terms.

1. $D_{KL}(p_B||f)$: the K-L divergence between the conditional probability $p_B(\boldsymbol{Y}|\boldsymbol{X}_\theta)$ and the predictive model $f(\boldsymbol{Y}|\boldsymbol{X}_\theta,\tau)$, which measures how well f approximates p_B given the selected feature subset $\boldsymbol{X}_\theta$. This parameter τ could be optimized by the predefined predictive model given the optimized parameters B^* and θ^*,

$$\tau^* = \arg\min_{\tau} D_{KL}(p_{B^*}(\boldsymbol{Y}|\boldsymbol{X}_{\theta^*})||f(\boldsymbol{Y}|\boldsymbol{X}_{\theta^*},\tau)). \tag{4.11}$$

   Thus (4.11) is the *parameter selection* problem. Distinct predictive models would produce the different τ. It depends on our choice of the baseline model f.

2. $I_{p_B}(\boldsymbol{X}_{\overline{\theta}};\boldsymbol{Y}|\boldsymbol{X}_\theta)$: the mutual information between the unselected features $\boldsymbol{X}_{\overline{\theta}}$ and the labels $\boldsymbol{Y}$ conditioned on the selected features $\boldsymbol{X}_\theta$. This term depends on both the approximate Bayesian network B and the selected features $\boldsymbol{X}_\theta$. Given the optimized B^*, the optimal θ^* can be obtained as

$$\theta^* = \arg\min_{\theta} I_{p_{B^*}}(\boldsymbol{X}_{\overline{\theta}};\boldsymbol{Y}|\boldsymbol{X}_\theta). \tag{4.12}$$

   In fact (4.12) is the *multi-label feature selection* problem.

3. $D_{KL}(p||p_B)$: the K-L divergence between the underlying probability $p(\boldsymbol{Y}|\boldsymbol{X})$ and the approximate probability $p_B(\boldsymbol{Y}|\boldsymbol{X})$ modeled by a Bayesian network B, which measures how well p_B approximate p. This term depends only on the Bayesian network B, hence we have

$$B^* = \arg\min_{B} D_{KL}(p(\boldsymbol{Y}|\boldsymbol{X})||p_B(\boldsymbol{Y}|\boldsymbol{X})). \tag{4.13}$$

   Note that (4.13) is actually the *model selection* problem, which aims to find the optimal Bayesian network to capture label correlations.

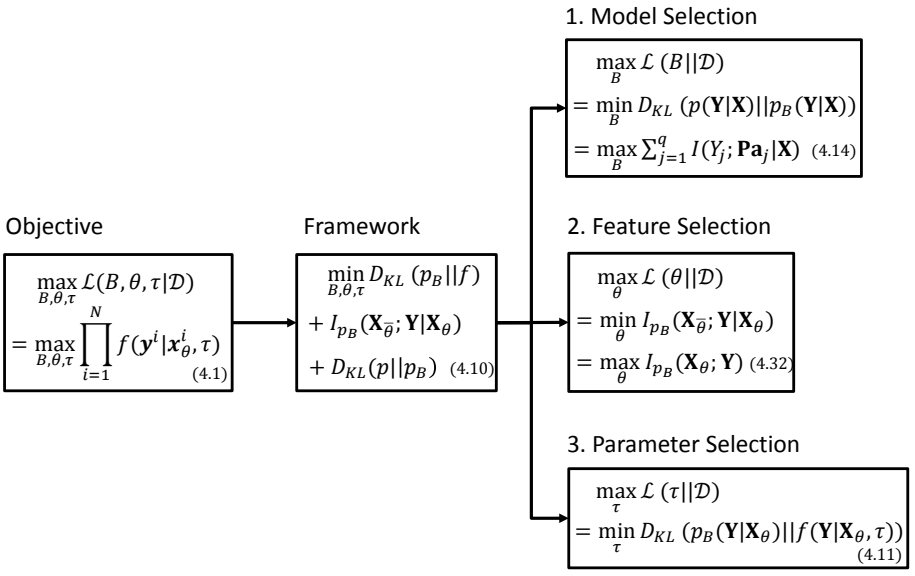


Figure 4.1: The framework of MLC via conditional likelihood maximization.

4. $H(\boldsymbol{Y}|\boldsymbol{X})$: the conditional entropy of labels $\boldsymbol{Y}$ given features $\boldsymbol{X}$. This term presents the uncertainty in the labels when all features are known, which is a bound on the Bayes error [26]. Since it is independent of all parameters, we can remove this term from our optimization problem.

Based on above discussion, we will take the following strategy:

Optimization Strategy. *We address the problem of multi-label classification with feature selection in three stages: first learning label space structure, then selecting useful feature subset, and last building the predictive model.*

Fig. 4.1 shows the framework of MLC via conditional likelihood maximization following the Optimization Strategy. In this strategy, instead of directly addressing the optimization problem (4.2), we solve the sub-problems (4.13), (4.12), and (4.11) independently. In the rest of this section, we shall discuss how the sub-problems (4.13) and (4.12) can be solved, respectively. In addition, we see that some popular MLC methods and multi-label feature selection algorithms are embedded in this strategy as the special cases of optimization of sub-problems with appropriate assumptions on the label or feature space.

4.2 Optimized Classifier Chains (OCC)

4.2.1 Model Selection for Classifier Chains

Under the Optimization Strategy, to model the underlying probability distribution $p(\boldsymbol{Y}|\boldsymbol{X})$, the optimal Bayesian network B^* of a special type could be obtained by optimizing

$$\arg\max_B \mathcal{L}(B|\mathcal{D}) = \arg\min_B D_{KL}(p(\boldsymbol{Y}|\boldsymbol{X})||p_B(\boldsymbol{Y}|\boldsymbol{X})) \tag{4.14}$$

Here we limit B in the k-Dependence Bayesian (kDB) network B:

$$p_B(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{j=1}^{L} p(Y_j|\boldsymbol{Pa}_j, \boldsymbol{X}) \tag{4.15}$$

where $\boldsymbol{Pa}_j$ represents the parents of label j, $|\boldsymbol{Pa}_j| = \min\{j-1, k\}$, $k \in [0, q-1]$. Note that kDB is limited in the number of parents up to k compared with the chain rule in the canonical order of $Y_1, Y_2, ..., Y_q$:

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{j=1}^{L} p(Y_j|Y_1, Y_2, ..., Y_{j-1}, \boldsymbol{X}). \tag{4.16}$$

The optimization problem (4.14) has been addressed in our previous work [81] as a theorem.

Theorem 2. *To approximate a conditional probability $p(\mathbf{Y}|\mathbf{X})$ in a certain family of Bayesian networks, the optimal Bayesian network B^* in K-L divergence is obtained if the sum of conditional mutual information between each variable of $\mathbf{Y}$ and its parent variables given the observation $\mathbf{X}$ is maximized.*

Proof. The optimization problem of (4.13) can be developed as follows:

$$\begin{aligned} B^* &= \arg\min_B \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p(\boldsymbol{Y}|\boldsymbol{X})}{p_B(\boldsymbol{Y}|\boldsymbol{X})} \right\} \\ &= \arg\max_B \mathbb{E}_{\boldsymbol{XY}} \left\{ \log p_B(\boldsymbol{Y}|\boldsymbol{X}) \right\} + H(\boldsymbol{Y}|\boldsymbol{X}), \end{aligned}$$

$H(\boldsymbol{Y}|\boldsymbol{X})$ can be omitted due to its independence with B, thus we have

$$\begin{aligned} B^* &= \arg\max_B \mathbb{E}_{\boldsymbol{XY}} \left\{ \log p_B(\boldsymbol{Y}|\boldsymbol{X}) \right\} \\ &= \arg\max_B \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \prod_{j=1}^{L} p(Y_j|\boldsymbol{Pa}_j, \boldsymbol{X}) \right\} \\ &= \arg\max_B \sum_{j=1}^{L} \mathbb{E}_{\boldsymbol{XY}} \left\{ \log \frac{p(Y_j, \boldsymbol{Pa}_j|\boldsymbol{X})}{p(Y_j|\boldsymbol{X}) p(\boldsymbol{Pa}_j|\boldsymbol{X})} \cdot p(Y_j|\boldsymbol{X}) \right\} \\ &= \arg\max_B \sum_{j=1}^{L} I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}) - \sum_{j=1}^{L} H(Y_j|\boldsymbol{X}). \end{aligned}$$

Since $\sum_{j=1}^{L} H(Y_j|\boldsymbol{X})$ is independent of B, we reach our conclusion:

$$B^* = \arg\min_B D_{KL}(p(\boldsymbol{Y}|\boldsymbol{X})||p_B(\boldsymbol{Y}|\boldsymbol{X})) = \arg\max_B \sum_{j=1}^{L} I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}). \quad (4.17)$$

□

According to Theorem 2, the following corollary is derived.

Corollary 1. *Given a specific order of labels, the probability p_B modeled by k-dependence Bayesian network optimally approximates $p(\boldsymbol{Y}|\boldsymbol{X})$ in terms of K-L divergence if the parents of the label Y_j holds the restriction $|\boldsymbol{Pa}_j| = j - 1$, $j = 1, ..., L$.*

Proof. We assume that labels are ordered in $Y_1, Y_2, ..., Y_L$, the possible parents of label j have been restricted as its previous labels $\boldsymbol{S}_j = \{Y_1, Y_2, ..., Y_{j-1}\}$, $j = 1, ..., L$. Hence, the optimization problem (4.17) can be independently solved by each label, and is equivalent to select optimally the parent labels from the previous labels.

$$B^* = \arg\max_B I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}), \quad j = 1, ..., L. \quad (4.18)$$

Let $\overline{\boldsymbol{Pa}}_j$ be $\boldsymbol{S}_j \setminus \boldsymbol{Pa}_j$. Based on the chain rule of mutual information, we have

$$I(Y_j; \boldsymbol{S}_j|\boldsymbol{X}) = I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}) + I(Y_j; \overline{\boldsymbol{Pa}}_j|\boldsymbol{Pa}_j, \boldsymbol{X}). \quad (4.19)$$

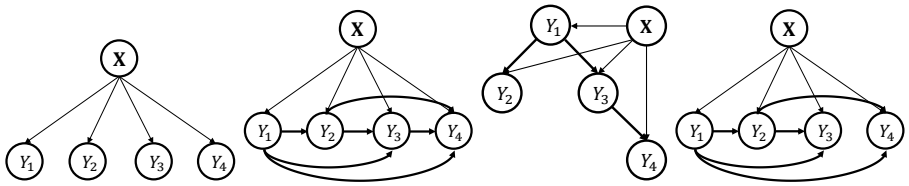
Since $I(Y_j; \overline{\boldsymbol{Pa}}_j|\boldsymbol{Pa}_j, \boldsymbol{X}) \geq 0$ (conditional mutual information is always non-negative), we have $I(Y_j; \boldsymbol{S}_j|\boldsymbol{X}) \geq I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X})$. Hence, (4.18) is optimized by treating all previous labels as the parents of Y_j, i.e., $\boldsymbol{Pa}_j = \boldsymbol{S}_j$. In this way, a fully-connected Bayesian network is chosen for B^*, thus $|\boldsymbol{Pa}_j| = j - 1$, $j = 1, ..., L$. □

Review CC-based methods

For the Classifier Chains (CC) and Probabilistic Classifier Chains (PCC) methods, given a predefined chain order, the classifier for each label is trained by taking the previous labels as extra features. In this sense, a fully-connected Bayesian network is constructed by CC and PCC according to a particular chain order. The difference between CC and PCC is that, in the testing phase, CC finds its prediction by the greedy search:

$$\hat{y}_j = \arg\max_{y_j} p(y_j|\hat{\boldsymbol{pa}}_j, \hat{\boldsymbol{x}}), \quad j = 1, ..., L. \quad (4.20)$$

In contrast, PCC aims to find the MAP assignment by searching 2^L paths according to (2.2), resulting in an exponential complexity in L for the testing



(a) BR (b) CC and PCC (c) BCC (d) kCC ($k = 2$)

Figure 4.2: Graphical models of CC-based methods in order Y_1, Y_2, Y_3, Y_4.

phase [19]. Thus PCC is actually a risk minimizer in terms of subset 0-1 loss [20], while CC can be considered as a deterministic approximation to PCC. The predictions of PCC ($\hat{\mathbf{y}}_{pcc}$) and CC ($\hat{\mathbf{y}}_{cc}$) shall be equal if the conditional probability of the MAP assignment ($\hat{\mathbf{y}}_{map}$) is more than 0.5 [19], i.e., $\hat{\mathbf{y}}_{cc} = \hat{\mathbf{y}}_{pcc}$, if $p(\hat{\mathbf{y}}_{map}|\hat{\mathbf{x}}) > 0.5$.

This study focuses on modeling $p(\mathbf{Y}|\mathbf{X})$ rather than finding the MAP assignment. The following corollary is introduced for four CC-based methods, CC, kCC, BCC and BR, which conduct the simple greedy search in the testing phase. Based on Corollary 1, the following corollary on these CC-based methods is derived.

Corollary 2. *In the same chain order of L labels, in the ideal case, classifier chains asymptotically outperform k-dependence classifier chains in terms of subset 0-1 loss, when $k < L - 1$. Similarly, k_1-dependence classifier chains asymptotically outperforms k_2-dependence classifier chains in subset 0-1 loss for $0 \leq k_2 < k_1 \leq L - 1$.*

Proof. Based on Corollary 1, given a chain order, classifier chains optimally models $p(\boldsymbol{Y}|\boldsymbol{X})$ with fully-connected Bayesian network. Therefore, according to (2.2), in fact CC is the optimized classifier in terms of subset 0-1 loss. Moreover, for the k-dependence classifier chains, K-L divergence between $p_B(\boldsymbol{Y}|\boldsymbol{X})$ and $p(\boldsymbol{Y}|\boldsymbol{X})$ always decreases as the value of k increases. According to (2.2) again, we reach our conclusion. □

Here we note that all of CC, BCC and BR can be viewed as the special cases of kCC by setting the value of k as $L - 1$, 1 and 0, respectively. CC works better than Bayesian Classifier Chains (BCC) and Binary Relevance (BR) in approximating ability of the underlying distribution $p(\boldsymbol{Y}|\boldsymbol{X})$. Therefore, by (2.2), CC outperforms BCC and BR in subset 0-1 loss in an asymptotic case. According to Corollary 2, given the same chain order, as an asymptotic analysis, we have the following order in performance:

$$\text{PCC} \succ \text{CC} \succ k\text{CC} \succ \text{BCC} \succ \text{BR} \ (1 < k < L - 1). \tag{4.21}$$

Here A $\succ$ B presents that A *asymptotically outperforms* B in *subset 0-1 loss*. It is worth noting that, PCC is considered in (4.21) since it is the exact optimizer in subset 0-1 loss. Fig. 4.2 shows the probabilistic graphical models of CC-based methods in label order Y_1, Y_2, Y_3, Y_4.

k-dependence Bayesian network for CC

As discussed above, using all preceding labels as $\boldsymbol{Pa}_j = \boldsymbol{S}_j$ is optimal for approximating $p(\boldsymbol{Y}|\boldsymbol{X})$ in a given chain order. However it is not always true in real-world problems with a finite number of instances. It is often possible to build a better model by eliminating irrelevant and redundant parent labels. Successful elimination of such useless features simplifies the learning model, and gives a better performance. In this study, therefore, we propose to use the k-dependence Bayesian network [28] where at most k preceding labels are adopted as the parents of a label.

As shown in the last section, given a chain order, the optimization problem (4.17) can solved independently for each label. Hence, it suffices to solve the problem

$$\arg\max_B \mathcal{L}(B|\mathcal{D}) = \arg\max_B I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}),\ \ j = 1, ..., L, \tag{4.22}$$

where $\boldsymbol{Pa}_j \in \boldsymbol{S}_j$ and $|\boldsymbol{Pa}_j| \leq k$. However, the calculation of conditional mutual information costs very highly for the high-dimensional feature vectors. Thus, we find an approximation for $I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X})$, relying on some appropriate assumptions.

Theorem 3.

$$\arg\max_B I(Y_j; \mathbf{Pa}_j) = \arg\max_B I(Y_j; \mathbf{Pa}_j|\mathbf{X}),$$

if the label Y_j is independent of the feature vector $\mathbf{X}$ conditioned on the parent labels $\mathbf{Pa}_j$.

Proof. According the fact $I(A; B|C) - I(A; B) = I(A; C|B) - I(A; C)$, we have

$$I(Y_j; \boldsymbol{Pa}_j|\boldsymbol{X}) = I(Y_j; \boldsymbol{Pa}_j) + I(Y_j; \boldsymbol{X}|\boldsymbol{Pa}_j) + I(Y_j; \boldsymbol{X}). \tag{4.23}$$

If the conditional independence assumption holds, the second term of (4.23) vanishes. In addition, the third term is independent of Bayesian network B. Thus we have the consistence. □

To simplify the computation, a forward parent label selection algorithm is developed here for the optimization problem $B^* = \arg\max_B I(Y_j; \boldsymbol{Pa}_j)$. Since

$$I(Y_j; \boldsymbol{Pa}_j \cup Y_l) = I(Y_j; \boldsymbol{Pa}_j) + I(Y_j; Y_l|\boldsymbol{Pa}_j), \tag{4.24}$$

starting from $\boldsymbol{Pa}_j = \emptyset$, we can update $\boldsymbol{Pa}_j$ by adding $Y_l \in \overline{\boldsymbol{Pa}}_j$ such that

$$Y_l = \arg\max_{Y_l \in \overline{\boldsymbol{Pa}}_j} I(Y_l; Y_j | \boldsymbol{Pa}_j), \tag{4.25}$$

until $|\boldsymbol{Pa}_j|$ reaches a predefined number. $I(Y_l; Y_j | \boldsymbol{Pa}_j)$ is further developed as,

$$\begin{aligned} I(Y_l; Y_j | \boldsymbol{Pa}_j) &= I(Y_l; Y_j) + I(Y_j; \boldsymbol{Pa}_j | Y_l) - I(Y_j; \boldsymbol{Pa}_j) \\ &= I(Y_l; Y_j) + H(\boldsymbol{Pa}_j | Y_l) - H(\boldsymbol{Pa}_j | Y_j, Y_l) - I(Y_j; \boldsymbol{Pa}_j) \end{aligned} \tag{4.26}$$

Parent Independence Assumption. *For target label Y_j, one preceding label $Y_k \in \boldsymbol{Pa}_j$, and $Y_l \in \overline{\boldsymbol{Pa}}_j$, we assume:*

$$p(\boldsymbol{Pa}_j | Y_l) = \prod_{Y_k \in \boldsymbol{Pa}_j} p(Y_k | Y_l), \tag{4.27}$$

$$p(\boldsymbol{Pa}_j | Y_j, Y_l) = \prod_{Y_k \in \boldsymbol{Pa}_j} p(Y_k | Y_j, Y_l). \tag{4.28}$$

These assumptions require that parent labels are conditional independent given one unselected parent Y_l (4.27) or given Y_l with the target label Y_j (4.28).

Under this assumption, we have the following:

$$\begin{aligned} I(Y_l; Y_j | \boldsymbol{Pa}_j) &= I(Y_l; Y_j) + \sum_{Y_k \in \boldsymbol{Pa}_j} H(Y_k | Y_l) - \sum_{Y_k \in \boldsymbol{Pa}_j} H(Y_k | Y_j, Y_l) - I(Y_j; \boldsymbol{Pa}_j), \\ &= I(Y_l; Y_j) + \sum_{Y_k \in \boldsymbol{Pa}_j} I(Y_k; Y_j | Y_l) - I(Y_j; \boldsymbol{Pa}_j). \end{aligned} \tag{4.29}$$

Substituting (4.29) into (4.25) and removing the last term independent of Y_l, we have

$$Y_l = \arg\max_{Y_l \in \overline{\boldsymbol{Pa}}_j} \left(I(Y_l; Y_j) + \sum_{Y_k \in \boldsymbol{Pa}_j} I(Y_j; Y_k | Y_l) \right). \tag{4.30}$$

In (4.30), the first term captures the *relevance* between the unselected parent label Y_l and the target label Y_j, while the second term models the mutual information between the selected feature Y_k and the target label Y_j, conditioned on the unselected label Y_l. Since we have $I(Y_j; Y_k | Y_l) = I(Y_j; Y_k) + I(Y_l; Y_k | Y_j) - I(Y_l; Y_k)$, the second term actually captures both *conditional redundancy* $I(Y_l; Y_k | Y_j)$ and *redundancy* $I(Y_l; Y_k)$. It means that the parent label which has the best trade-off between relevancy and redundancy would be selected. It is worth noting that in [10], the authors reach the similar conclusion for information theoretic feature selection. The difference is that here our objective is to perform parent label selection to find the optimal k-dependence Bayesian network given a chain order.

We can generalize (4.30) as

$$Y_l = \arg\max_{Y_l \in \overline{\boldsymbol{Pa}}_j} \left(I(Y_l; Y_j) + \alpha \sum_{Y_k \in \boldsymbol{Pa}_j} I(Y_j; Y_k | Y_l) \right), \ j = 1, ..., L, \tag{4.31}$$

where $\alpha \geq 0$ denotes a parameter controlling the weights assigned to the two terms. In practice, we prefer to set $\alpha = \frac{1}{|\boldsymbol{Pa}_j|}$ to prevent from sweeping the first term as the number parent labels grows. Based on (4.31), we propose the k-dependence Classifier Chains (k**CC**) method.

4.2.2 Multi-Label Feature Selection

Based on the Optimization Strategy, the following equation can be developed based on the built Bayesian network B,

$$\arg\max_{\theta} \mathcal{L}(\theta|\mathcal{D}) = \arg\min_{\theta} I_{p_B}(\boldsymbol{X}_{\overline{\theta}}; \boldsymbol{Y}|\boldsymbol{X}_{\theta}). \tag{4.32}$$

Note that there is no monotonicity in the conditional mutual information, so that $\theta = [0, 0, ..., 0]^\intercal$ or $\theta = [1, 1, ..., 1]^\intercal$ is not always the solution. According to the chain rule of mutual information and $\boldsymbol{X} = \{\boldsymbol{X}_{\theta}, \boldsymbol{X}_{\overline{\theta}}\}$, we have

$$I_{p_B}(\boldsymbol{X}; \boldsymbol{Y}) = I_{p_B}(\boldsymbol{X}_{\theta}; \boldsymbol{Y}) + I_{p_B}(\boldsymbol{X}_{\overline{\theta}}; \boldsymbol{Y}|\boldsymbol{X}_{\theta}). \tag{4.33}$$

From the above formula, we see that minimization of $I_{p_B}(\boldsymbol{X}_{\overline{\theta}}; \boldsymbol{Y}|\boldsymbol{X}_{\theta})$ is equivalent to maximization of $I_{p_B}(\boldsymbol{X}_{\theta}; \boldsymbol{Y})$, thus (4.32) is transformed into:

$$\arg\max_{\theta} \mathcal{L}(\theta|\mathcal{D}) = \arg\max_{\theta} I_{p_B}(\boldsymbol{X}_{\theta}; \boldsymbol{Y}). \tag{4.34}$$

Note that the similar optimization problem has been proposed by intuition in the field of information theoretic feature selection for the single-label variable Y in a variety of papers [65]. Here we theoretically derive the optimization problem from conditional likelihood maximization for the multi-label case, and take label correlations into account by the learned Bayesian network B.

However, directly solving (4.34) is a non-trivial thing due to the difficulty on the calculation of mutual information between high-dimensional $\boldsymbol{X}_{\theta}$ and $\boldsymbol{Y}$. Hence, similar with Section 4.2.1, the greedy sequential optimization is applied for (4.34). Since

$$I_{p_B}(\boldsymbol{X}_{\theta} \cup X_m; \boldsymbol{Y}) = I_{p_B}(\boldsymbol{X}_{\theta}; \boldsymbol{Y}) + I_{p_B}(\boldsymbol{X}_m; \boldsymbol{Y}|\boldsymbol{X}_{\theta}), \tag{4.35}$$

Starting from $\boldsymbol{X}_{\theta} = \emptyset$, we can update $\boldsymbol{X}_{\theta}$ by adding $X_m \in \boldsymbol{X}_{\overline{\theta}}$ such that

$$X_m = \arg\max_{X_m \in \boldsymbol{X}_{\overline{\theta}}} I_{p_B}(X_m; \boldsymbol{Y}|\boldsymbol{X}_{\theta}). \tag{4.36}$$

until $|\boldsymbol{X}_{\theta}|$ reaches a predefined number. By taking the Bayesian network B into consideration, we have

$$\begin{aligned} I_{p_B}(X_m; \boldsymbol{Y}|\boldsymbol{X}_{\theta}) &= \sum_{j=1}^{L} I(X_m; Y_j|\boldsymbol{Pa}_j, \boldsymbol{X}_{\theta}), \\ &= \sum_{j=1}^{q} \left(I(X_m; Y_j) + I(\boldsymbol{Pa}_j, \boldsymbol{X}_{\theta}; Y_j|X_m) - I(\boldsymbol{Pa}_j, \boldsymbol{X}_{\theta}; Y_j)\right). \end{aligned} \tag{4.37}$$

Parent and Feature Independence Assumption. *For each feature* $X_n \in \boldsymbol{X}_\theta$*, given one unselected feature* $X_m \in \boldsymbol{X}_{\overline{\theta}}$ *and the target labels* $\boldsymbol{Y}$*, we assume the following:*

$$p(\boldsymbol{Pa}_j, \boldsymbol{X}_{\theta_j}|Y_j) = \prod_{Y_k \in \boldsymbol{Pa}_j} p(Y_k|Y_j) \prod_{X_n \in \boldsymbol{X}_\theta} p(X_n|Y_j),$$
$$p(\boldsymbol{Pa}_j, \boldsymbol{X}_\theta|Y_j, X_m) = \prod_{Y_k \in \boldsymbol{Pa}_j} p(Y_k|Y_j, X_m) \prod_{X_n \in \boldsymbol{X}_\theta} p(X_n|Y_j, X_m),$$

It requires that the parents $\boldsymbol{Pa}_j$ *and selected features* $\boldsymbol{X}_\theta$ *are conditional independent of* Y_j *with or without the unselected feature* X_m*.*

Based on the above assumption and (4.37), (4.36) becomes:

$$X_m = \arg\max_{X_m \in \boldsymbol{X}_{\overline{\theta}}} \sum_{j=1}^{q} \left(I(X_m; Y_j) + \sum_{Y_k \in \boldsymbol{Pa}_j} I(Y_j; Y_k|X_m) + \sum_{X_n \in \boldsymbol{X}_\theta} I(Y_j; X_n|X_m) \right). \tag{4.38}$$

The first term captures the *Relevance* between the unselected feature X_m and the target label Y_j, and the second term captures *Label Correlations* modeled in k-dependence Bayesian network. For the third term of (4.38), we express it as

$$I(Y_j; X_n|X_m) = I(Y_j; X_n) + I(X_m; X_n|Y_j) - I(X_m; X_n). \tag{4.39}$$

Since $I(Y_j; X_n)$ is independent of X_m, $I(Y_j; X_n|X_m)$ can be represented by last two terms. Here we call $I(Y_j; X_n|X_m)$ as *Redundancy Difference*, since it in fact reflects the difference of *Conditional Redundancy* $I(X_m; X_n|Y_j)$ and *Redundancy* $I(X_m; X_n)$. Hence, according to (4.38), one feature would be selected in that it provides the best trade-off among relevance, label correlations and redundancy.

We can further generalize (4.38) as,

$$X_m = \arg\max_{X_m \in \boldsymbol{X}_{\overline{\theta}}} \sum_{j=1}^{L} \left(I(X_m; Y_j) + \beta \sum_{Y_k \in \boldsymbol{Pa}_j} I(Y_j; Y_k|X_m) + \gamma \sum_{X_n \in \boldsymbol{X}_\theta} I(Y_j; X_n|X_m) \right). \tag{4.40}$$

where $\beta, \gamma \geq 0$ are two parameters controlling the weights on correlation and redundancy difference terms, respectively. In practice, we typically set $\beta = \frac{1}{|\boldsymbol{Pa}|}$ and $\gamma = \frac{1}{|\boldsymbol{X}_\theta|}$ to normalize the terms, preventing from sweeping either term as the number of $\boldsymbol{Pa}_j$ or $\boldsymbol{X}_{\theta_j}$ grow. This means we hold a relatively weak assumptions on conditional label independence (the 2nd term) and feature independence (the 3rd term).

In the derivation of (4.40), we assume the relevant features are identical for all the labels. However, it is more general and reasonable to assume that the selected feature subset is label-specific. For the target label j, its relevant features are denoted as $\boldsymbol{X}_{\theta_j}$, thus $\boldsymbol{X} = \{\boldsymbol{X}_{\theta_j}, \boldsymbol{X}_{\overline{\theta}_j}\}$, $j = 1, ..., L$. Based on the

Table 4.1: The generality of the proposed framework for MLC.

	Parameter setting	Algorithm	Authors
Model selection	$\lvert \boldsymbol{Pa}_j \rvert = 0$	Binary Relevance (BR)	Boutell et al. (2004) [8]
	$\lvert \boldsymbol{Pa}_j \rvert \leq 1$	Bayesian CC (BCC)	Zaragoza et al.(2011) [107]
	$\lvert \boldsymbol{Pa}_j \rvert = j-1$	Classifier Chains (CC)	Read et al. (2011) [69]
	$\lvert \boldsymbol{Pa}_j \rvert \leq k$	k**CC** (Proposed)	**Sun and Kudo (2016)**
Feature selection	$L=1$, $\beta=\gamma=0$	MI Maximization (MIM)	Lewis (1992) [52]
	$L=1$, $\beta=0$, $\gamma=\frac{1}{\lvert \boldsymbol{X}_\theta \rvert}$	Joint MI (JMI)	Yang and Moody (1999)[102]
		Max-Rel Min-Red†	Peng et al. (2005) [65]
	$L>1$, $\beta=\gamma=0$	Multi-Label MIM	Scchidis et al (2012) [77]
	$L>1$, $\beta=0$, $\gamma=\frac{1}{\lvert \boldsymbol{X}_\theta \rvert}$	Multi-Label JMI	Scchidis et al. (2012) [77]
	$L>1$, $\beta=\frac{1}{\lvert \boldsymbol{Pa}_j \rvert}$, $\gamma=\frac{1}{\lvert \boldsymbol{X}_\theta \rvert}$	**MLFS** (Proposed)	**Sun and Kudo (2016)**

† Max-Rel Min-Red ignores the conditional redundancy term in (4.40).

similar derivation and assumption of (4.40), we can obtain the following feature selection criterion for label j, $j = 1, ..., L$,

$$X_m = \underset{X_m \in \boldsymbol{X}_{\overline{\theta}_j}}{\arg\max} \left(I(X_m; Y_j) + \beta \sum_{Y_k \in \boldsymbol{Pa}_j} I(Y_j; Y_k | X_m) + \gamma \sum_{X_n \in \boldsymbol{X}_{\theta_j}} I(Y_j; X_n | X_m) \right). \tag{4.41}$$

In this case, we set $\beta = \frac{1}{|\boldsymbol{Pa}_j|}$ and $\gamma = \frac{1}{|\boldsymbol{X}_{\theta_j}|}$. Note that the criterion (4.41) is optimized label-independently, since $X_m \in \boldsymbol{X}_{\theta_j}$ and $\boldsymbol{X}_{\theta_j}$ is label-specific. In this paper, we employ (4.40) and (4.41) for *global* and *local* Multi-Label Feature Selection (**MLFS**), respectively.

4.2.3 Summary of Theoretical Findings

In Table 4.1, we summarize the techniques above for comparing model selection and feature selection algorithms. By combining kCC (4.31) and local MLFS (4.41), we propose the Optimized Classifier Chains (**OCC**) method, whose procedure is outlined in Algorithm 3.

4.3 Experimental Results

4.3.1 Implementation Issues

In the optimizations of both model and feature selection, calculation of mutual information is extensively used. For the discrete/categorical feature variables $\boldsymbol{X}$ ($\boldsymbol{Y}$ is originally binary), the calculation of mutual information is simple and straightforward. Given a sample of N i.i.d. observations $\{x^i, y^i\}_{i=1}^N$, based on the law of large numbers, we have the following approximation:

$$I(X;Y) = \sum_{xy} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \approx \frac{1}{N} \sum_{i=1}^{N} \log \frac{\hat{p}(x^i, y^i)}{\hat{p}(x^i)\hat{p}(y^i)}, \tag{4.42}$$

Algorithm 3 The algorithm of OCC (kCC + MLFS)

Input: $\mathcal{D} = \{\mathbf{x}^i, \mathbf{y}^i\}_{i=1}^N$: training set with L labels, **chain**: label order, $\hat{\mathbf{x}}$: test instance, k: maximal size of parents, M_s: size of selected features, $f_1, ..., f_L$: L predictive models

Output: $\hat{\mathbf{y}}$: predicted label set of $\hat{\mathbf{x}}$

Training:

$\boldsymbol{S} \leftarrow \emptyset$, $\{\boldsymbol{Pa}_j\}_{j=1}^L \leftarrow \emptyset$;
for $j \in$ **chain do**
  for $l = 1, ..., k$ **do**
    select Y_l from $\boldsymbol{S}$ according to (4.31);
    $\boldsymbol{Pa}_j \leftarrow \boldsymbol{Pa}_j \cup Y_l$;
  for $m = 1, ..., M_s$ **do**
    select X_m from $\boldsymbol{X}$ according to (4.41);
    $\boldsymbol{X}_{\theta_j} \leftarrow \boldsymbol{X}_{\theta_j} \cup X_m$;
  build classifier f_j on $\mathcal{D}_j$, where $\mathcal{D}_j = \{\mathbf{x}_{\theta_j}^i \cup \mathbf{pa}_j^i, \mathbf{y}_j^i\}_{i=1}^N$;
  $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup Y_j$;

Testing:

for $j \in$ **chain do**
  $\hat{y}_j \leftarrow f_j(\hat{\mathbf{x}}_{\theta_j} \cup \hat{\mathbf{pa}}_j)$;
$\hat{\mathbf{y}} \leftarrow (\hat{y}_1, \hat{y}_2, ..., \hat{y}_L)$.

where $\hat{p}$ denotes the empirical probability distribution. In contrast, when the feature variable X is continuous, it becomes quite difficult to compute mutual information $I(X;Y)$, since it is typically impossible to obtain $\hat{p}$. One of the solutions is to use kernel density estimation, in which case, $\hat{p}$ is approximated by the following:

$$\hat{p}(x) = \frac{1}{N} \sum_{i=1}^{N} K_h(x - x^i), \tag{4.43}$$

where $K_h(\cdot)$ is a non-negative kernel function with bandwidth h ($h > 0$). Typically, the Gaussian kernel can be used as $K_h(\cdot)$. However, it is computationally expensive to compute (4.43) and usually difficult to select good value of the bandwidth h.

Another solution for computing $I(X;Y)$ with continuous feature X is to apply data discretization as preprocessing. In this study, we adopt this method and discretize the continuous feature X based on its mean μ_X and standard deviation σ_X. For example, we can threshold the values of X into three categories $\{-1, 0, 1\}$ at $\mu_X \pm \sigma_X$. The experimental results demonstrate the efficiency of such data discretization approach for approximating $I(X;Y)$ to perform feature selection.

Algorithm 4 The algorithm of chain order selection

Input: $\mathcal{D} = \{\mathbf{x}^i, \mathbf{y}^i\}_{i=1}^N$: training data with L labels;
Output: $\boldsymbol{\pi}$: selected chain order of L labels;

1: $\mathbf{S}_\pi \leftarrow \emptyset$, $\overline{\mathbf{S}}_\pi \leftarrow \{1, 2, ..., L\}$;
2: Initialize $\boldsymbol{\pi}(1) \leftarrow \arg\max_{\boldsymbol{\pi}(1)} \sum_{j=2}^{L} \sum_{m=1}^{M} I(Y_{\pi(1)}; Y_{\pi(j)}|X_m)$;
3: $\mathbf{S}_\pi \leftarrow \boldsymbol{\pi}(1)$, $\overline{\mathbf{S}}_\pi \leftarrow \mathbf{S}_\pi / \boldsymbol{\pi}(1)$;
4: **for** $j = 2, ..., L$ **do**
5: Select $\boldsymbol{\pi}(j) \in \overline{\mathbf{S}}_\pi$ according to (4.45);
6: $\mathbf{S}_\pi \leftarrow \mathbf{S}_\pi \cup \boldsymbol{\pi}(j)$, $\overline{\mathbf{S}}_\pi \leftarrow \mathbf{S}_\pi / \boldsymbol{\pi}(j)$;

4.3.2 Chain order selection

The chain order is crucial to the performance of CC-based methods [69, 19]. To improve the performance, [49] and [70] propose to select the correct chain order by using the Beam Search and Monte Carlo algorithm, respectively. In this study, we develop a greedy algorithm for efficient chain order selection according to Theorem 2. Different from Theorem 2, where model selection is conducted to find the optimal k-dependence Bayesian network ($|\boldsymbol{Pa}_j| \leq k$, $\forall j$) based on the canonical order, here we aim to perform model selection by selecting the optimal chain order $\boldsymbol{\pi}$ for a fully-connected Bayesian network B_π ($|\boldsymbol{Pa}_j| = j - 1$, $\forall j$).

Suppose that a permutation $\boldsymbol{\pi} : \{1, 2, ..., L\} \mapsto \{1, 2, ..., L\}$ determines the optimal chain order of labels, so that the jth label is placed at the $\boldsymbol{\pi}(j)$th position of the chain. Thus, following Theorem 2, we have

$$\boldsymbol{\pi}^* = \arg\min_{B_\pi} D_{KL}(p(\boldsymbol{Y}|\boldsymbol{X})||p_{B_\pi}(\boldsymbol{Y}|\boldsymbol{X})) = \arg\max_{\boldsymbol{\pi}} \sum_{j=1}^{L} I(Y_{\pi(j)}; \boldsymbol{Pa}_j|\boldsymbol{X}), \tag{4.44}$$

where $\boldsymbol{Pa}_j = \{Y_{\pi(k)}\}_{k=1}^{j-1}$. Similar to the derivation in Section 4.2, a greedy forward search algorithm can be developed to solve the optimization problem in (4.44) with appropriate independence assumption. In the jth iteration, we select the best place $\boldsymbol{\pi}(j)$ for the jth label according to the following formula:

$$\boldsymbol{\pi}^*(j) = \arg\max_{\boldsymbol{\pi}(j)} \sum_{k=1}^{j-1} \left(\frac{1}{M} \sum_{m=1}^{M} I(Y_{\pi(j)}; Y_{\pi(k)}|X_m) + \frac{1}{k-1} \sum_{l=1}^{k-1} I(Y_{\pi(k)}; Y_{\pi(l)}|Y_{\pi(j)}) \right). \tag{4.45}$$

Algorithm 4 outlines the procedure of chain order selection.

4.3.3 Comparison with the State-of-the-Art

In the experiments we compare the following seven MLC methods:

BR [8]: it transforms a multi-label problem with L labels into L binary classification problems according to the one-vs-all strategy.

ECC [69]: ensemble of CCs. A number of CCs are established by randomly selecting the chain orders, and the final prediction is made by majority votes.

EBCC [107]: ensemble of BCCs. A number of BCCs are established by randomly selecting theirs roots, and its prediction is made by the same way ECC does.

MDDM [113]: a *global* FS-DR method. By maximizing the feature-label dependence, the projection is built to project the features into the low-dimensional subspace.

LLSF [38]: a *local* FS-DR method. Label-specific features are selected by optimizing the least squares problem with constraints of label correlation and feature sparsity.

OCC[3]: the proposed Optimized Classifier Chains (OCC) method.

EOCC[3]: ensemble of OCCs (EOCC). The same ensemble strategy of ECC is applied.

BR was used as the baseline MLC method. As the ensemble CC-based methods, ECC and EBCC was introduced due to their superior performances over some MLC decomposition methods as shown in [69, 107]. MDDM was chosen as a representative of global FS-DR methods, which outperformed several global FS-DR methods, like PCA, LPP [34], MLSI [105], CCA [80], as reported in [113]. As a local FS-DR method, LLSF was chosen due to its performance advantage in comparison with another local FS-DR method, LIFT [108], as reported in [38].

In the experiments, *5-fold cross validation* was performed to evaluate the classification performance. For fair comparison, a linear SVM implemented in Liblinear [25] was used as the baseline binary classifier for all the comparing methods. In parameter setting, we tune the important parameters on controlling the feature sparsity for the FS-DR methods, such as MDDM, LLSF and OCC/EOCC. Specifically, the dimensionality of feature space in MDDM (thr) and OCC/EOCC (M) is selected from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ of total number M of features, while the value of β in LLSF is tuned in range of $\{0.001, 0.01, 0.1, 1, 10\}$. As suggested in [113] and [38], we set the regularization parameter $\mu = 0.5$ in MDDM, and the parameters $\alpha = 0.5$ and $\gamma = 0.01$ in LLSF. To balance the performance of oCC in Exact-Match and Macro/Micro-F1, we set the parameters as $k = \lceil 0.8 \times L \rceil$. For the ensemble methods, we use an ensemble of 10 CCs/BCCs/OCCs for building ECC/EBCC/EoCC. In order to scale up the ensemble methods, random sampling is applied to randomly

[3]We provide the codes of OCC and EOCC at: https://github.com/futuresun912/oCC.git

Table 4.2: Experimental results (mean±std rank) of comparing methods on thirteen multi-label datasets in terms of **Exact-Match**.

Dataset	BR	ECC	EBCC	MDDM	LLSF	oCC	EoCC
emotions	.233±.019 5.5	.270±.024 2.5	.238±.001 4	.206±.002 7	.233±.017 5.5	.270±.023 2.5	**.290±.011 1**
scene	.516±.013 6	.650±.021 3	.572±.013 4	.511±.010 7	.524±.010 5	**.669±.011 1**	.655±.007 2
yeast	.148±.009 6	.201±.014 3	.179±.011 4	.147±.008 7	.152±.010 5	**.210±.010 1**	.204±.009 2
birds	.474±.019 5.5	.482±.022 4	.489±.001 2	.464±.006 8	.474±.032 5.5	**.496±.013 1**	.487±.015 3
genbase	.982±.005 2.5	.973±.001 6	.973±.001 6	.977±.015 4	.982±.001 2.5	**.985±.005 1**	.973±.004 6
medical	.670±.015 4	.671±.018 3	.660±.001 6	.601±.007 7	.663±.011 5	**.694±.012 1**	.683±.014 2
enron	.113±.008 6	**.144±.006 1**	.130±.001 4	.112±.004 7	.114±.006 5	.140±.010 3	.143±.007 2
languagelog	.158±.002 6	.162±.001 2.5	**.163±.001 1**	.158±.004 6	.158±.001 6	.161±.005 4	.162±.003 2.5
rcv1s1	.073±.004 6	.138±.004 2	.105±.001 4	.064±.002 7	.074±.003 5	**.148±.004 1**	.137±.005 3
bibtex	.149±.003 5.5	.172±.001 2.5	.172±.001 2.5	.133±.003 7	.149±.001 5.5	.163±.004 4	**.176±.003 1**
corel16k1	.008±.001 5	.015±.002 2.5	.010±.001 4	.007±.001 6.5	.007±.001 6.5	**.023±.001 1**	.015±.002 2.5
corel5k	.006±.001 6	.013±.001 2	.009±.001 4	.005±.002 7	.007±.001 5	**.020±.003 1**	.012±.002 3
delicious	.002±.001 6	**.005±.001 1**	.003±.001 4	.002±.001 6	.002±.001 6	.004±.002 2.5	.004±.001 2.5
Rank	5.231 5.5	2.731 3	3.808 4	6.692 7	5.231 5.5	**1.846 1**	2.462 2

Table 4.3: Experimental results (mean±std rank) of comparing methods on thirteen multi-label datasets in terms of **Hamming-Score**.

Dataset	BR	ECC	EBCC	MDDM	LLSF	oCC	EoCC
emotions	.785±.006 2.5	.778±.007 4.5	.778±.001 4.5	.776±.006 6	.785±.010 2.5	.767±.006 7	**.787±.006 1**
scene	.890±.003 7	.907±.005 2	.898±.001 4	.892±.002 5.5	.892±.002 5.5	.900±.004 3	**.909±.001 1**
yeast	.798±.004 2	.794±.005 5.5	.795±.001 4	.796±.005 3	**.799±.002 1**	.787±.005 7	.794±.005 5.5
birds	.948±.002 5.5	.950±.004 3.5	.950±.001 3.5	.946±.009 7	.948±.002 5.5	**.951±.002 1.5**	**.951±.002 1.5**
genbase	**.999±.000 4**	**.999±.001 4**	**.999±.001 4**	**.999±.001 4**	**.999±.001 4**	**.999±.000 4**	**.999±.000 4**
medical	**.990±.000 3.5**	**.990±.000 3.5**	**.990±.001 3.5**	.987±.001 7	**.990±.001 3.5**	**.990±.000 3.5**	**.990±.000 3.5**
enron	.942±.001 5	.950±.001 2	.940±.001 7	.948±.000 3	.941±.001 6	.946±.001 4	**.951±.001 1**
languagelog	.797±.000 6.5	.830±.001 2.5	.828±.001 4	.822±.000 5	.797±.000 6.5	**.836±.000 1**	.830±.000 2.5
rcv1s1	.966±.000 6.5	**.973±.001 1.5**	.972±.001 3	.971±.000 4	.966±.000 6.5	.967±.000 5	**.973±.000 1.5**
bibtex	.985±.000 5	**.988±.001 2**	**.988±.001 2**	.984±.000 7	.985±.001 5	.985±.000 5	**.988±.000 2**
corel16k1	.980±.000 5.5	**.981±.000 2.5**	**.981±.001 2.5**	**.981±.000 2.5**	.980±.000 5.5	.977±.000 7	**.981±.000 2.5**
corel5k	.988±.000 5	**.990±.001 2**	**.990±.001 2**	.987±.000 7	.988±.001 5	.988±.000 5	**.990±.000 2**
delicious	.981±.000 4.5	**.982±.000 2**	**.982±.001 2**	.980±.000 6	.981±.000 4.5	.979±.000 7	**.982±.000 2**
Rank	4.885 6	3.269 2	3.462 3	5.308 7	4.654 5	4.231 4	**2.192 1**

select 75% of instances and 50% of features for building each single model of the ensemble, as recommended in [69]. All the comparing methods were implemented in Matlab, and experiments were performed in a computer configured with an Intel Quad-Core i7-4770 CPU at 3.4GHz with 4GB RAM.

Tables 4.2 to 4.5 report the experimental results of seven comparing MLC methods over the thirteen benchmark multi-label datasets. For each evaluation metric, the larger the value, the better the performance. Among seven comparing methods, the best performance is highlighted in boldface. The average rank of each method over the datasets is reported in the last row of each Table.

The proposed oCC outperformed the other methods in terms of Exact-Match, Macro-F1 and Micro-F1 on average. It demonstrates the necessity of removing useless parents and features in classifier chains, indicating the effec-

Table 4.4: Experimental results (mean±std rank) of comparing methods on thirteen multi-label datasets in terms of **Macro-F1**.

Dataset	BR	ECC	EBCC	MDDM	LLSF	oCC	EoCC
emotions	.545±.015 5.5	.588±.006 2	.570±.001 4	.543±.009 7	.545±.015 5.5	.580±.012 3	**.613±.013 1**
scene	.684±.007 6	.742±.011 2	.712±.001 4	.673±.005 7	.688±.006 5	.724±.012 3	**.746±.004 1**
yeast	.355±.006 4.5	.359±.003 2.5	.353±.001 6	.355±.008 4.5	.351±.003 7	**.410±.007 1**	.359±.006 2.5
birds	.135±.018 3.5	.125±.006 7	.131±.001 5	.139±.002 2	.135±.014 3.5	**.177±.017 1**	.127±.013 6
genbase	**.769±.016 1.5**	.725±.001 6.5	.745±.001 5	.765±.012 3	**.769±.001 1.5**	.758±.016 4	725±.015 6.5
medical	.374±.007 4	.328±.013 6	.323±.001 7	.383±.007 2	.378±.008 3	**.387±.006 1**	.330±.007 5
enron	**.220±.010 1**	.192±.010 6.5	.197±.001 4.5	.218±.003 2.5	.218±.004 2.5	.194±.007 6.5	.197±.003 4.5
languagelog	.391±.003 2	.387±.008 6	.389±.001 4.5	.389±.006 4.5	.390±.005 3	.365±.004 7	**.398±.004 1**
rcv1s1	**.246±.007 1.5**	.211±.008 6	.219±.001 5	.234±.002 4	**.246±.005 1.5**	.235±.009 3	.209±.007 7
bibtex	**.329±.001 1.5**	.248±.001 7	.254±.001 5	.308±.003 4	**.329±.001 1.5**	.328±.003 3	.250±.002 6
corel16k1	.060±.002 2	.048±.003 4	.033±.001 7	.041±.001 6	.059±.003 3	**.064±.001 1**	.045±.001 5
corel5k	.046±.001 2.5	.031±.001 5.5	.031±.001 5.5	.043±.002 4	**.047±.001 1**	.046±.001 2.5	.030±.001 7
delicious	.123±.002 3	.102±.003 5	.098±.001 6	**.130±.001 1**	.122±.003 4	.124±.005 2	.096±.001 7
Rank	3.039 2	5.039 6	5.308 7	4.000 4	3.115 3	**2.923 1**	3.039 5

Table 4.5: Experimental results (mean±std rank) of comparing methods on thirteen multi-label datasets in terms of **Micro-F1**.

Dataset	BR	ECC	EBCC	MDDM	LLSF	oCC	EoCC
emotions	.591±.016 5.5	.620±.005 2	.608±.001 3	.578±.010 7	.591±.016 5.5	.606±.011 4	**.647±.010 1**
scene	.677±.010 6	.734±.010 2	.704±.001 4	.668±.008 7	.682±.006 5	.715±.010 3	**.738±.003 1**
yeast	.633±.008 5	.642±.005 2	.637±.001 3	.631±.010 6.5	.631±.004 6.5	.636±.009 4	**.645±.009 1**
birds	.266±.026 6	.286±.011 2	.279±.001 3	.266±.001 6	.266±.011 6	**.300±.025 1**	.277±.024 4
genbase	**.993±.002 2**	.989±.001 5.5	.988±.001 7	.991±.007 4	**.993±.001 2**	**.993±.002 2**	.989±.002 5.5
medical	.809±.007 4	.804±.007 5	.799±.001 6	.764±.004 7	**.815±.011 1**	.811±.006 3	.812±.005 2
enron	.521±.005 6	.566±.008 3	.569±.001 2	.553±.003 4	.517±.003 7	.532±.006 5	**.573±.004 1**
languagelog	.520±.006 6.5	.573±.004 3	.570±.001 4	.553±.002 5	.520±.003 6.5	**.579±.006 1**	.577±.005 2
rcv1s1	.401±.003 4.5	.429±.004 2	.427±.001 3	.392±.004 7	.401±.003 4.5	.394±.003 6	**.432±.004 1**
bibtex	.429±.002 2.5	.416±.001 6	.419±.001 4	.412±.002 7	.429±.001 2.5	**.430±.002 1**	.418±.003 5
corel16k1	.106±.002 4	.137±.005 2.5	.088±.001 6	.078±.002 7	.105±.002 5	**.167±.007 1**	.137±.003 2.5
corel5k	.167±.003 4	.169±.001 2	.151±.001 6	.131±.003 7	.167±.001 4	**.191±.005 1**	.167±.003 4
delicious	**.257±.001 1.5**	.234±.005 4	.231±.001 6	.250±.002 3	**.257±.002 1.5**	.233±.004 5	.223±.003 7
Rank	4.423 6	3.115 3	4.385 5	6.077 7	4.346 4	**2.769 1**	2.885 2

Table 4.6: Results of the Friedman Statistics F_F (7 methods, 13 datasets) and the Critical Value (0.05 significance level). The null hypothesis as the equal performance is rejected, if the values of F_F in terms of all metrics are higher than the Critical Value.

Friedman test	Exact-Match	Hamming-Score	Macro-F1	Micro-F1
F_F	25.017	3.967	3.273	5.052
Critical Value		2.230		

tiveness of the proposed framework on handling the MLC problems. On the other hand, benefiting from the ensemble strategy and oCC, EoCC ranked 1st in Hamming-Score, and worked the best among the three ensemble methods in four metrics. For the other ensemble methods, ECC achieved competitive results in all metrics except Macro-F1, and performed consistently better than EBCC, verifying the conclusion we reached at (4.21). Such observation is consistent with our theoretical analysis in Section 4.1, which shows BCC would worked worse than CC in modeling label correlations due to its limitation of $|\boldsymbol{Pa}| \leq 1$. For the FS-DR methods, the local method LLSF performed consistently better than the global method MDDM in all metrics. It is probably because that selection of global feature subset would loss some label-specific discriminative information. As the baseline method, BR is competitive with the best methods only in Macro-F1. The worse performance of BR probably results from its simple decomposition strategy which ignores label correlations.

To perform comparative analysis on experimental results in Tables 4.2 to 4.5 by statistical test, we utilized the evaluation methodology for MLC used in [15, 58], i.e., *Friedman test* with a post-hoc *Nemenyi test*. We fist conducted Friedman test [21] (7 methods, 13 datasets) with significance level 0.05, aiming to reject the null-hypothesis as equal performance among the comparing methods. The results are shown in Table 8.3. Since the values of the Friedman Statistic F_F in terms of all metrics are higher than the Critical Value, the null hypothesis as the equal performance was rejected. Therefore, a post-hoc test, Nemenyi test, is subsequently conducted to evaluate the performance between every two methods. According to [21], the performance of two methods is regarded as significantly different if their average ranks differ by at least the *critical difference* (CD). Figure 8.5 shows the CD diagrams for four evaluation metrics at 0.05 significance level. In each subfigure, the CD is given above the axis, where the averaged rank is marked. In Figure 8.5, algorithms which are not significantly different are connected by a thick line. In terms of Exact-Match, among 91 comparisons (7 methods $\times$ 13 datasets), OCC/EOCC outperformed other methods, and achieved statistically superior performance than the other methods except ECC/EBCC, consistent with the theoretical

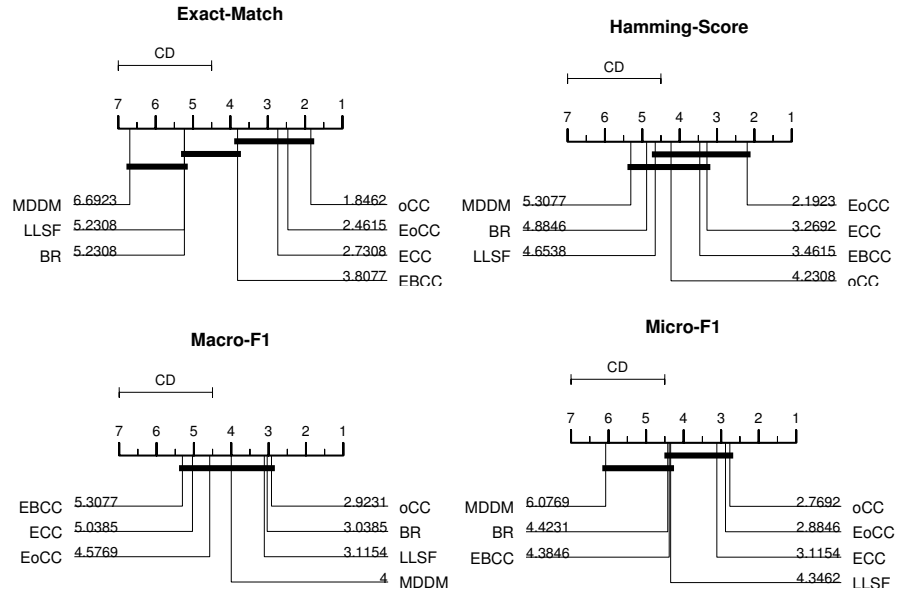


Figure 4.3: CD diagrams (0.05 significance level) of seven comparing methods in four evaluation metrics. The performance of two methods is regarded as significantly different if their average ranks differ by at least the Critical Difference.

analysis. Moreover, OCC/EOCC worked better than ECC on average, showing the effectiveness of the proposed framework on optimizing CC. In Hamming-Score, OCC ranked fourth, following the three ensemble methods. It is because the objective of OCC is to approximate the optimizer in Exact-Match, which probably brings in the performance loss in Hamming-Score, as shown in [18]. In Macro-F1 and Micro-F1, OCC ranked first, while EoCC performed better than ECC and EBCC. In summary, OCC and EOCC achieved competitive performances in terms of four metrics. If our objective is to learn an optimizer in Exact-Match, OCC or EOCC should be the first choice.

4.3.4 Parameter Sensitivity Analysis

On k-dependence classifier chains

To evaluate the performance of model selection, an experiment is performed by kCC on four datasets, where we increase the number of parents k from 0 to $L-1$ by step 1. For convenience, the values of each metric are normalized by its maximum. Fig. 4.4 shows the experimental results in four metrics averaged by 5-fold cross validation. Consistent with the theoretical analysis, the performance of kCC in Exact-Match upgrades as the value of k increases except on the medical dataset. Moreover, as the value of k approaches $0.8\times L$, the performance becomes stable. In terms of Macro/Micro-F1, the performances share the similar

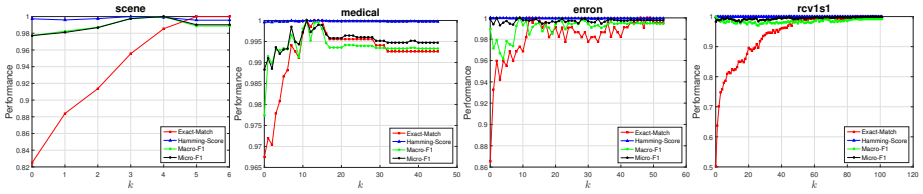


Figure 4.4: Performances of kCC on four different datasets in four evaluation metrics, whose values in each metric are normalized by its maximum. The number of parents k is increased from 0 to $L-1$ by step 1. The indices of the maximum in Exact-Match over the six datasets are 5, 11, 12 and 88, respectively.

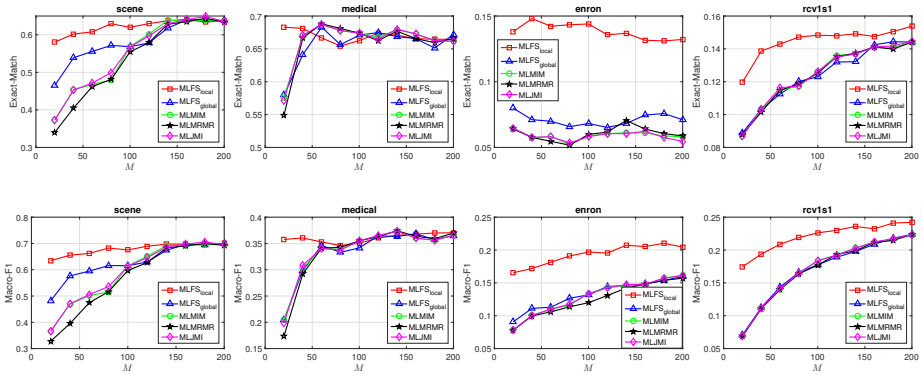


Figure 4.5: Comparing five information theoretic feature selection algorithms on four datasets in Exact-Match (the top row) and Macro-F1 (the bottom row). The number M_s of selected features is chosen from 10% to 100% by step 10% of $\min\{M, 200\}$.

tendency with Exact-Match, but the changes on values are relatively small. In Hamming-Score, its performance seems irrelevant to the change of k. Therefore, it is suggested to set the value of k as $\lceil 0.8 \times L \rceil$ if the objective is to optimize Exact-Match.

On multi-label feature selection

To evaluate the performance of multi-label feature selection, another experiment is performed by five information theoretic feature selection algorithms. The algorithms are generated according to (4.40) and (4.41) as MLMIM [77] ($\beta = \gamma = 0$ of (4.40)), MLJMI [77] and MLMRMR [65] ($\beta = 0$, $\gamma = 1/|\boldsymbol{X}_\theta|$ of (4.40), but MLMRMR ignores the conditional redundancy term.), $\text{MLFS}_{\text{local}}$ ($\beta = 1/|\boldsymbol{Pa}_j|$, $\gamma = 1/|\boldsymbol{X}_{\theta_j}|$ of (4.41)), and $\text{MLFS}_{\text{global}}$ ($\beta = 1/|\boldsymbol{Pa}_j|$, $\gamma = 1/|\boldsymbol{X}_\theta|$ of (4.40)). Fig. 4.5 shows the experimental results on four different

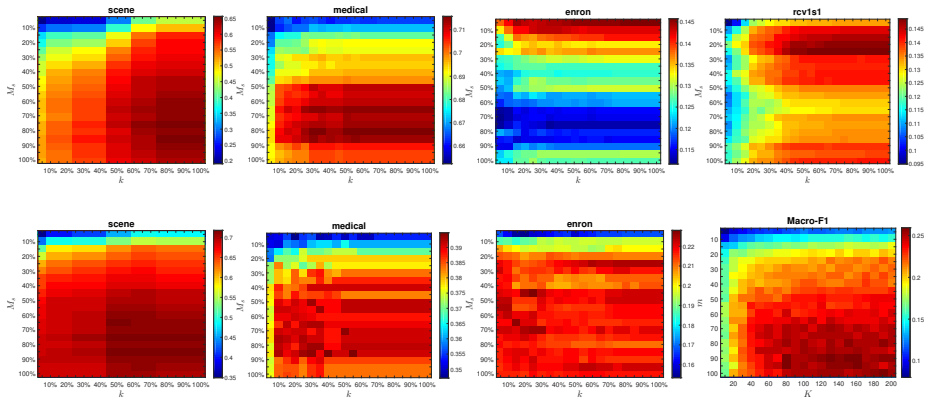


Figure 4.6: Comparison of oCC by varing the size k of parents and the number M of selected features on four different datasets in Exact-Match (the top row) and Macro-F1 (the bottom row). The values of k and M are varied by the percentages from 5% to 100% by step 5%.

datasets in Exact-Match and Macro-F1. We increase the number of features from 10% to 100% by step 10% of $\min\{M, 200\}$. The local algorithm, $\text{MLFS}_{\text{local}}$, outperformed the other ones on average, and its performance was more stable to the change of M_s compared with other algorithms. Among the global algorithms, $\text{MLFS}_{\text{global}}$ worked better than MLMIM, MLMRMR and MLJMI. Note that the performance of $\text{MLFS}_{\text{local}}$ and $\text{MLFS}_{\text{global}}$ achieved significant advantage with a smaller value of M_s, probably because of the success on modeling label correlations in (4.40). Similar pattern can be observed in other metrics. In summary, performing local feature selection is more effective than the global way, and it is important to take label correlations into account. Thus, we shall only employ $\text{MLFS}_{\text{local}}$ in the following experiments.

To evaluate the potential of OCC, we further conduct experiments by varying its two parameters: the number of parents k and the number of selected features M_s. The experimental results on the four datasets in Exact-Match and Macro-F1 are shown in Fig. 4.6. We select the number of k and M_s according to the percentages from 5% to 100% by step 5%. In terms of k, we can see that the performance becomes stable when a small percent of parents are selected. In terms of M_s, OCC achieved best performances when a fraction of features are selected. The results verify that our assumption on the existence of useless parents and features in CC, and the improvement achieved by remove such useless parents and features.

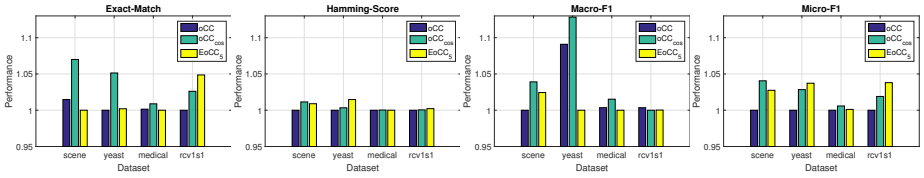


Figure 4.7: Comparison of OCC (with the canonical label order), OCC_{cos} and EOCC_5 on four datasets in four metrics. The values in each metric are normalized by dividing its minimum.

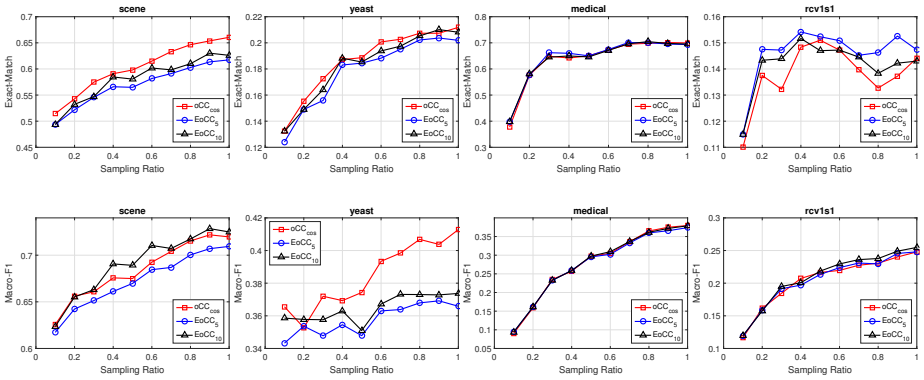


Figure 4.8: Comparison of OCC_{cos}, EOCC_5 and EOCC_{10} on four datasets in Exact-Match (the top row) and Macro-F1 (the bottom row). The sampling ratio on training instances is increased from 10% to 100% by step 10%.

4.3.5 On chain order selection

We performed another experiment to evaluate the performance of chain order selection developed in Algorithm 4. In this experiment, we incorporated Algorithm 4 with oCC (Algorithm 3), and set the two parameters of oCC as $k = L$ and $M_s = M$. To simplify the computation of Algorithm 4, we reduced the dimensionality of features to 1 by Principle Component Analysis (PCA) and ignored the second term of (4.45). We first compared OCC (with canonical label order $Y_1 \to Y_2 \to \cdots \to Y_{q-1} \to Y_q$), OCC_{cos} (with chain order selection) and EOCC_5 (ensemble of five oCCs with randomly selected orders). As shown in Fig. 4.7, comparing to the canonical order, order selection works for improving the performance of OCC to some extent, although the extent is comparable to that of ensemble of different orders in many cases.

Next, OCC_{cos} was compared with EoCC_5 and EoCC_{10}, whose subscript denotes the ensemble number of OCCs with randomly selected chain orders. The number of training instances is varied from 10% to 100% by step 10% of the total number of instances. Fig. 4.8 shows the experimental results on four

datasets in Exact-Match and Macro-F1. As shown in Fig. 4.8, although $EOCC_5$ and $EOCC_{10}$ outperform OCC_{cos} on rcv1s1 in Exact-Match, OCC_{cos} works significantly better than $EOCC_5$ on scene and yeast, and is competitive with both $EOCC_5$ and $EOCC_{10}$ on medical. In terms of processing time, EOCC cost several times more time than the compared OCC_{cos} in all cases. Therefore, in the following of this paper, we include this order selection procedure (Algorithm 4) in oCC otherwise stated elsewhere.

Part II

Multi-Label Dimension Reduction

Chapter 5

Introduction

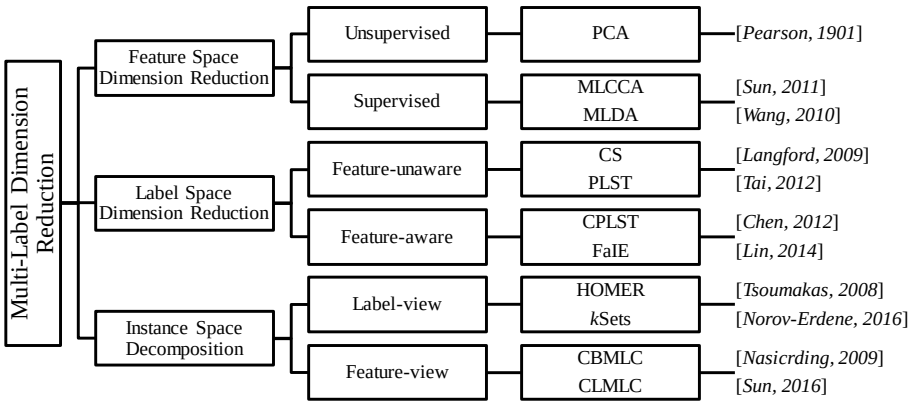
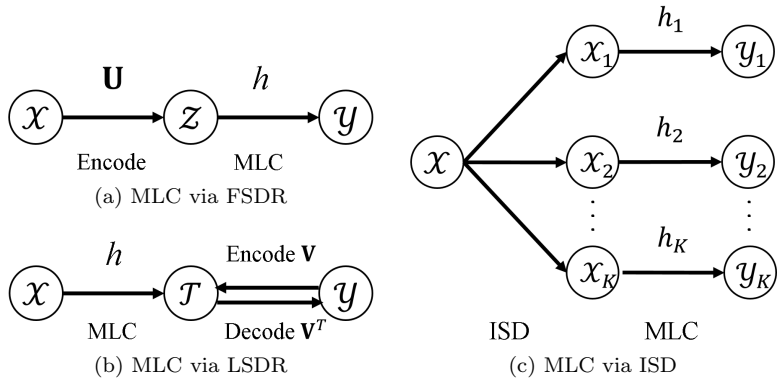


Figure 5.1: Categorization of existing ML-DR methods.

According to the objective of dimension reduction, we categorize the existing Multi-Label Dimension Reduction (ML-DR) methods into three categorizations:

- Feature Space Dimension Reduction (FS-DR)
  - Build the MLC model on the feature subspace
  - Unsupervised (PCA), Supervised (CCA [80], LDA [97], MNMTF [42], MLSF [84])
- Label Space Dimension Reduction (LS-DR)
  - By compressing the label matrix, the original problem transforms to a small number of learning tasks
  - Feature-unaware (PLST [88]), Feature-aware (CPLST [14], FaIE [53], MLLEM [43])



(a) MLC via FSDR (b) MLC via LSDR (c) MLC via ISD

Figure 5.2: The frameworks of three strategies for MLDR.

- Instance Space Decomposition (ISD)
  - Decompose the original dataset into a set of subsets, where local models are built individually
  - Label-guided (HOMER [93], k-sets [62]), Feature-guided (CBMLC [59], CLMLC [85])

For convenience, here we define the centering matrix $\mathbf{C}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\intercal$ and the the scatter (covariance) matrix $\mathbf{S}_{AB} = (\mathbf{C}_n\mathbf{A})^\intercal\mathbf{C}_n\mathbf{B} = \mathbf{A}^\intercal\mathbf{C}_n\mathbf{B}$ between arbitrary matrices $\mathbf{A} \in \mathbb{R}^{n\times m}$ and $\mathbf{B} \in \mathbb{R}^{n\times q}$. Fig. 5.2 shows the the frameworks of three strategies for MLDR.

5.1 Feature Space Dimension Reduction

The objective of FS-DR is to project the original feature space in $\mathbb{R}^M$ into a feature subspace in $\mathbb{R}^m$ $(m < M)$ by a projection matrix $\mathbf{U} \in \mathbb{R}^{M\times m}$. Such a feature projection matrix $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_m]$ with $\mathbf{u} \in \mathbb{R}^M$ is typically induced based on the input feature matrix $\mathbf{X}$ and label matrix $\mathbf{Y}$. There respentive FS-DR approaches of MLC are summarized as follows.

- Principal Component Analysis (PCA)
  - Variance maximization:

$$\max_{\mathbf{u}} \frac{\mathbf{u}^\intercal\mathbf{S}_{XX}\mathbf{u}}{\mathbf{u}^\intercal\mathbf{u}}. \tag{5.1}$$

  - The optimization problem:

$$\begin{aligned} &\max_{\mathbf{U}} \ \mathrm{Tr}(\mathbf{U}^\intercal\mathbf{S}_{XX}\mathbf{U}), \\ &\text{s.t. } \mathbf{U}^\intercal\mathbf{U} = \mathbf{I}. \end{aligned} \tag{5.2}$$

- Canonical Correlation Analysis (CCA) [80]
  - Correlation maximization:

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\intercal \mathbf{S}_{XY} \mathbf{v}}{\sqrt{\mathbf{u}^\intercal \mathbf{S}_{XX} \mathbf{u}} \sqrt{\mathbf{v}^\intercal \mathbf{S}_{YY} \mathbf{v}}}, \tag{5.3}$$

    where $\mathbf{v} \in \mathbb{R}^L$ is a label projection vector.
  - The optimization problem:

$$\begin{aligned} &\max_{\mathbf{U}} \operatorname{Tr}(\mathbf{U}^\intercal \mathbf{S}_{XY} \mathbf{S}_{YY}^{-1} \mathbf{S}_{YX} \mathbf{U}), \\ &\text{s.t. } \mathbf{U}^\intercal \mathbf{S}_{XX} \mathbf{U} = \mathbf{I}. \end{aligned} \tag{5.4}$$

- Linear Discriminant Analysis (LDA) [97]
  - Maximization of between-class to within-class covariance:

$$\max_{\mathbf{u}} \frac{\mathbf{u}^\intercal \mathbf{S}_B \mathbf{u}}{\mathbf{u}^\intercal \mathbf{S}_W \mathbf{u}}. \tag{5.5}$$

  - The optimization problem:

$$\begin{aligned} &\max_{\mathbf{U}} \operatorname{Tr}(\mathbf{U}^\intercal \mathbf{S}_B \mathbf{U}), \\ &\text{s.t. } \mathbf{U}^\intercal \mathbf{S}_W \mathbf{U} = \mathbf{I}. \end{aligned} \tag{5.6}$$

The algorithm of MLC via FS-DR is depicted in Algorithm 5, where $FSDR$ represents a FS-DR approach.

Algorithm 5 The algorithm of MLC via FS-DR

Input: $\mathbf{X} \in \mathbb{R}^{N \times M}$, $\mathbf{Y} \in \{0,1\}^{N \times L}$, $\hat{\mathbf{x}} \in \mathbb{R}^M$
Output: $\hat{\mathbf{y}} \in \{0,1\}^L$

Training:

1: $\mathbf{U} \leftarrow FSDR(\mathbf{X}, \mathbf{Y})$;
2: $\mathbf{Z} \leftarrow \mathbf{C}_N \mathbf{X} \mathbf{U}$;
3: $h : \mathbf{Z} \mapsto \mathbf{Y}$;

Testing:

4: $\hat{\mathbf{z}} = \mathbf{U}^\intercal (\hat{\mathbf{x}} - \frac{1}{N} \mathbf{X}^\intercal \mathbb{1})$;
5: $\hat{\mathbf{y}} \leftarrow h(\hat{\mathbf{z}})$.

5.2 Label Space Dimension Reduction

The objective of LS-DR is to project the original label space in $\{0,1\}^L$ into a label subspace in $\mathbb{R}^l$ ($l < L$) by a projection matrix $\mathbf{V} \in \mathbb{R}^{L \times l}$. Such a label projection matrix $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_l]$ with $\mathbf{v} \in \mathbb{R}^L$ is typically induced based on the input feature matrix $\mathbf{X}$ and label matrix $\mathbf{Y}$. Two respentive LS-DR approaches for MLC are summarized as follows.

- Principal Label Space Transformation (PLST) [88]
  - Variance maximization:

$$\max_{\mathbf{v}} \frac{\mathbf{v}^\intercal \mathbf{S}_{YY} \mathbf{v}}{\mathbf{v}^\intercal \mathbf{v}}. \tag{5.7}$$

  - The optimization problem:

$$\begin{aligned} &\max_{\mathbf{V}} \ \mathrm{Tr}(\mathbf{V}^\intercal \mathbf{S}_{YY} \mathbf{V}), \\ &\text{s.t. } \mathbf{V}^\intercal \mathbf{V} = \mathbf{I}. \end{aligned} \tag{5.8}$$

- Conditional Principal Label Space Transformation (CPLST) [14]
  - Correlation maximization:

$$\max_{\mathbf{u},\mathbf{v}} \frac{\mathbf{u}^\intercal \mathbf{S}_{XY} \mathbf{v}}{\sqrt{\mathbf{u}^\intercal \mathbf{S}_{XX} \mathbf{u}} \sqrt{\mathbf{v}^\intercal \mathbf{S}_{YY} \mathbf{v}}}. \tag{5.9}$$

  - The optimization problem:

$$\begin{aligned} &\max_{\mathbf{V}} \ \mathrm{Tr}(\mathbf{V}^\intercal \mathbf{S}_{YX} \mathbf{S}_{XX}^{-1} \mathbf{S}_{XY} \mathbf{V}), \\ &\text{s.t. } \mathbf{V}^\intercal \mathbf{V} = \mathbf{I}. \end{aligned} \tag{5.10}$$

Note that PLST and CPLST is the counterparts of PCA and CCA, respectively. The algorithm of MLC via LS-DR is depicted in Algorithm 6, where $LSDR$ represents a LS-DR approach.

Algorithm 6 The algorithm of MLC via LS-DR

Input: $\mathbf{X} \in \mathbb{R}^{N \times M}$, $\mathbf{Y} \in \{0,1\}^{N \times L}$, $\hat{\mathbf{x}} \in \mathbb{R}^M$
Output: $\hat{\mathbf{y}} \in \{0,1\}^L$

Training:

1: $\mathbf{V} \leftarrow LSDR(\mathbf{X}, \mathbf{Y})$;
2: $\mathbf{T} \leftarrow \mathbf{C}_N \mathbf{Y} \mathbf{V}$;
3: $h : \mathbf{X} \mapsto \mathbf{T}$;

Testing:

4: $\hat{\mathbf{t}} \leftarrow h(\hat{\mathbf{x}})$
5: $\hat{\mathbf{y}} = \mathrm{round}(\mathbf{V}\hat{\mathbf{t}} + \frac{1}{N}\mathbf{Y}^\intercal \mathbb{1})$

5.3 Instance Space Decomposition

We summarize two representative ISD methods as follows.

- Feature-guided ISD
  - Clustering-Based Multi-Label Classification (CBMLC) [59]

- Label-guided ISD
  - Hierarchy Of Multi-label classifiERs (HOMER) [93]

Algorithm 7 and 8 depict two ISD methods, CBMLC and HOMER, respectively. Note that, for CBMLC, $\sum_k n_k = N$ and $\sum_k l_k \geq L$; for HOMER, $\sum_k l_k = L$ and $\sum_k n_k \geq N$.

Algorithm 7 The algorithm of MLC via feature-guided ISD

Input: $\mathbf{X} \in \mathbb{R}^{N \times M}$, $\mathbf{Y} \in \{0,1\}^{N \times L}$, $\hat{\mathbf{x}} \in \mathbb{R}^M$

Output: $\hat{\mathbf{y}} \in \{0,1\}^L$

Training:

partition $\mathbf{X} \in \mathbb{R}^{N \times M}$ into $\{\mathbf{X}^k \in \mathbb{R}^{n_k \times M}\}_{k=1}^K$;

$\{[\mathbf{X}^k, \mathbf{Y}^k] \in \mathbb{R}^{n_k \times (M+l_k)}\}_{k=1}^K$, $\forall j, k$, $\mathbf{Y}^k \leftarrow \mathbf{Y} \setminus \mathbf{y}_{\cdot j}$, if $\left\|\mathbf{y}_{\cdot j}^k\right\|_1 = 0$;

$h_k : \mathbf{X}^k \rightarrow \mathbf{Y}^k$, $\forall k$;

Testing:

$k = \arg\min_{\forall k} \left\|\hat{\mathbf{x}} - \frac{1}{n_k}\mathbf{X}^{k\intercal}\mathbb{1}\right\|_2$;

$\hat{\mathbf{y}} \leftarrow h_k(\hat{\mathbf{x}})$.

Algorithm 8 The algorithm of MLC via label-guided ISD

Input: $\mathbf{X} \in \mathbb{R}^{N \times M}$, $\mathbf{Y} \in \{0,1\}^{N \times L}$, $\hat{\mathbf{x}} \in \mathbb{R}^M$

Output: $\hat{\mathbf{y}} \in \{0,1\}^L$

Training:

partition $\mathbf{Y} \in \{0,1\}^{N \times L}$ into $\{\mathbf{Y}^k \in \{0,1\}^{N \times l_k}\}_{k=1}^K$;

$[\mathbf{X}, \mathbf{Y}^{meta}] \in \mathbb{R}^{N \times (M+K)}$, where $\forall i$, $y_{ik}^{meta} = 1$ iff $\left\|\mathbf{y}_{i\cdot}^k\right\|_1 > 0$;

$\{[\mathbf{X}^k, \mathbf{Y}^k] \in \mathbb{R}^{n_k \times (M+l_k)}\}_{k=1}^K$, $\forall i, k$, $\mathbf{X}^k \leftarrow \mathbf{X} \setminus \mathbf{x}_i$, if $\left\|\mathbf{y}_{i\cdot}^k\right\|_1 = 0$;

$h^{meta} : \mathbf{X} \rightarrow \mathbf{Y}^{meta}$; $\forall k$, $h_k : \mathbf{X}^k \rightarrow \mathbf{Y}^k$;

Testing:

$\hat{\mathbf{y}}^{meta} \leftarrow h^{meta}(\hat{\mathbf{x}})$;

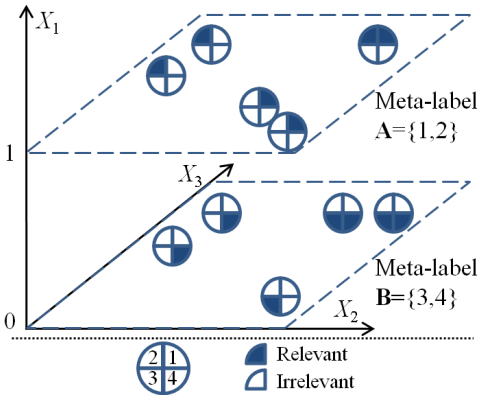
$\hat{\mathbf{y}} \leftarrow \cup h_k(\hat{\mathbf{x}})$, $\forall k \in \{k | \hat{y}_k^{meta} = 1\}$.

Chapter 6

Feature Space Dimension Reduction

6.1 Related Work

In Feature Space Dimensionality Reduction (FS-DR), a variety of traditional supervised DR approaches have been specifically extended to match the setting of MLC. There are two types of FS-DR approaches: unsupervised FS-DR and supervised FS-DR. The representative unsupervised FS-DR approach is Principal Component Analysis (PCA), transforming the features into a small number of uncorrelated variables. In MLSI [105], a supervised Latent Semantic Indexing (LSI) approach is developed to map the input features into a subspace by preserving the label information. By maximizing the feature-label dependence under the Hilbert-Schmidt independence criterion, MDDM [113] derived a closed-form solution to efficiently find the projection into the feature subspace. Originated from Nonnegative Matrix Tri-Factorization (NMTF), the Multi-label NMTF (MNMTF) [42] is proposed to take the label information into account for FS-DR by decomposing a data matrix into three factor matrices. On the other hand, traditional supervised dimension reduction approaches, such as Linear Discriminant Analysis, Canonical Correlation Analysis and Hypergraph Spectral Learning, are specifically extended to match the MLC setting [97, 80, 79]. On the other hand, in order to improve the discriminative ability for each label or meta-label, LIFT [108], LLSF [38] and MLSF [84], are proposed to extract the label-specific or meta-label-specifc features.



No.	Features			Labels			
	X_1	X_2	X_3	Y_1	Y_2	Y_3	Y_4
1	1	1.0	0.1	1	0	0	0
2	1	0.9	0.2	1	0	0	0
3	1	0.8	0.8	1	1	0	0
4	1	0.2	0.8	0	1	0	0
5	1	0.1	0.7	0	1	0	0
6	0	0.9	0.1	0	0	1	0
7	0	0.7	0.8	0	0	1	1
8	0	0.9	0.8	0	0	1	1
9	0	0.2	0.8	0	0	0	1
10	0	0.1	0.7	0	0	0	1

Figure 6.1: Meta-labels with specific feature subsets.

6.2 Mining Meta-Label-Specific Features (MLSF)

In this study, we propose an MLC method based on two assumptions: (a) meta-labels, strong and reasonable label combinations, exist implicitly in the label space; (b) only a fraction of features is relevant to a meta-label, and different meta-labels relate with different feature subsets. Such assumptions hold in several real-world observations. For example, in image annotation, the tags "ocean" and "sky" can be viewed as forming a meta-label, since they highly correlates and shares similar color features; in text categorization, the topics "science" and "technology" have strong correlation with specific features, like "research", "laboratory", "institute", etc. Fig. 6.1 shows a toy multi-label example satisfying the assumptions. In Fig. 6.1, two meta-labels $\boldsymbol{A} = \{1, 2\}$ and $\boldsymbol{B} = \{3, 4\}$ can be found by preserving the strong label dependency within each meta-label. In addition, mining meta-label-specific features is also interesting, since feature X_1 is useful for separating meta-labels, but is useless for classification inside a meta-label.

In order to justify the assumptions, we propose a novel MLC method using Meta-Label-Specific Features (MLSF). MLSF consists of meta-label learning and specific features mining. In meta-label learning, highly correlated labels are grouped together using the information from both the label and instance spaces. We discuss the usage of the Spectral Clustering [6] technique. In specific feature selection, we use the LASSO [90] with an efficient optimization approach, Alternating Direction Method of Multipliers (ADMM) [9]. To capture label correlations in each meta-label, Classifier Chains (CC) [69] is built on the meta-label-specific features. To evaluate the performance of MLSF, extensive experiments are conducted on twelve multi-label datasets.

6.2.1 Meta-Label Learning

In this section, we embed label correlations into meta-labels in such a way that the member labels in a meta-label share strong dependency with each other but have weak dependency with the other non-member labels. To this end, we construct a graph $\boldsymbol{G} =< \boldsymbol{V}, \boldsymbol{E} >$ in the label space, where $\boldsymbol{V}$ denotes the vertex/label set, and $\boldsymbol{E}$ is the edge set containing edges between each label pair. Given an appropriate affinity matrix $\mathbf{A}$ on $\boldsymbol{E}$, meta-label learning can be considered as a graph cut problem: cutting the graph $\boldsymbol{G}$ into a set of sub-graphs.

For constructing affinity matrix $\mathbf{A}$, we use two different sources: the label space and the instance space. To model the affinity obtained from the label space, Jaccard index, a metricated variant of mutual information, is used:

$$A_{jk}^{(L)} \coloneqq \frac{\sum_{i=1}^{N} y_{ij} y_{ik}}{\sum_{i=1}^{N} (y_{ij} + y_{ik} - y_{ij} y_{ik})}. \tag{6.1}$$

Next, focusing on the instance space, we have

$$A_{jk}^{(I)} \coloneqq \mathrm{e}^{-\|\boldsymbol{\mu}_j - \boldsymbol{\mu}_k\|_2^2}, \quad \text{where } \boldsymbol{\mu}_j = \frac{\sum_{i=1}^{N} y_{ij} \mathbf{x}_i}{\sum_{i=1}^{N} y_{ij}}. \tag{6.2}$$

Last, by ϵ-neighborhood, we combine these two matrices into one the affinity matrix $\mathbf{A} = \{A_{jk}\}_{j,k=1}^{L}$ as follows,

$$A_{jk} \coloneqq \begin{cases} \alpha A_{jk}^{(L)} + (1-\alpha) A_{jk}^{(I)} & (A_{jk} > \epsilon) \\ 0 & (A_{jk} \leq \epsilon), \end{cases} \tag{6.3}$$

where $\alpha \in [0, 1]$ is a balance factor.

By regarding $\mathbf{A}$ as the edge weight matrix on $\boldsymbol{E}$, $\boldsymbol{G}$ becomes a graph representation of the label space. Then, to cut $\boldsymbol{G}$ into K sub-graphs, i.e., K meta-labels, is equivalent to perform k-means on $\mathbf{U} \in \mathbb{R}^{N \times K}$, which can be solved by the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{U}} \ \mathrm{Tr}(\mathbf{U}^{\intercal}(\mathbf{D} - \mathbf{A})\mathbf{U}) \\ &\text{s.t. } \mathbf{U}^{\intercal}\mathbf{D}\mathbf{U} = \mathbf{I}, \end{aligned} \tag{6.4}$$

where $\mathbf{D}$ is the diagonal degree matrix, $\mathbf{D} = (D_{jj}) = (\sum_k A_{jk})$, and $\mathbf{L}$ denotes the Laplacian matrix, $\mathbf{L} = \mathbf{D} - \mathbf{A}$. Applying k-means on the rows of $\mathbf{U}$, we obtain the meta-label membership vector $\mathbf{m} = \{m_j\}_{j=1}^{L} \in \{1, ..., K\}^L$, where $m_j = k$ indicates the jth label belongs to the kth meta-label. The pesudo code of meta-label learning is depicted in Algorithm 9.

6.2.2 Meta-Label-Specific Feature Selection

Next, we find meta-label-specific feature subsets. For this end, we transform $\mathbf{Y} \in \{0, 1\}^{N \times L}$ into the meta-label matrix $\mathbf{Z} = [\mathbf{z}_1, ..., \mathbf{z}_K] \in \mathbb{Z}^{N \times K}$. Here $\mathbf{Y}$ is

Algorithm 9 The algorithm of meta-label learning

Input: $\mathbf{X}$: feature matrix, $\mathbf{Y}$: label matrix, α, ϵ, K: parameters

Output: $\mathbf{m}$: meta-label membership vector, $\mathbf{m} \in \{1, ..., K\}^L$

1: Compute $\mathbf{A}$ by (8.10) according to α and ϵ;
2: $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D} = \text{diag}(\sum_k A_{jk})$;
3: Solve $\mathbf{Lu} = \lambda\mathbf{Du}$ by K smallest eigenvalues;
4: $\mathbf{m} \leftarrow k$-means$(\mathbf{U}, K)$, where $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_K]$.

Algorithm 10 The algorithm of specific feature selection

Input: $\mathbf{X}$: feature matrix, $\mathbf{Y}$: label matrix, $\mathbf{m}$: meta-label membership, γ, ρ: parameters

Output: $\mathbf{V}$: regression parameter matrix

1: **for** $k \in \{1, ..., K\}$ **do**
2: $\quad \mathbf{Z}(:, k) \leftarrow \text{bi2de}(\mathbf{Y}(:, \mathbf{m}==k))$;
3: $\mathbf{V} \coloneqq \mathbf{0}$, $\mathbf{\Lambda} \coloneqq \mathbf{0}$;
4: **repeat**
5: $\quad \mathbf{W} \leftarrow (\mathbf{X}^\intercal\mathbf{X} + \rho\mathbf{I})^{-1}(\mathbf{X}^\intercal\mathbf{Z} + \rho\mathbf{V} - \mathbf{\Lambda})$;
6: $\quad \mathbf{V} \leftarrow \mathbf{W} + \mathbf{\Lambda}/\rho$;
7: $\quad V_j = \text{sign}(V_j) \cdot \max(0, |V_j| - \gamma/\rho), \forall j$;
8: $\quad \mathbf{\Lambda} \leftarrow \mathbf{\Lambda} + \rho(\mathbf{W} - \mathbf{V})$;
9: **until** Convergence

firstly partitioned into K parts by $\mathbf{m}$, then each part is encoded into a meta-label vector $\mathbf{z}$ by converting binary to decimal. Hence, we can use multivariate linear regression with ℓ_1-norm regularization. That is, LASSO [90], whose objective function is given by

$$\min_{\mathbf{W}} \frac{1}{2} \|\mathbf{XW} - \mathbf{Z}\|_F^2 + \gamma \|\mathbf{W}\|_1 , \tag{6.5}$$

where γ controls the sparsity of parameter matrix $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_K] \in \mathbb{R}^{D \times K}$. Here we treat the indexes of nonzero elements of $\mathbf{w}_k$ as the indexes of meta-label specific features for the kth meta-label.

Lasso regression is a convex optimization problem, so it looks easy to solve. However, it is not trivial to efficiently optimize the objective function due to the non smoothness resulting from the ℓ_1-norm. In this study, we employ the Alternating Direction Method of Multiplier (ADMM) [9] to separate (6.5) into two sub-problems, which could be efficiently addressed. By employing a dummy variable matrix $\mathbf{V} \in \mathbb{R}^{M \times K}$ into (6.5), it can be rewritten in the augmented Lagragian form $L(\mathbf{W}, \mathbf{V}, \mathbf{\Lambda})$:

$$\frac{1}{2} \|\mathbf{XW} - \mathbf{Z}\|_F^2 + \gamma \|\mathbf{V}\|_1 + \frac{\rho}{2} \|\mathbf{W} - \mathbf{V}\|_F^2 + \text{vec}(\mathbf{\Lambda})^\intercal \text{vec}(\mathbf{W} - \mathbf{V}), \tag{6.6}$$

Algorithm 11 The algorithm of MLSF

Input: $\mathbf{X}$: feature matrix, $\mathbf{Y}$: label matrix, $\hat{\mathbf{x}}$: test instance, α, ϵ, K, γ, ρ: parameters

Output: $\hat{\mathbf{y}}$: prediction label set

Training:

1: $\mathbf{m} \leftarrow$ <Algorithm 9>$(\mathbf{X}, \mathbf{Y}, \alpha, \epsilon, K)$;
2: $\mathbf{V} \leftarrow$ <Algorithm 10>$(\mathbf{X}, \mathbf{Y}, \mathbf{m}, \gamma, \rho)$;
3: **for** $k \in \{1, ..., K\}$ **do**
4: $\quad \mathbf{h}_k : \mathbf{X}(:, \mathbf{v}_k \neq 0) \mapsto \mathbf{Y}(:, \mathbf{m} == k)$;

Testing:

5: **for** $k \in \{1, ..., K\}$ **do**
6: $\quad \hat{\mathbf{y}}(\mathbf{m} == k)) \leftarrow \mathbf{h}_k(\hat{\mathbf{x}}(\mathbf{v}_k \neq 0))$;

where $\mathbf{\Lambda} \in \mathbb{R}^{M \times K}$ is the Lagrange multiplier matrix. ADMM performs the following iterations to optimize (6.6):

$$\mathbf{W}^{t+1} = \arg\min_{\mathbf{W}} L(\mathbf{W}, \mathbf{V}^t, \mathbf{\Lambda}^t), \tag{6.7}$$

$$\mathbf{V}^{t+1} = \arg\min_{\mathbf{V}} L(\mathbf{W}^{t+1}, \mathbf{V}, \mathbf{\Lambda}^t), \tag{6.8}$$

$$\mathbf{\Lambda}^{t+1} = \mathbf{\Lambda}^t + \rho(\mathbf{W}^{t+1} - \mathbf{V}^{t+1}). \tag{6.9}$$

In this way, ADMM separates the original optimization problem (6.5) into two sub-problems (6.7) and (6.8). Specifically, (6.7) is simply addressed by ridge regression, while (6.8) can be solved by the soft thresholding technique. After the convergence, $\mathbf{V}$ instead of $\mathbf{W}$ will be used as the sparse matrix indicating meta-label-specific features. The pesudo code of specific feature selection is given in Algorithm 10.

Discussion

After meta-label learning and specific features mining, in order to capture the correlations preserved in meta-labels, we employ a multi-label classifier, such as PairWise (PW) [30], Label Powerset (LP) [94] and Classifier Chains (CC) [69]. In this study CC is constructed for each meta-label, because CC can efficiently captures label dependency by randomly building a fully-connected Bayesian network in the label space.

Algorithm 11 illustrates the complete procedure of MLSF. In the training phase (Steps 1 to 4), MLSF firstly finds meta-label membership $\mathbf{m}$ to know which label belongs to which meta-label and regression matrix $\mathbf{V}$ by Steps 1 and 2; Then a CC classifier $\mathbf{h}_k$ is built on the data matrix with specific features for the kth meta-label, $k \in \{1, ..., K\}$, by Steps 3 and 4. In the testing phase (Steps 5 and 6), a test instance with meta-label-specific features is feed to each

h for the prediction on corresponding meta-label.

In time complexity of MLSF, Step 1 has $O(KL^2 + i_1K^2L)$ and Step 2 has $O(i_2KM^2)$ in Algorithm 11, where i_1 and i_2 are the number of iterations in k-means and ADMM, respectively. The complexity in Steps 3 and 4 is $O(KT(mNl))$, where $T(\cdot)$ denotes the complexity of the classifier **h**, while m ($m < D$) and l ($l < L$) is the number of specific features and labels of each meta-label, respectively.

6.3 Robust Semi-Supervised Dimension Reduction (READER)

With the rapid increase of web-related applications, more and more multi-label datasets emerge in high-dimensionality. Such high-dimensionality of multi-label data significantly increases the time and space complexity in learning, and degrades the classification performance due to the possible existence of noisy features and labels. Previous studies have demonstrated that only a subset of high-dimensional features, i.e., discriminative features, are useful for the learning process. In addition, irrelevant and redundant features would negatively influence the classification performance. Thus it is necessary to apply feature selection on high-dimensional data as an effective pre-process, bringing in less time and space cost on classification and better performance and interpretation.

However, it is a non-trivial thing to conduct traditional feature selection algorithms on multi-label data due to its intrinsic properties. First, the labels in multi-label datasets are probably correlated and dependent with each other, thus it is important to model label correlations during feature selection. One simple example is that, in semantic image annotation, the concepts "lake" and "reflection" share a strong correlation, therefore common features should be selected in order to model the correlation. Second, the existence of noisy labels (outliers) and incomplete labels in multi-label data should be considered. Such noisy outliers, usually resulting from the mistakes in the label annotation by human beings, would misguide the selection of discriminative features. Third, a large part of training data is unlabeled in various real-world applications. It is intractable to annotate each data point with multiple labels from a huge number of instances and candidate labels. Although numerous methods have been proposed for multi-label dimension reduction, most of them focus only on solving one of the three problems, preventing from selecting most discriminative features and thus limiting the classification performance.

To cope with all the three aforementioned problems, we propose a novel method named **R**obust s**E**mi-supervised multi-l**A**bel **D**im**E**nsion **R**eduction (READER) [86] from the viewpoint of empirical risk minimization. Specifically, the loss function and sparsity regularization term in $\ell_{2,1}$-norm make READER

robust against data outliers and able to jointly select features across labels. In addition, rather than the original label space, a low-dimensional latent space found by non-linear embedding is used to select discriminative features. Note that such a label embedding saves label correlations and alleviates the negative effect of imperfect label information. Moreover, manifold learning is applied to select features where originally neighbor instances keep close to each other. In this way, READER enables to utilize a large amount of unlabeled data to improve its performance. To optimize the objective function, we transform the optimization problem into a generalized eigenvalue problem, and develop an efficient algorithm that successfully converges to the global optimum.

6.3.1 Proposed Formulation

Suppose that in the setting of semi-supervised MLC, we have N training instances $\tilde{\mathbf{X}} \in \mathbb{R}^{N \times M}$, where only n ($< N$) instances $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^\intercal \in \mathbb{R}^{n \times M}$ are associated with labels $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_n]^\intercal \in \{0, 1\}^{n \times L}$. The $\ell_{2,1}$-norm of $\mathbf{Z} \in \mathbb{R}^{M \times n}$ is defined as $\|\mathbf{Z}\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{M} \mathbf{Z}_{ij}^2} = \sum_{i=1}^{n} \|\mathbf{Z}_{i\cdot}\|_2$.

We can formulate in general the MLC problems via empirical risk minimization in the following objective function:

$$\min_{h} \sum_{i=1}^{n} loss(h(\mathbf{x}_i), \mathbf{y}_i) + \alpha \Omega(h), \tag{6.10}$$

where $loss(\cdot)$ denotes a loss function and $\Omega(h)$ is the regularization term on the multi-label classifier h. In practice, there are several available loss functions (ℓ_2 loss, hinge loss, logistic loss) and regularization terms (lasso regularization, ridge regularization). Similar with [61], we apply $\ell_{2,1}$-norm based loss function and regularization in order to conduct *robust feature selection*.

$$\begin{aligned} &\min_{\mathbf{W}} \sum_{i=1}^{n} \|\mathbf{W}^\intercal \mathbf{x}_i - \mathbf{y}_i\|_2 + \alpha \sum_{j=1}^{M} \|\mathbf{W}_{j\cdot}\|_2 \\ =&\min_{\mathbf{W}} \|\mathbf{X}\mathbf{W} - \mathbf{Y}\|_{2,1} + \alpha \|\mathbf{W}\|_{2,1}, \end{aligned} \tag{6.11}$$

where $\mathbf{W} \in \mathbb{R}^{M \times L}$ is a projection matrix, the ℓ_2 norm of whose row $\|\mathbf{W}_{i\cdot}\|_2$ ($\forall i$) indicates the importance of ith feature. The $\ell_{2,1}$ norm based loss function makes the outliers less important than the least square loss (Frobenius norm). In addition, the $\ell_{2,1}$ based regularization term assures the row-sparsity of $\mathbf{W}$, enabling to couple feature selection across multiple labels.

To utilize the $(N - n)$ unlabeled training instances, we assume that the similar instances $\mathbf{x}_i$ and $\mathbf{x}_j$ in original feature space shall keep close to each other in the projected feature subspace $\mathbf{W}^\intercal \mathbf{x}_i$ and $\mathbf{W}^\intercal \mathbf{x}_j$. With all N instances,

thus we have,

$$\begin{aligned} &\min_{\mathbf{W}} \sum_{i,j=1}^{N} (\mathbf{S}_x)_{ij} \left\| \mathbf{W}^\intercal \mathbf{x}_i - \mathbf{W}^\intercal \mathbf{x}_j \right\|_2^2, \\ &\text{s.t. } \mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{D}_x \tilde{\mathbf{X}} \mathbf{W} = \mathbf{I}, \end{aligned} \tag{6.12}$$

where $\mathbf{S}_x$ is the similarity matrix whose element $(\mathbf{S}_x)_{ij}$ measures the similarity score between $\mathbf{x}_i$ and $\mathbf{x}_j$, and $\mathbf{D}_x$ is the degree matrix with diagonal element $(\mathbf{D}_x)_{ii} = \sum_j (\mathbf{S}_x)_{ij}$.

$$(\mathbf{S}_x)_{ij} \coloneqq \begin{cases} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{\sigma^2}\right) & (\mathbf{x}_i \in \mathcal{N}_p(\mathbf{x}_j) \vee \mathbf{x}_j \in \mathcal{N}_p(\mathbf{x}_i)) \\ 0 & (\text{otherwise}), \end{cases} \tag{6.13}$$

where $\mathcal{N}_p(\mathbf{x}_i)$ is the p-nearest neighbors of $\mathbf{x}_i$, $\forall i$. Thus the *semi-supervised* version of (6.11) is induced by adding (6.12),

$$\begin{aligned} &\min_{\mathbf{W}} \left\| \mathbf{X}\mathbf{W} - \mathbf{Y} \right\|_{2,1} + \alpha \left\| \mathbf{W} \right\|_{2,1} + \beta \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} \mathbf{W}), \\ &\text{s.t. } \mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{D}_x \tilde{\mathbf{X}} \mathbf{W} = \mathbf{I}, \end{aligned} \tag{6.14}$$

where $\mathbf{L}_x = \mathbf{D}_x - \mathbf{A}_x \in \mathbb{R}^{N \times N}$ is the Laplacian matrix on $\tilde{\mathbf{X}} \in \mathbb{R}^{N \times M}$.

In order to capture the *label correlations* in multi-label feature selection, we apply a non-linear label embedding to produce a k-dimensional latent label space $\mathbf{V} \in \mathbb{R}^{n \times k}$ from $\mathbf{Y} \in \{0,1\}^{n \times L}$, $k < L$. The label embedding aims to preserve the neighborhood structure of original labels in the latent subspace, which is implemented by manifold learning,

$$\min_{\mathbf{V}} \sum_{i,j=1}^{n} (\mathbf{S}_y)_{ij} \left\| \mathbf{V}_{i\cdot} - \mathbf{V}_{j\cdot} \right\|_2^2. \text{ s.t. } \mathbf{V}^\intercal \mathbf{D}_y \mathbf{V} = \mathbf{I}, \tag{6.15}$$

where $(\mathbf{D}_y)_{ii} = \sum_j (\mathbf{S}_y)_{ij}$, $\forall i$. $\mathbf{S}_y$ denotes the similarity matrix, and $(\mathbf{S}_y)_{ij}$ measures the similarity between $\mathbf{y}_i$ and $\mathbf{y}_j$,

$$(\mathbf{S}_y)_{ij} \coloneqq \begin{cases} \frac{<\mathbf{y}_i, \mathbf{y}_j>}{\|\mathbf{y}_i\|_2 \|\mathbf{y}_j\|_2} & (\mathbf{y}_i \in \mathcal{N}_p(\mathbf{y}_j) \vee \mathbf{y}_j \in \mathcal{N}_p(\mathbf{y}_i)) \\ 0 & (\text{otherwise}), \end{cases} \tag{6.16}$$

where $\mathcal{N}_p(\mathbf{y}_i)$ denotes the p-nearest neighbors of $\mathbf{y}_i$, $\forall i$. Thus the label correlations are captured in the following optimization problem,

$$\begin{aligned} &\min_{\mathbf{W},\mathbf{V}} \left\| \mathbf{X}\mathbf{W} - \mathbf{V} \right\|_{2,1} + \alpha \left\| \mathbf{W} \right\|_{2,1} + \beta \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} \mathbf{W}) + \gamma \mathrm{Tr}(\mathbf{V}^\intercal \mathbf{L}_y \mathbf{V}), \\ &\text{s.t. } \mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{D}_x \tilde{\mathbf{X}} \mathbf{W} = \mathbf{V}^\intercal \mathbf{D}_y \mathbf{V} = \mathbf{I}, \end{aligned} \tag{6.17}$$

where $\mathbf{L}_y = \mathbf{D}_y - \mathbf{S}_y \in \mathbb{R}^{n \times n}$ is the Laplacian matrix on $\mathbf{X} \in \mathbb{R}^{n \times M}$. Note that here we have $\mathbf{W} \in \mathbb{R}^{M \times k}$.

Once obtaining $\mathbf{W}$, we can select the most discriminative features by the row-sparsity of $\mathbf{W}$ originating from the $\ell_{2,1}$-norm. Since in practice many rows of the optimal $\mathbf{W}$ are close to rather than equal to 0, we rank features according to $\|\mathbf{W}_{j\cdot}\|_2$ $(\forall j)$ in a descending order, and feed the top ranked features to the subsequent learning process.

Remarks

Although the proposed READER method is designed for *Feature Space Dimension Reduction* (FS-DR), it is worth noting that READER is easily extended for *Label Space Dimension Reduction* (LS-DR) [14]. To this end, we introduce a linear projection $\mathbf{P} \in \mathbb{R}^{L\times k}$ to approximate the non-linear label embedding $\mathbf{V}$ in (6.17), i.e., $\mathbf{V} = \mathbf{YP}$. In addition, in order to make round-based decoding[1] available, we relax the constraint $\mathbf{P}^\intercal\mathbf{Y}^\intercal\mathbf{D}_y\mathbf{YP} = \mathbf{I}$ as $\mathbf{P}^\intercal\mathbf{P} = \mathbf{I}$. Therefore, for LS-DR, (6.17) becomes

$$\begin{aligned}&\min_{\mathbf{W},\mathbf{P}} \ \|\mathbf{XW} - \mathbf{YP}\|_{2,1} + \alpha\|\mathbf{W}\|_{2,1} + \beta\mathrm{Tr}(\mathbf{W}^\intercal\tilde{\mathbf{X}}^\intercal\mathbf{L}_x\tilde{\mathbf{X}}\mathbf{W}) + \gamma\mathrm{Tr}(\mathbf{P}^\intercal\mathbf{Y}^\intercal\mathbf{L}_y\mathbf{YP}),\\ &\text{s.t. } \mathbf{W}^\intercal\tilde{\mathbf{X}}^\intercal\mathbf{D}_x\tilde{\mathbf{X}}\mathbf{W} = \mathbf{P}^\intercal\mathbf{P} = \mathbf{I}.\end{aligned} \tag{6.18}$$

Thus, after obtaining $\mathbf{W}$ and $\mathbf{P}$, the prediction $\hat{\mathbf{y}}$ on a text instance $\hat{\mathbf{x}}$ is made by $\hat{\mathbf{y}} \leftarrow \mathrm{round}(\mathbf{PW}^\intercal\hat{\mathbf{x}})$.

6.3.2 Optimization Algorithm

Theorem 4. *By the setting the followings,*

$$\begin{aligned}\mathbf{A} &\coloneqq \begin{bmatrix}\mathbf{X} & -\mathbf{I}_n\\ \alpha\mathbf{I}_M & \mathbf{0}\end{bmatrix},\ \mathbf{B} \coloneqq \begin{bmatrix}\beta\tilde{\mathbf{X}}^\intercal\mathbf{L}_x\tilde{\mathbf{X}} & \mathbf{0}\\ \mathbf{0} & \gamma\mathbf{L}_y\end{bmatrix},\\ \mathbf{E} &\coloneqq \begin{bmatrix}\tilde{\mathbf{X}}^\intercal\mathbf{D}_x\tilde{\mathbf{X}} & \mathbf{0}\\ \mathbf{0} & \mathbf{D}_y\end{bmatrix},\ \mathbf{U} \coloneqq \begin{bmatrix}\mathbf{W}\\ \mathbf{V}\end{bmatrix},\end{aligned} \tag{6.19}$$

and $\mathbf{F} = \mathbf{A}^\intercal\mathbf{HA} + \mathbf{B}$, *where* $\mathbf{H}$ *is a diagonal matrix with* i*th diagonal element* $\mathbf{H}_{ii} = \frac{1}{2\|(\mathbf{AU})_{i\cdot}\|_2}$, *the optimization problem in (6.17) is equivalent to the generalized eigenvalue problem,*

$$\mathbf{FU} = \mathbf{EU}\tilde{\mathbf{\Lambda}}, \tag{6.20}$$

where $\tilde{\mathbf{\Lambda}}$ *is a diagonal matrix with diagonal element* $\tilde{\lambda}$. *Solving (6.20) produces eigenvectors* $\mathbf{u}_1, ..., \mathbf{u}_k$ *with eigenvalues* $\tilde{\lambda}_1 \le ... \le \tilde{\lambda}_k$. *Thus we have the solution* $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_k] \in \mathbb{R}^{(n+M)\times k}$.

[1] round-based decoding maps the component of the vector to the closet value in $\{0, 1\}$, denoted by $\mathrm{round}(\cdot)$.

Proof. The formula (6.17) is equivalent to,

$$\begin{aligned}\min_{\mathbf{W},\mathbf{V}} \left\| \begin{bmatrix} \mathbf{X}\mathbf{W} - \mathbf{V} \\ \alpha\mathbf{W} \end{bmatrix} \right\|_{2,1} + \mathrm{Tr}(\beta\mathbf{W}^\intercal\tilde{\mathbf{X}}^\intercal\mathbf{L}_x\tilde{\mathbf{X}}\mathbf{W} + \gamma\mathbf{V}^\intercal\mathbf{L}_y\mathbf{V}),\\ \text{s.t. } \mathbf{W}^\intercal\tilde{\mathbf{X}}^\intercal\mathbf{D}_x\tilde{\mathbf{X}}\mathbf{W} = \mathbf{V}^\intercal\mathbf{D}_y\mathbf{V} = \mathbf{I}. \end{aligned} \tag{6.21}$$

The problem in (6.21) can be rewritten as

$$\begin{aligned}\min_{\mathbf{W},\mathbf{V}} \left\| \begin{bmatrix} \mathbf{X} & -\mathbf{I}_n \\ \alpha\mathbf{I}_M & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{W} \\ \mathbf{V} \end{bmatrix} \right\|_{2,1} + \mathrm{Tr}\left(\begin{bmatrix} \mathbf{W} \\ \mathbf{V} \end{bmatrix}^\intercal \begin{bmatrix} \beta\tilde{\mathbf{X}}^\intercal\mathbf{L}_x\tilde{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \gamma\mathbf{L}_y \end{bmatrix} \begin{bmatrix} \mathbf{W} \\ \mathbf{V} \end{bmatrix} \right)\\ \text{s.t. } \begin{bmatrix} \mathbf{W} \\ \mathbf{V} \end{bmatrix}^\intercal \begin{bmatrix} \tilde{\mathbf{X}}^\intercal\mathbf{D}_x\tilde{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_y \end{bmatrix} \begin{bmatrix} \mathbf{W} \\ \mathbf{V} \end{bmatrix} = \mathbf{I}_{n+M}. \end{aligned} \tag{6.22}$$

According to the definitions in (6.19), (6.22) becomes

$$\begin{aligned}\min_{\mathbf{U}} \ \|\mathbf{A}\mathbf{U}\|_{2,1} + \mathrm{Tr}(\mathbf{U}^\intercal\mathbf{B}\mathbf{U}),\\ \text{s.t. } \mathbf{U}^\intercal\mathbf{E}\mathbf{U} = \mathbf{I}. \end{aligned} \tag{6.23}$$

Here we relax $\|\mathbf{A}\mathbf{U}\|_{2,1}$ as $\mathrm{Tr}(\mathbf{U}^\intercal\mathbf{A}^\intercal\mathbf{H}\mathbf{A}\mathbf{U})$, where $\mathbf{H}$ is a diagonal matrix with its ith diagonal element $\mathbf{H}_{ii} = \frac{1}{2\|(\mathbf{A}\mathbf{U})_{i\cdot}\|_2}$, $\forall i$. Thus, by $\mathbf{F} = \mathbf{A}^\intercal\mathbf{H}\mathbf{A} + \mathbf{B}$, (6.23) is rewritten as

$$\begin{aligned}\min_{\mathbf{U}} \ \mathrm{Tr}(\mathbf{U}^\intercal\mathbf{F}\mathbf{U}),\\ \text{s.t. } \mathbf{U}^\intercal\mathbf{E}\mathbf{U} = \mathbf{I}. \end{aligned} \tag{6.24}$$

In fact, (6.24) is equivalent to the generalized eigenvalue problem in (6.20). □

However, the computation on the smallest eigenvalues of (6.20) is unstable. Thus, by setting $\lambda = 1/\tilde{\lambda}$ and diagonal matrix $\mathbf{\Lambda}$ with $\Lambda_{ii} = \lambda_i$, (6.20) is equivalent to

$$\mathbf{E}\mathbf{U} = \mathbf{F}\mathbf{U}\mathbf{\Lambda}, \tag{6.25}$$

where we are interested in the k eigenvectors with largest eigenvalues. According to (6.25), it seems that $\mathbf{U}$ can be computed by directly solving the eigenvalue problem. However, $\mathbf{F}$ depends on $\mathbf{U}$, which is also unknown. In this paper, we propose an iterative algorithm, Algorithm 12, to obtain $\mathbf{U}$, and prove that Algorithm 12 will converge to the global optimum, since (6.24) is a convex optimization problem.

Algorithm Analysis

We shall show that the iterative algorithm in Algorithm 12 by which the objective function in (6.17) monotonically decreases in each iteration and converges to the global optimum.

Algorithm 12 Optimization algorithm for READER

Input: $\tilde{\mathbf{X}} \in \mathbb{R}^{N\times M}$, $\mathbf{X} \in \mathbb{R}^{n\times M}$, $\mathbf{Y} \in \mathbb{B}^{n\times l}$

Output: Top ranked features

1: Calculate $\mathbf{D}_x$, $\mathbf{D}_y$, $\mathbf{L}_x$ and $\mathbf{L}_y$;
2: Calculate $\mathbf{A}$, $\mathbf{B}$, $\mathbf{E}$ according to (6.19);
3: Set $t = 0$, and initialize $\mathbf{H}^t = \mathbf{I}$;
4: **repeat**
5: $\quad \mathbf{F}^t \leftarrow \mathbf{A}^\intercal \mathbf{H}^t \mathbf{A} + \mathbf{B}$;
6: $\quad$ Solve $\mathbf{E}\mathbf{U}^t = \mathbf{F}^t \mathbf{U}^t \Lambda$ with k largest eigenvalues;
7: $\quad \forall i,\ \mathbf{H}_{ii}^{t+1} \leftarrow \frac{1}{2\|(\mathbf{A}\mathbf{U}^t)_{i\cdot}\|_2}$;
8: $\quad t \leftarrow t + 1$;
9: **until** *Convergence*
10: Return top ranked features in descending order of $\|\mathbf{W}_{j\cdot}\|_2$ $(\forall j)$.

Theorem 5. *In Algorithm 12 (Steps 4 to 9) the objective function in (6.17) does not increase in each iteration.*

Proof. Since solving the generalized eigenvalue problem of $\mathbf{E}\mathbf{U} = \mathbf{F}\mathbf{U}\Lambda$ in Step 6 of Algorithm 12 is equivalent to solving the optimization problem in (6.24), in the t iteration we have

$$\mathbf{U}_{t+1} = \arg\min_{\mathbf{U}} \mathrm{Tr}(\mathbf{U}^\intercal \mathbf{F}_t \mathbf{U}),\ \text{s.t. } \mathbf{U}^\intercal \mathbf{E} \mathbf{U} = \mathbf{I}, \tag{6.26}$$

indicating that

$$\mathrm{Tr}(\mathbf{U}_{t+1}^\intercal \mathbf{F}_t \mathbf{U}_{t+1}) \leq \mathrm{Tr}(\mathbf{U}_t^\intercal \mathbf{F}_t \mathbf{U}_t). \tag{6.27}$$

According to $\mathbf{F} = \mathbf{A}^\intercal \mathbf{H} \mathbf{A} + \mathbf{B}$, (6.27) becomes

$$\mathrm{Tr}(\mathbf{U}_{t+1}^\intercal \mathbf{A}^\intercal \mathbf{H}_t \mathbf{A} \mathbf{U}_{t+1}) + \mathrm{Tr}(\mathbf{U}_{t+1}^\intercal \mathbf{B} \mathbf{U}_{t+1}) \leq \mathrm{Tr}(\mathbf{U}_t^\intercal \mathbf{A}^\intercal \mathbf{H}_t \mathbf{A} \mathbf{U}_t) + \mathrm{Tr}(\mathbf{U}_t^\intercal \mathbf{B} \mathbf{U}_t), \tag{6.28}$$

which can be rewritten as

$$\sum_{i=1}^{n+M} \frac{\|(\mathbf{A}\mathbf{U}_{t+1})_{i\cdot}\|_2^2}{2\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2} + \mathrm{Tr}(\mathbf{U}_{t+1}^\intercal \mathbf{B} \mathbf{U}_{t+1}) \leq \sum_{i=1}^{n+M} \frac{\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2^2}{2\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2} + \mathrm{Tr}(\mathbf{U}_t^\intercal \mathbf{B} \mathbf{U}_t). \tag{6.29}$$

Thus the following inequality holds:

$$\begin{aligned} &\sum_{i=1}^{n+M} \left[\|(\mathbf{A}\mathbf{U}_{t+1})_{i\cdot}\|_2 - \left(\|(\mathbf{A}\mathbf{U}_{t+1})_{i\cdot}\|_2 - \frac{\|(\mathbf{A}\mathbf{U}_{t+1})_{i\cdot}\|_2^2}{2\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2} \right) \right] + \mathrm{Tr}(\mathbf{U}_{t+1}^\intercal \mathbf{B} \mathbf{U}_{t+1}) \\ \leq &\sum_{i=1}^{n+M} \left[\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2 - \left(\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2 - \frac{\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2^2}{2\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2} \right) \right] + \mathrm{Tr}(\mathbf{U}_t^\intercal \mathbf{B} \mathbf{U}_t). \end{aligned} \tag{6.30}$$

Note that the following inequality holds:

$$\|(\mathbf{A}\mathbf{U}_{t+1})_{i\cdot}\|_2 - \frac{\|(\mathbf{A}\mathbf{U}_{t+1})_{i\cdot}\|_2^2}{2\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2} \leq \|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2 - \frac{\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2^2}{2\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2}, \tag{6.31}$$

simply because for any real number a and b, the inequality $2ab \leq a^2 + b^2$ always holds. Thus according to (6.31), (6.30) is equivalent to

$$\sum_{i=1}^{n+M} \|(\mathbf{A}\mathbf{U}_{t+1})_{i\cdot}\|_2 + \mathrm{Tr}(\mathbf{U}_{t+1}^\intercal \mathbf{B}\mathbf{U}_{t+1}) \leq \sum_{i=1}^{n+M} \|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2 + \mathrm{Tr}(\mathbf{U}_t^\intercal \mathbf{B}\mathbf{U}_t), \quad (6.32)$$

which also equals to

$$\|\mathbf{A}\mathbf{U}_{t+1}\|_{2,1} + \mathrm{Tr}(\mathbf{U}_{t+1}^\intercal \mathbf{B}\mathbf{U}_{t+1}) \leq \|\mathbf{A}\mathbf{U}_t\|_{2,1} + \mathrm{Tr}(\mathbf{U}_t^\intercal \mathbf{B}\mathbf{U}_t). \quad (6.33)$$

By definitions in (6.19), we can conclude that the objective function in (6.17) does not increase by Algorithm 12. □

A More Efficient Algorithm

In Algorithm 12, we need to solve a large generalized eigenvalue problem ($\mathbf{E}, \mathbf{F} \in \mathbb{R}^{(n+M)\times(n+M)}$) in each iteration, which probably limits the application of READER for large-scale problems. Here we propose an efficient alternating algorithm to approximate Step 6 in Algorithm 12. We relax the optimization problem in (6.24) by simply removing the constraints so that the objective function becomes $\min_{\mathbf{U}} \mathcal{Q}(\mathbf{U})$, where $\mathcal{Q}(\mathbf{U}) = \mathrm{Tr}(\mathbf{U}^\intercal \mathbf{F}\mathbf{U})$. Since it is a convex function w.r.t. $\mathbf{U}$, we can recover the optimal $\mathbf{U}$ by setting the partial derivative $\frac{\partial \mathcal{Q}}{\partial \mathbf{U}} = 0$,

$$(\mathbf{F} + \mathbf{F}^\intercal)\mathbf{U} = \mathbf{0}. \quad (6.34)$$

Since $\mathbf{F} = \mathbf{A}^\intercal \mathbf{H}\mathbf{A} + \mathbf{B}$ and $\mathbf{F} = \mathbf{F}^\intercal$, we can rewrite (6.34) as

$$\left(\begin{bmatrix} \mathbf{X}^\intercal & \alpha \mathbf{I}_M \\ -\mathbf{I}_n & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{H}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_M \end{bmatrix} \begin{bmatrix} \mathbf{X} & -\mathbf{I}_n \\ \alpha \mathbf{I}_M & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \beta \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \gamma \mathbf{L}_y \end{bmatrix} \right) \begin{bmatrix} \mathbf{W} \\ \mathbf{V} \end{bmatrix} = \mathbf{0}. \quad (6.35)$$

$\mathbf{H} = \begin{bmatrix} \mathbf{H}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_M \end{bmatrix}$, where $\mathbf{H}_n \in \mathbb{R}^{n\times n}$ and $\mathbf{H}_M \in \mathbb{R}^{M\times M}$ are diagonal matrices. (6.35) actually equals to a set of linear equations. Hence solving (6.35) produces

$$\begin{aligned} \mathbf{W} &= \left[\mathbf{X}^\intercal \mathbf{H}_n \mathbf{X} + \alpha^2 \mathbf{H}_M + \beta \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} \right]^{-1} \mathbf{X}^\intercal \mathbf{H}_n \mathbf{V} \\ \mathbf{V} &= [\mathbf{H}_n + \gamma \mathbf{L}_y]^{-1} \mathbf{H}_n \mathbf{X}\mathbf{W}. \end{aligned} \quad (6.36)$$

Therefore in the tth iteration of Algorithm 12, we execute the following two steps instead of Steps 5 and 6:

$$\begin{aligned} \mathbf{W}_{t+1} &\leftarrow \left[\mathbf{X}^\intercal (\mathbf{H}_n)_t \mathbf{X} + \alpha^2 (\mathbf{H}_M)_t + \beta \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} \right]^{-1} \mathbf{X}^\intercal (\mathbf{H}_n)_t \mathbf{V}_t \\ \mathbf{V}_{t+1} &\leftarrow \left[(\mathbf{H}_n)_t + \gamma \mathbf{L}_y \right]^{-1} (\mathbf{H}_n)_t \mathbf{X}\mathbf{W}_{t+1}. \end{aligned} \quad (6.37)$$

Algorithm 13 depicts the efficient algorithm of READER. It is easy to prove that such an alternating optimization problem monotonically decreases the objective function in (6.17) in each iteration, and derives to the global optimum.

Algorithm 13 An efficient optimization algorithm for READER

Input: $\tilde{\mathbf{X}} \in \mathbb{R}^{N\times M}$, $\mathbf{X} \in \mathbb{R}^{n\times M}$, $\mathbf{Y} \in \{0,1\}^{n\times L}$

Output: Top ranked features

1: Calculate Laplacian matrices $\mathbf{L}_x$ and $\mathbf{L}_y$;
2: Set $t = 0$, and initialize $\mathbf{W}$ and diagonal matrices $\mathbf{H}_n$, $\mathbf{H}_M$;
3: **repeat**
4: $\quad \mathbf{V}^{t+1} \leftarrow \left[\mathbf{H}_n^t + \gamma\mathbf{L}_y\right]^{-1}\mathbf{H}_n^t\mathbf{X}\mathbf{W}^{t+1}$;
5: $\quad \mathbf{W}^{t+1} \leftarrow \left[\mathbf{X}^\intercal\mathbf{H}_n^t\mathbf{X} + \alpha\mathbf{H}_M^t + \beta\tilde{\mathbf{X}}^\intercal\mathbf{L}_x\tilde{\mathbf{X}}\right]^{-1}\mathbf{X}^\intercal\mathbf{H}_n^t\mathbf{V}^t$;
6: $\quad \forall i,\ (\mathbf{H}_n^{t+1})_{ii} \leftarrow \frac{1}{2\|(\mathbf{X}\mathbf{W}^{t+1}-\mathbf{Y})_{i\cdot}\|_2}$;
7: $\quad \forall i,\ (\mathbf{H}_M^{t+1})_{ii} \leftarrow \frac{1}{2\|(\mathbf{W}^{t+1})_{i\cdot}\|_2}$;
8: **until** *Convergence*
9: Return top ranked features by the descending order in $\|\mathbf{W}_{j\cdot}\|_2$, $\forall j$.

Because $\mathcal{Q}(\mathbf{U}) = \mathrm{Tr}(\mathbf{U}^\intercal\mathbf{F}\mathbf{U})$ is a convex function w.r.t. $\mathbf{U}$, $\mathbf{U} = \begin{bmatrix}\mathbf{W}\\ \mathbf{V}\end{bmatrix}$ recovered by $\frac{\partial\mathcal{Q}}{\partial\mathbf{U}} = 0$ in (6.46) would reduce the value of $\mathcal{Q}(\mathbf{U})$ in each iteration, i.e., $\mathcal{Q}(\mathbf{U}_{t+1}) \leq \mathcal{Q}(\mathbf{U}_t)$. Hence, according to the proof of Theorem 5, we reach the conclusion.

Time Complexity

In terms of time complexity, the optimization algorithm consists of two major stages: initialization and iteration. It is worth noting that $\mathbf{H}_n$, $\mathbf{H}_M$, $\mathbf{L}_x$ and $\mathbf{L}_y$ are sparse matrices, and typically we have $k \ll n, M$ and $n < N$. Therefore, in the initialization stage, calculation of $\mathbf{L}_x$, $\mathbf{L}_y$ and $\tilde{\mathbf{X}}^\intercal\mathbf{L}_x\tilde{\mathbf{X}}$ leads to a complexity of $\mathcal{O}(N^2 + M^2N)$. In iterations, updating $\mathbf{W}$ and $\mathbf{V}$ equals to solve a system of linear equations. Note that $(\mathbf{H}_n + \gamma\mathbf{L}_y)$ is also a sparse matrix, thus each iteration has a complexity of $\mathcal{O}(M^3+M^2n+Mn^2)$. In total, the time complexity of READER is $\mathcal{O}(N^2 + M^2N + t(M^3 + M^2n + n^2M))$ with t denoting the number of iterations. Due to the quadratic and cubic complexity in N and M, respectively, READER is effective for regular-scale datasets but not for large-scale ones.

6.4 An Improved Version of READER

In the previous section, we consider the loss function comes from two sources: **prediction loss** $loss_P(\cdot)$ and **embedding loss** $loss_E(\cdot)$. Specifically, by supposing that the features $\mathbf{X}$ and labels $\mathbf{Y}$ are embedded into corresponding subspace $\mathbf{X}\mathbf{W}$ and $\mathbf{V}$, respectively, the embedding loss $loss_E(\cdot)$ consists of two parts: $loss_{E_x}(\cdot)$ from the feature embedding and $loss_{E_y}(\cdot)$ from the label em-

bedding. Thus, the loss function in (6.17) can be rewritten as

$$loss(\tilde{\mathbf{X}}\mathbf{W}, \mathbf{Y}) = loss_P(\mathbf{X}\mathbf{W}, \mathbf{V}) + \beta loss_{E_x}(\tilde{\mathbf{X}}\mathbf{W}, \tilde{\mathbf{X}}) + \gamma loss_{E_y}(\mathbf{V}, \mathbf{Y}), \quad (6.38)$$

According to (6.38), we observe that the loss function in (6.17) is given based mainly on the n labeled training data, except the feature embedding loss function $loss_{E_x} = \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}}\mathbf{W})$. In order to fully utilize the unlabeled data, we rebuild the loss function in the following,

$$loss(\tilde{\mathbf{X}}\mathbf{W}, \mathbf{Y}) = loss_P(\tilde{\mathbf{X}}\mathbf{W}, \tilde{\mathbf{V}}) + \beta loss_{E_x}(\tilde{\mathbf{X}}\mathbf{W}, \tilde{\mathbf{X}}) + \gamma loss_{E_y}(\tilde{\mathbf{V}}, \mathbf{Y}), \quad (6.39)$$

where $\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \\ \mathbf{X}_u \end{bmatrix} \in \mathbb{R}^{N\times M}$ and $\tilde{\mathbf{V}} = \begin{bmatrix} \mathbf{V} \\ \mathbf{V}_u \end{bmatrix} \in \mathbb{R}^{N\times k}$. Note that $\mathbf{X}_u \in \mathbf{R}^{(N-n)\times M}$ and $\mathbf{V}_u \in \mathbf{R}^{(N-n)\times k}$ denotes unlabeled data and the corresponding embedded labels, respectively.

First, we rewrite the *prediction loss* $\boldsymbol{loss}_{\mathbf{P}}$. To take all instances, including both labeled and unlabeld instances into account for empirical risk minimization, we re-define the prediction loss as

$$\begin{aligned} loss_P(\tilde{\mathbf{X}}\mathbf{W}, \tilde{\mathbf{V}}) &:= \sum_{i=1}^{n} c_a \left\| \mathbf{W}^\intercal \mathbf{x}_i - \tilde{\mathbf{V}}_{i\cdot} \right\|_2 + \sum_{i=n+1}^{N} c_b \left\| \mathbf{W}^\intercal \mathbf{x}_i - \tilde{\mathbf{V}}_{i\cdot} \right\|_2, \\ &= \left\| \mathbf{C}\left(\tilde{\mathbf{X}}\mathbf{W} - \tilde{\mathbf{V}}\right) \right\|_{2,1}, \end{aligned} \quad (6.40)$$

where $\mathbf{C} = \begin{bmatrix} c_a\mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & c_b\mathbf{I}_{N-n} \end{bmatrix} \in \mathbb{R}^{N\times N}$ is a diagonal score matrix, with ith diagonal element $\mathbf{C}_{ii} = c_i$ indicating the importance of the ith training instance, $\forall i$. Here we simply assume that the labeled training instances should contribute more to the loss function than the unlabeled ones, i.e., $c_a > c_b \geq 0$.

Second, we focus on rebuilding the *feature embedding loss* $\boldsymbol{loss}_{\mathbf{E_x}}$. Same with (6.17), it is easy to build it in the setting of semi-supervised learning,

$$\begin{aligned} loss_{E_x}(\tilde{\mathbf{X}}\mathbf{W}, \tilde{\mathbf{X}}) &:= \sum_{i,j=1}^{N} (\mathbf{S}_X)_{ij} \left\| \mathbf{W}^\intercal \mathbf{x}_i - \mathbf{W}^\intercal \mathbf{x}_j \right\|_2^2, \\ &= \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal (\mathbf{D}_X - \mathbf{S}_X)\tilde{\mathbf{X}}\mathbf{W}), \\ &= \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{L}_X \tilde{\mathbf{X}}\mathbf{W}). \end{aligned} \quad (6.41)$$

For convenience, we denote the Laplacian matrix $\mathbf{L}_x$ on instances in (6.17) by $\mathbf{L}_X = \begin{bmatrix} \mathbf{L}_X^{(ll)} & \mathbf{L}_X^{(lu)} \\ \mathbf{L}_X^{(ul)} & \mathbf{L}_X^{(uu)} \end{bmatrix}$, with $\mathbf{L}_X^{(ll)} \in \mathbb{R}^{n\times n}$ and $\mathbf{L}_X^{(uu)} \in \mathbb{R}^{(N-n)\times(N-n)}$, which are the Laplacian matrices on the labeled and unlabeled data, respectively.

Third, we rewrite the *label embedding loss* $\boldsymbol{loss}_{\mathbf{E_y}}$ based on (6.40) and (6.41). To this end, we induce the new label embedding loss $loss_{E_y}$ by extending (6.15)

as

$$
\begin{aligned}
loss_{E_y}(\tilde{\mathbf{V}}, \mathbf{Y}) \coloneqq & c_a \sum_{i,j=1}^{n} (\mathbf{S}_y)_{ij} \left\| \tilde{\mathbf{V}}_{i\cdot} - \tilde{\mathbf{V}}_{j\cdot} \right\|_2^2 + c_b \sum_{\substack{i=1:n \\ j=(n+1):N}} (\mathbf{S}_X)_{ij} \left\| \tilde{\mathbf{V}}_{i\cdot} - \tilde{\mathbf{V}}_{j\cdot} \right\|_2^2 + \\
& c_b \sum_{\substack{i=(n+1):N \\ j=1:n}} (\mathbf{S}_X)_{ij} \left\| \tilde{\mathbf{V}}_{i\cdot} - \tilde{\mathbf{V}}_{j\cdot} \right\|_2^2 + c_b \sum_{i,j=n+1}^{N} (\mathbf{S}_X)_{ij} \left\| \tilde{\mathbf{V}}_{i\cdot} - \tilde{\mathbf{V}}_{j\cdot} \right\|_2^2 \\
= & \mathrm{Tr}(\mathbf{V}^\intercal c_a \mathbf{L}_y \mathbf{V}) + \mathrm{Tr}(\mathbf{V}^\intercal c_b \mathbf{L}_X^{(lu)} \mathbf{V}_u) + \\
& \mathrm{Tr}(\mathbf{V}_u^\intercal c_b \mathbf{L}_X^{(ul)} \mathbf{V}) + \mathrm{Tr}(\mathbf{V}_u^\intercal c_b \mathbf{L}_X^{(uu)} \mathbf{V}_u) \\
= & \mathrm{Tr}\left(\begin{bmatrix} \mathbf{V} \\ \mathbf{V}_u \end{bmatrix}^\intercal \begin{bmatrix} c_a \mathbf{L}_y & c_b \mathbf{L}_X^{(lu)} \\ c_b \mathbf{L}_X^{(ul)} & c_b \mathbf{L}_X^{(uu)} \end{bmatrix} \begin{bmatrix} \mathbf{V} \\ \mathbf{V}_u \end{bmatrix} \right) \\
= & \min_{\tilde{\mathbf{V}}} \mathrm{Tr}(\tilde{\mathbf{V}}^\intercal \mathbf{L}_Y \tilde{\mathbf{V}}).
\end{aligned} \tag{6.42}
$$

Note that, $\mathbf{L}_Y \in \mathbb{R}^{N \times N}$ in (6.42) is different with $\mathbf{L}_y \in \mathbb{R}^{n \times n}$ in (6.17). In other words, we can embedding the whole label matrix $\tilde{\mathbf{Y}} = \begin{bmatrix} \mathbf{Y} \\ \mathbf{Y}_u \end{bmatrix}$, including both labeled data $\mathbf{Y}$ and unlabeld data $\mathbf{Y}_u$, into a low-dimensional latent label space $\tilde{\mathbf{V}}$. In fact, (6.42) is built based on two reasonable assumptions we made on $loss_{E_y}$: first, the instance similarity $(\mathbf{S}_X)_{ij}$ is used to model the similarity between two unknow label vectors $\mathbf{y}_i$ and $\mathbf{y}_j$ from $\mathbf{Y}_u$; Second, the embedding loss from $\mathbf{Y}$ is more important than the loss from $\mathbf{Y}_u$, i.e., $c_a > c_b$. Similar with $\mathbf{L}_Y$, we infer the degree matrix on labels from $\mathbf{D}_X$ by setting $\mathbf{D}_Y = \begin{bmatrix} c_a \mathbf{D}_y & c_b \mathbf{D}_X^{(lu)} \\ c_b \mathbf{D}_X^{(ul)} & c_b \mathbf{D}_X^{(uu)} \end{bmatrix}$.

Finally, according to the loss function rewritten in (6.40), (6.41) and (6.42), we obtain the objective function of the improved READER for semi-supervised learning,

$$
\begin{aligned}
& \min_{\mathbf{W},\tilde{\mathbf{V}}} \left\| \mathbf{C} \left(\tilde{\mathbf{X}} \mathbf{W} - \tilde{\mathbf{V}} \right) \right\|_{2,1} + \alpha \left\| \mathbf{W} \right\|_{2,1} + \beta \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{L}_X \tilde{\mathbf{X}} \mathbf{W}) + \gamma \mathrm{Tr}(\tilde{\mathbf{V}}^\intercal \mathbf{L}_Y \tilde{\mathbf{V}}), \\
& \text{s.t. } \mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{D}_X \tilde{\mathbf{X}} \mathbf{W} = \tilde{\mathbf{V}}^\intercal \mathbf{D}_Y \tilde{\mathbf{V}} = \mathbf{I},
\end{aligned} \tag{6.43}
$$

According to (6.43), we can update following definitions for Theorem 4.

$$
\begin{aligned}
\mathbf{A} \coloneqq & \begin{bmatrix} \mathbf{C}\tilde{\mathbf{X}} & -\mathbf{C} \\ \alpha \mathbf{I}_M & \mathbf{0} \end{bmatrix}, \ \mathbf{B} \coloneqq \begin{bmatrix} \beta \tilde{\mathbf{X}}^\intercal \mathbf{L}_X \tilde{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \gamma \mathbf{L}_Y \end{bmatrix}, \\
\mathbf{E} \coloneqq & \begin{bmatrix} \tilde{\mathbf{X}}^\intercal \mathbf{D}_X \tilde{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_Y \end{bmatrix}, \ \mathbf{U} \coloneqq \begin{bmatrix} \mathbf{W} \\ \tilde{\mathbf{V}} \end{bmatrix},
\end{aligned} \tag{6.44}
$$

Thus Algorithm 12 can be used to solve the optimization problem in (6.43) according to the definitions in (6.44). In addition, we can also derive an more efficient optimization algorithm in a similar way with READER. By setting the

partial derivative $\frac{\partial \mathcal{Q}}{\partial \widetilde{\mathbf{U}}} = \mathbf{0}$, we have

$$\begin{cases} \frac{\partial \mathcal{Q}}{\partial \widetilde{\mathbf{W}}} &= \left[\tilde{\mathbf{X}}^\intercal \mathbf{C}^\intercal \mathbf{H}_N \mathbf{C}\tilde{\mathbf{X}} + \alpha^2 \mathbf{H}_M + \beta \tilde{\mathbf{X}}^\intercal \mathbf{L}_X \tilde{\mathbf{X}}\right] \mathbf{W} - \tilde{\mathbf{X}}^\intercal \mathbf{C}^\intercal \mathbf{H}_N \mathbf{C}\tilde{\mathbf{V}} = \mathbf{0} \\ \frac{\partial \mathcal{Q}}{\partial \widetilde{\mathbf{V}}} &= \left[\mathbf{C}^\intercal \mathbf{H}_N \mathbf{C} + \gamma \mathbf{L}_Y\right] \tilde{\mathbf{V}} - \mathbf{C}^\intercal \mathbf{H}_N \mathbf{C}\tilde{\mathbf{X}}\mathbf{W} = \mathbf{0}. \end{cases} \tag{6.45}$$

Therefore in the tth iteration of Algorithm 12, we execute the following two steps instead of Step 5 and 6:

$$\begin{aligned} \mathbf{V}_{t+1} &\leftarrow \left[\mathbf{C}^\intercal (\mathbf{H}_N)_t \mathbf{C} + \gamma \mathbf{L}_Y\right]^{-1} \mathbf{C}^\intercal (\mathbf{H}_N)_t \mathbf{C}\tilde{\mathbf{X}}\mathbf{W}_{t+1}, \\ \mathbf{W}_{t+1} &\leftarrow \left[\tilde{\mathbf{X}}^\intercal \mathbf{C}^\intercal (\mathbf{H}_N)_t \mathbf{C}\tilde{\mathbf{X}} + \alpha^2 (\mathbf{H}_M)_t + \beta \tilde{\mathbf{X}}^\intercal \mathbf{L}_X \tilde{\mathbf{X}}\right]^{-1} \tilde{\mathbf{X}}^\intercal \mathbf{C}^\intercal (\mathbf{H}_N)_t \mathbf{C}\mathbf{V}_t. \end{aligned} \tag{6.46}$$

6.5 Experimental Results

6.5.1 Experimental Results on MLSF

The proposed **MLSF** was compared with four popular MLC methods:

- **BR** [8]: a baseline method. A multi-label problem is transformed to L single-label problems.
- **MDDM** [113]: a global FS-DR method. The project is built by maximizing the feature-label dependence.
- **LIFT** [108]: a local FS-DR method. Label-specific features are selected by cluster analysis.
- **LLSF** [38]: a local FS-DR method. Label-specific features are selected by preserving label correlations.

BR was introduced as a baseline MLC method with linear time complexity in the problem size. MDDM was chosen as a representative of global FS-DR methods, which outperformed several global FS-DR methods, like PCA, LPP [34], MLSI [105], as reported in [113]. As local FS-DR methods, LIFT and LLSF were chosen. They have been shown their performance advantages in comparison with several state-of-the-art MLC methods in [108, 38].

In the experiments, *5-fold cross validation* was performed to evaluate the classification performance. For fair comparison, BR with a linear SVM of LIBSVM [11] is used as the multi-label classifier for MDDM, LIFT, LLSF and MLSF. In parameter setting, we set the parameters of LIFT and LLSF as suggested by the authors, and set the dimensionality of the feature subspace to 30 for its optimal average performance. For MLSF, to balance the classification accuracy and processing time, we set the five parameters K, ϵ, α, γ and ρ as

Table 6.1: Experimental results (mean±std.) on twelve multi-label datasets in four evaluation metrics.

Method	Emotions	Scene	Yeast	Genbase	Medical	Enron	Rcv1s1	Rcv1s2	Mediamill	Bibtex	Corel16k1	Corel16k2
Exact-Match												
BR	.285±.015	.533±.013	.148±.009	**.982±.005**	.665±.008	.111±.009	.071±.004	.172±.003	.066±.004	.143±.004	.006±.001	.004±.001
MDDM	.263±.013	.529±.007	.137±.009	.980±.005	.609±.014	.121±.006	.065±.002	.174±.006	.068±.004	.143±.003	.000±.000	.001±.000
LIFT	.184±.014	**.637±.010**	.154±.008	.953±.012	.574±.010	.119±.004	.059±.003	.145±.003	.069±.004	.139±.003	.000±.000	.000±.000
LLSF	.285±.015	.531±.014	.148±.009	**.982±.005**	.662±.008	.111±.009	.072±.004	.172±.003	.066±.004	**.144±.004**	.006±.001	.004±.000
MLSF	**.315±.016**	**.637±.007**	**.212±.012**	**.982±.005**	**.689±.009**	**.122±.009**	**.128±.004**	**.214±.006**	**.070±.003**	.143±.002	**.008±.001**	**.007±.003**
Rank	MLSF ≻ LLSF ≻ BR ≻ MDDM ≻ LIFT											
Hamming-Score												
BR	**.805±.007**	.895±.003	.801±.004	**.999±.000**	**.990±.000**	.940±.001	.965±.000	.969±.000	.968±.000	.984±.000	.980±.000	.981±.000
MDDM	.788±.004	.899±.001	.798±.004	**.999±.000**	.988±.000	.953±.001	.973±.000	.974±.000	**.969±.000**	**.988±.000**	**.981±.000**	**.983±.000**
LIFT	.755±.005	**.919±.002**	**.804±.003**	.998±.000	.987±.000	**.955±.000**	**.974±.000**	**.977±.000**	**.969±.000**	**.988±.000**	**.981±.000**	**.983±.000**
LLSF	**.805±.007**	.895±.003	.801±.004	**.999±.000**	**.990±.000**	.940±.001	.965±.000	.969±.000	.968±.000	.984±.000	.980±.000	.981±.000
MLSF	.793±.016	.891±.002	.789±.004	**.999±.000**	**.990±.000**	.940±.001	.966±.000	.970±.000	.968±.000	.984±.000	.980±.000	.981±.003
Rank	LIFT ≻ MDDM ≻ BR ≿ LLSF ≻ MLSF											
Macro-F1												
BR	.633±.013	.694±.008	.322±.005	.761±.020	.366±.011	**.222±.005**	.250±.008	.235±.004	.028±.000	.328±.003	**.047±.001**	**.051±.004**
MDDM	.583±.017	.684±.005	.318±.005	.754±.017	.323±.008	.201±.009	.156±.005	.217±.005	**.035±.001**	.159±.001	.008±.001	.012±.001
LIFT	.496±.010	**.759±.005**	.319±.005	.704±.020	.240±.009	.136±.005	.134±.005	.096±.002	**.035±.000**	.145±.001	.003±.001	.004±.000
LLSF	.633±.013	.693±.009	.322±.005	**.769±.016**	.370±.016	**.222±.005**	.246±.007	.236±.004	.028±.000	**.329±.001**	.045±.001	.049±.003
MLSF	**.657±.016**	.699±.007	**.346±.009**	**.769±.016**	**.387±.012**	.221±.005	**.255±.007**	**.240±.005**	.029±.003	.328±.002	**.047±.002**	**.051±.003**
Rank	MLSF ≻ BR ≿ LLSF ≻ MDDM ≻ LIFT											
Micro-F1												
BR	.661±.014	.688±.009	.631±.008	**.993±.002**	.810±.004	.515±.005	.399±.004	.413±.005	.510±.002	.422±.002	**.072±.002**	**.079±.003**
MDDM	.627±.010	.682±.005	.627±.009	.992±.002	.780±.007	**.579±.006**	.356±.005	.395±.003	**.528±.002**	.364±.004	.007±.001	.016±.001
LIFT	.557±.011	**.755±.007**	.632±.007	.980±.005	.679±.005	.570±.003	.345±.004	.327±.007	.519±.002	.338±.002	.005±.001	.012±.001
LLSF	.661±.014	.686±.010	.631±.008	**.993±.002**	.804±.005	.515±.005	.396±.003	.412±.005	.510±.002	**.423±.001**	.069±.002	**.079±.002**
MLSF	**.665±.016**	.692±.005	**.639±.008**	**.993±.002**	**.815±.004**	.515±.004	**.407±.004**	**.419±.004**	.491±.011	**.423±.001**	.070±.002	.076±.003
Rank	MLSF ≻ BR ≻ LLSF ≻ MDDM ≻ LIFT											

$\lceil L/10 \rceil$, 0.01, 0.8, 0.01 and 1, respectively. In addition, to make MDDM executable for all datasets, we randomly sampled 8000 instances for Mediamill, Corel16k1 and Corel16k2. We implemented the MATLAB codes of BR[1] and MLSF[1], and obtained the MATLAB codes of MDDM, LIFT and LLSF from the authors. Experiments were performed in a computer configured with a Intel Quad-Core i7-4770 CPU at 3.4GHz with 4GB RAM.

Next we compared the classification accuracies of five MLC methods. The experimental results are shown in Table 8.1 where the averaged performance order over all datasets is shown in the last row Rank. MLSF outperformed the other methods on the average in three metrics of Exact-Match, Macro/Micro-F1. It demonstrates the effectiveness of MLSF and verifies the existence of meta-labels with specific features. Nevertheless MLSF worked the worst in term of Hamming-Score. It is probably because MLSF tends to optimize Exact-Match by learning meta-labels, which would harm the performance in Hamming-Score [18]. LIFT ranked at the first in Hamming-Score, and was even better than the Hamming-Score optimizer, BR, showing the success of local FS-DR strategy. However, LIFT worked the worst in other three metrics. The unsatisfactory

[1]We provide the MATLAB codes at: https://github.com/futuresun912/MLSF.git

performance of LIFT possibly shows that the extracting way of label-specific features is not enough for handling MLC problems. By taking label correlations into account for local FS-DR, LLSF ranked at the second in Exact-Match and worked better than LIFT, except in Hamming-Score, showing the importance of modeling label correlations. As the global FS-DR method, MDDM performed worse than the baseline BR, except in Hamming-Score. It seems that projecting features into the identical subspace possibly weakens the discriminative ability for some labels. On the other hand, as the simplest MLC method, BR provided competitive classification performance in comparison with FS-DR methods, especially in large-scale datasets. It is probably because large-scale datasets typically need a sufficient number of instances for training, while FS-DR tends to remove too many features.

6.5.2 Experimental Results on READER

To evaluate the performance of the proposed READER, we conducted experiments on benchmark multi-label datasets in Mulan [95]. The statistics of experimental datasets are summarized in Table 2.1. We compared the performance of **READER**[2] with the following feature selection methods:

- **F-Score**: A classical filter-based method, which evaluates features one by one across all labels according to Fisher Score [23], and returns the top ranked most discriminative features;

- **RFS**: Robust Feature Selection via joint $\ell_{2,1}$-norm minimization [61]. RFS applies the $\ell_{2,1}$-norm on both the loss function and the regularization term, thereby be robust to outliers and select features across labels. The optimization problem of RFS is given as;

$$\min_{\mathbf{W}} \|\mathbf{X}\mathbf{W} - \mathbf{Y}\|_{2,1} + \alpha \|\mathbf{W}\|_{2,1}; \tag{6.47}$$

- **CSFS**: Convex Semi-supervised multi-label Feature Selection [12]. It utilizes both labeled and unlabeled data to select features while saving correlations among different features;

$$\begin{aligned} &\min_{\mathbf{W}} \sum_{i=1}^{N} c_i \left\| \mathbf{W}^{\intercal}\mathbf{x}_i - \tilde{\mathbf{Y}}_{i\cdot} \right\|_2^2 + \alpha \|\mathbf{W}\|_{2,1}, \\ &\text{s.t. } 0 \leq \tilde{\mathbf{Y}}_{ij} \leq 1, \ \forall i, j; \end{aligned} \tag{6.48}$$

- **MIFS**: Multi-label Informed Feature Selection [39]. It makes use of the latent label space to guide the feature selection phase, and exploits label

[2]We provide the MATLAB codes of READER at: https://github.com/futuresun912/READER.git

correlations to find features across multiple labels.

$$\min_{\mathbf{W},\mathbf{V},\mathbf{P}} \|\mathbf{X}\mathbf{W}-\mathbf{V}\|_F^2+\alpha\|\mathbf{W}\|_{2,1}+\beta\|\mathbf{Y}-\mathbf{V}\mathbf{P}^\intercal\|_F^2+\gamma\mathrm{Tr}(\mathbf{V}^\intercal\mathbf{L}_x\mathbf{V}). \quad (6.49)$$

- **SCCA**: Semi-supervised CCA (SCCA) [41]. This algorihtm utilizes feature extraction instead of feature selection. It is a trade-off combination of eigenvalue problems of supervised (Canonical Correlation Analysis) CCA and unsupervised Principal Component Analysis (PCA):

$$\mathbf{B}\begin{bmatrix}\mathbf{w}_x\\ \mathbf{w}_y\end{bmatrix}=\lambda\mathbf{C}\begin{bmatrix}\mathbf{w}_x\\ \mathbf{w}_y\end{bmatrix}, \quad (6.50)$$

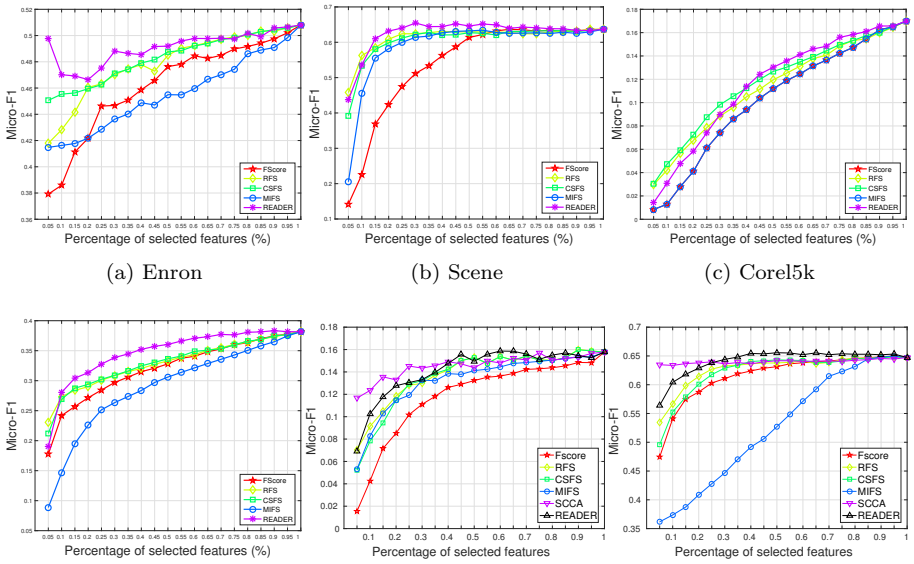
where

$$\begin{aligned}\mathbf{B}&=\beta\begin{bmatrix}\mathbf{0} & \mathbf{X}^\intercal\mathbf{Y}\\ \mathbf{Y}^\intercal\mathbf{X} & \mathbf{0}\end{bmatrix}+(1-\beta)\begin{bmatrix}\tilde{\mathbf{X}}^\intercal\tilde{\mathbf{X}} & \mathbf{0}\\ \mathbf{0} & \mathbf{Y}^\intercal\mathbf{Y}\end{bmatrix}\\ \mathbf{C}&=\beta\begin{bmatrix}\mathbf{X}^\intercal\mathbf{X} & \mathbf{0}\\ \mathbf{0} & \mathbf{Y}^\intercal\mathbf{Y}\end{bmatrix}+(1-\beta)\begin{bmatrix}\mathbf{I}_m & \mathbf{0}\\ \mathbf{0} & \mathbf{I}_\ell\end{bmatrix}.\end{aligned} \quad (6.51)$$

F-Score is selected as a representative of filter-based methods. RFS is introduced due to its simple implementation on $\ell_{2,1}$-norm-based sparsity regularization, and its performance superiority over several classical methods. As a semi-supervised method, CSFS outperforms several state-of-the-art methods, such as SFUS [56], SFSS [57] and LSDR [114], over popular multi-label datasets. MIFS is chosen because it can capture label correlations as READER does. Although feature extraction is not our main concern, but SCCA is included as a representative of semi-supervised CCA methods for comparison.

In parameter setting, to model the local consistency of $\mathbf{X}$ in READER and MIFS, we set the parameters p and σ as 5 and 1, respectively. The regularization parameters (α, β, γ in READER and MIFS, μ in CSFS and γ in RFS) are tuned in the range of $\{0.001, 0.01, 0.1, 1, 10, 100\}$ by grid search. In addition, the dimensionality k of latent label space is chosen from $\{10\%, 30\%, 50\%, 70\%, 90\%\}$ of total number l of labels. The parameter s on the importance of unlabeled data in CSFS is tuned in the range of $\{0.001, 0.01, 0.1\}$. In SCCA, the parameter β is selected from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. For each iterative optimization algorithm, we terminate it once the relative change of its objective is below 10^{-5}. For fair comparison, Binary Relevance [8] with linear SVM implementation in Liblinear [25] is used as the baseline multi-label classifier for all the comparing feature selection methods. For all the methods, we report the best results in terms of Macro/Micro-F1 by 5-fold cross validation. All the comparing methods are implemented in Matlab, and experiments are performed in a computer configured with an Intel Quad-Core i7-4770 CPU at 3.4GHz with 4GB RAM.

In the first experiment, we randomly sample 30% training instances as the labeled instances ($n/N = 30\%$), and vary the number of selected features from 5%



(a) Enron (b) Scene (c) Corel5k

(d) Rcv1s1 (e) Mirflickr (f) Tmc2007

Figure 6.2: Comparison of six feature selection/extraction algorithms in Micro-F1 on six datasets by varying m/M from 5% to 100% by step 5% at $n/N = 30\%$.

to 100% by step 5% of the total number of features ($m/M = 5\%, 10\%, ..., 100\%$). Fig. 6.2 compares five feature selection methods and one feature extraction method in Micro-F1 on six multi-label datasets. We observe that as the number of selected features inceases, the performances of feature selection methods increase first and then converge. READER outperforms the other four feature selection algorithms in most cases, except on the cases of Corel5k with a small number of selected features. This advantage comes probably from its ability on capturing label correlations and handling semi-supervised learning. CSFS is the second and is better than the three supervised methods, FScore, RFS and MIFS, indicating the importance of utilizing unlabeled data on selecting discriminative features. As a feature extraction algorithm, SCCA gains a significant performance advantage against the feature selection algorithms when a smaller percentage of features are extracted. However, such advantage becomes smaller or even disappears as the number of extracted features increases. This is because an extracted feature includes the information from all the original features, even noisy features, so that it can find a small number of good combinations of original features, but it becomes harder to find many such good combinations. Rather, feature selection is useful for finding many informative features.

In the second experiment, we vary the percentage n/N in $\{10\%, 30\%, 50\%\}$ at $m/M = 60\%$. The results in Macro/Micro-F1 are reported in Table 6.2, where

Table 6.2: Experimental results on six multi-label datasets in Macro-F1 and Micro-F1 by varying the percentage n/N of labeled data from $\{10\%, 30\%, 50\%\}$ at $m/M = 60\%$.

		Macro-F1						**Micro-F1**					
Dataset	n/N	FScore	RFS	CSFS	MIFS	SCCA	READER	FScore	RFS	CSFS	MIFS	SCCA	READER
Enron	10%	0.129	0.129	0.124	0.119	0.132	**0.134**	0.475	0.481	0.480	0.484	0.462	**0.494**
	30%	0.149	0.154	0.156	0.147	0.154	**0.167**	0.483	0.489	0.490	0.474	0.452	**0.508**
	50%	0.164	0.180	0.181	0.161	0.179	**0.186**	0.478	0.509	0.509	0.476	0.468	**0.511**
Scene	10%	0.606	0.602	0.603	0.626	0.614	**0.630**	0.599	0.592	0.594	0.620	0.608	**0.625**
	30%	0.652	0.653	0.651	0.659	0.641	**0.665**	0.630	0.645	0.644	0.653	0.633	**0.657**
	50%	0.660	0.664	0.665	**0.676**	0.671	0.669	0.653	0.658	0.657	**0.669**	0.664	0.663
Corel5k	10%	0.019	0.020	**0.021**	**0.021**	**0.021**	**0.021**	0.120	0.129	0.131	0.128	**0.142**	0.135
	30%	0.028	0.032	0.030	0.032	**0.034**	0.032	0.133	0.143	0.144	0.144	**0.163**	0.145
	50%	0.030	0.032	0.033	0.034	**0.040**	0.036	0.123	0.143	0.144	0.139	**0.169**	0.144
Rcv1s1	10%	0.129	0.130	0.127	0.134	**0.155**	0.137	0.334	0.333	0.325	0.327	**0.338**	**0.338**
	30%	0.191	0.192	**0.193**	0.191	**0.193**	**0.193**	0.368	0.369	0.370	0.349	0.348	**0.379**
	50%	0.223	0.219	**0.226**	0.208	0.206	0.219	0.378	0.380	0.380	0.354	0.358	**0.381**
Mirflickr	10%	0.058	0.065	0.065	0.056	0.063	**0.067**	0.163	0.180	0.179	0.155	0.179	**0.182**
	30%	0.050	0.054	**0.057**	0.052	0.055	0.055	0.138	0.154	0.156	0.146	0.151	**0.160**
	50%	0.048	0.053	0.053	0.050	**0.054**	0.053	0.134	0.147	0.145	0.139	**0.150**	0.145
Tmc2007	10%	0.464	0.466	0.453	0.443	0.458	**0.468**	0.602	0.604	0.599	0.600	0.590	**0.607**
	30%	0.509	0.514	0.520	0.417	0.522	**0.524**	0.645	0.647	0.646	0.575	0.645	**0.651**
	50%	0.530	0.531	**0.540**	0.436	0.538	0.531	0.666	0.670	**0.674**	0.595	0.670	0.671

the best performance is highlighted in boldface. In Table 6.2, as the percentage of labeled data increases, the performance increases either except for some cases. READER shows the best performance in over half cases. It is noteworthy that READER works best at $n/N = 10\%$ on two large-scale datasets, indicating its success on using the large number of unlabeled data, which is important for a semi-supervised algorithm.

To demonstrate the effectiveness of using manifold learning terms (6.12) and (6.15) in READER, we further conduct the third experiment by varying n/N in range of $\{10\%, 20\%, 30\%, 40\%, 50\%\}$ at $m/M = 30\%$, while adding the value 0 to the search space of parameters β and γ. Note that β measures the contribution of unlabeled training instances by manifold learning, thus $\beta = 0$ indicates the ignorance of unlabeled training data in (6.17). In addition, γ controls the importance of label embedding, therefore, $\mathbf{V}$ is replaced by $\mathbf{Y}$ and no label embedding is conducted if $\gamma = 0$. Figure 6.3 shows the experimental results in Macro/Micro-F1 on two datasets. As shown in Fig. 6.3, using the whole training data ($\beta \neq 0$), rather than labeled data only ($\beta = 0$), the classification performance is increased on all the experimental datasets. As n/N increases, the degree of such an improvement first increases, and then degrades. In addition, saving label correlations in feature selection is also important for the classification performance. From Fig. 6.3, we can observe that it is usually better to project labels into the low-dimensional latent space ($\gamma \neq 0$), rather

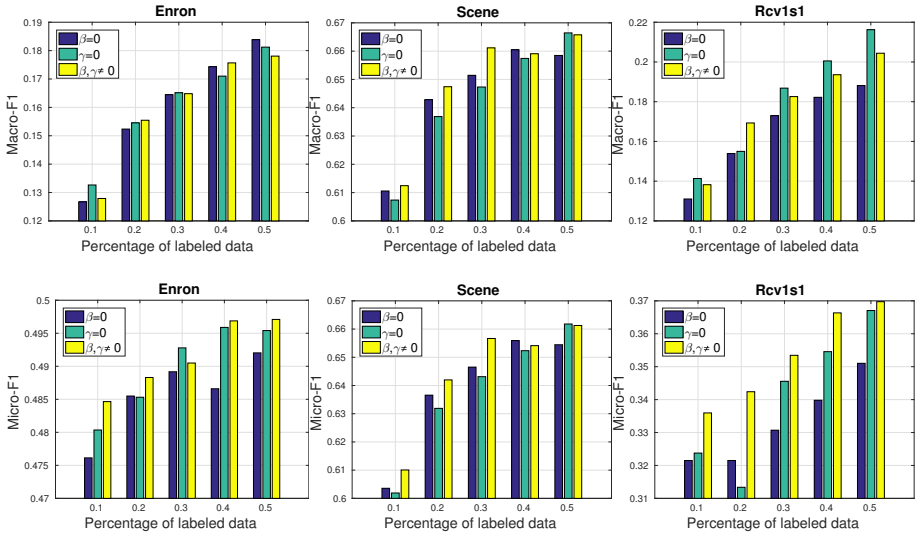


Figure 6.3: The performances of three variants of READER on three datasets by varying n/N in $\{10\%, 20\%, 30\%, 40\%, 50\%\}$ at $m/M = 30\%$.

than use the original label information ($\gamma = 0$) for feature selection.

To show the robustness of READER benefited from the $\ell_{2,1}$-norm loss function, we perform the fourth experiment by comparing the performances between READER and READER-F, where READER-F employs the square loss function in (6.17). Figure 6.4 shows the experimental results on six datasets in Macro/Micro-F1 by fixing the percentage of labeled data and selected features at 30%. Indeed, using the $\ell_{2,1}$-norm improves READER's robustness.

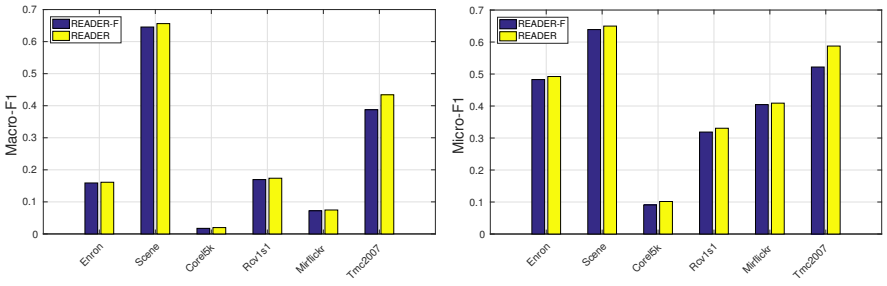


Figure 6.4: The analysis on the robustness of READER at $n/N = m/M = 30\%$. Here READER-F employs the square loss function in (6.17).

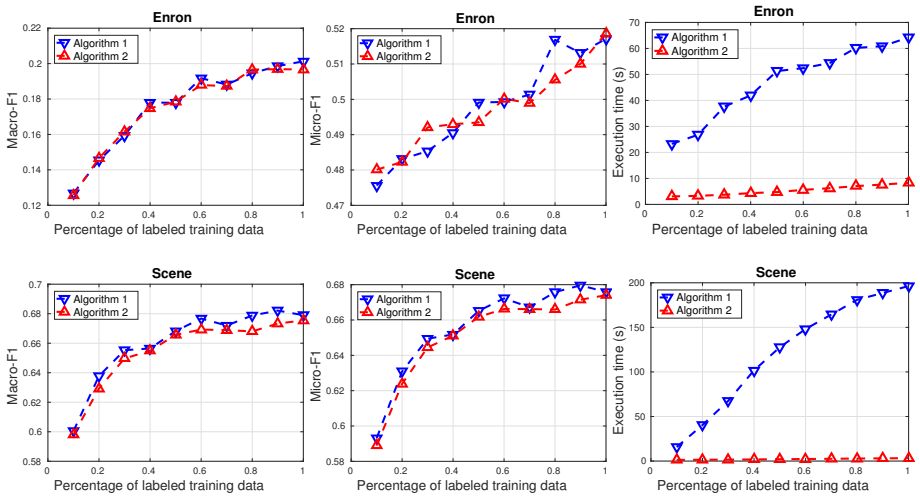


Figure 6.5: Comparing two optimization algorithms of READER on Enron and Scene. The percentage n/N is varied from 10% to 100% by step 10% at $m/M = 30\%$.

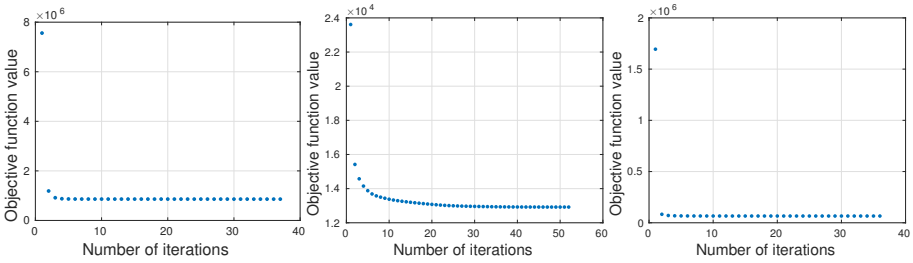
6.5.3 Optimization Algorithm Analysis

In Sec. 4.1, we developed an efficient optimization algorithm by relaxing the constraints in (6.17). To show its efficiency of the optimization algorithm developed in Sec. 4.1, we compare its performance with Algorithm 1, and term the efficient algorithm as Algorithm 2. In this experiment, n/N is varied from 10% to 100% by step 10%, and top 30% features are feed to the multi-label classifiers. The same strategy on parameter tuning and result reporting introduced in Sec. 5.1 is used here. Figure 6.5 shows the experimental results on the Scene and Enron datasets. As shown in Fig. 6.5, at the expense of slight degradation of performance, Algorithm 2 obtains a large amount of efficiency in execution time.

Next we demonstrate that Algorithm 2 converges at the global optimum. In this experiment, we used all the instances in each dataset as the training set in $n/N = 50\%$. The parameters of READER are fixed as $\alpha = \beta = \gamma = 1, k = 0.3$. Figure 6.6 shows the convergence curves of the objective function value in (6.17) by Algorithm 2 in Section 4.1. From Figure 6.6, we can observe that the objective function value converges after a few number of iterations, demonstrating the efficiency of the proposed algorithm.

Parameter Sensitivity Analysis

To evaluate the potential of READER, parameter sensitivity analysis is conducted on the Enron dataset in terms of three important regularization param-



(a) Enron (b) Scene (c) Rcv1s1

Figure 6.6: Convergence analysis of the optimization algorithm in Sec.4.1. The algorithm converged at the 37th, 52nd and 36th iteration on Enron, Scene and Rcv1s1, respectively.

eters α, β and γ. Specifically, α controls the sparsity of the proposed model, β measures the importance of unlabeled instances and γ controls the strongness on preserving useful label information by low-dimensional latent space. We select the value of the three parameters from $\{0.001, 0.01, 0.1, 0.5, 1, 5, 10\}$. Figure 6.3 shows the experimental results in Macro/Micro-F1 by varying the percentage m/M of selected features from 10% to 100% by step 10%. Specifically, figures on α are shown by fixing $\beta = \gamma = 1$, and the similar setting is used for the figures on β and γ. We can achieve the following observations according to Figure 8.6.

- The classification performance is sensitive to the changes of values of α, β and γ, only if the percentage of selected features is less than 30%;
- Compared with α, β and γ, the performance is more sensitive to the percentage of selected features;
- READER achieves its best performance on Enron by setting parameters $\alpha = 1$, $\beta = 0.1$ and $\gamma = 10$;
- Generally, it is recommended to set larger values for α and γ, while a smaller value for β.

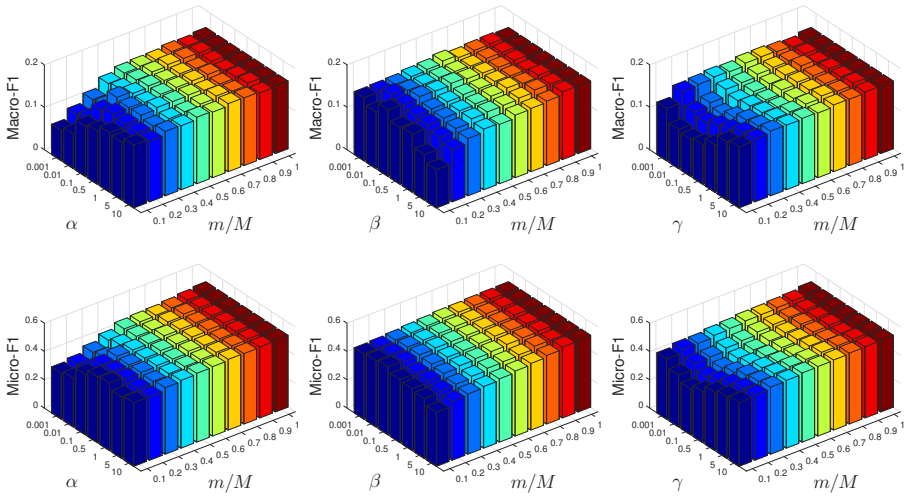


Figure 6.7: Parameter sensitivity analysis of α, β and γ on the Enron dataset by varying the percentage m/M of selected features from 10% to 100% by step 10%. The value of three parameters is selected from $\{0.001, 0.01, 0.1, 0.5, 1, 5, 10\}$.

Chapter 7

Label Space Dimension Reduction

7.1 Related Work

LS-DR consists of two main approaches: feature-unaware and feature-aware. The typical instance of feature-unaware approach is Principal Label Space Transformation (PLST), which can be considered as a counterpart of PCA in the label space. On the other hand, motivating by the high performance of supervised FS-DR, feature-aware LS-DR approaches are proposed. Based on the assumption of low-rank of label matrix, several embedding methods, such as Compressive Sensing [51], CPLST [14] and FaIE [53], encode the sparse label space by preserving label correlations and maximizing predictability of latent label space. By combining FS-DR and LS-DR, several methods have been proposed in recent years. WSABIE [99] learns a low-dimensional joint embedding space by approximately optimizing the precision on the top k relevant labels. By modeling MLC as a general empirical risk minimization problem with a low-rank constraint, LEML [104] scales to very large datasets even with missing labels. MLLEM [43] employs a nonlinear mapping to preserve three kinds of relationships, label-instance, label-label and instance-instance, in the embedded low-dimensional space. In the prediction phase, MLLEM first maps a test instance into such low-dimensional space, and then utilizes k-NN to output the prediction scores for labels.

7.2 Extending READER for Label Embedding (READER-LE)

Although the proposed READER method is designed for Feature Space Dimension Reduction (FSDR), It is worth noting that READER can be easily extended for *Label Space Dimension Reduction* (LSDR). To this end, we introduce a linear projection $\mathbf{P} \in \mathbb{R}^{L \times k}$ to approximate the non-linear label embedding $\mathbf{V}$ in (6.17), i.e., $\mathbf{V} = \mathbf{YP}$. In addition, in order to make round-based decoding[1] available, we relax the constraint $\mathbf{P}^\intercal \mathbf{Y}^\intercal \mathbf{D}_y \mathbf{YP} = \mathbf{I}$ as $\mathbf{P}^\intercal \mathbf{P} = \mathbf{I}$. Therefore, (6.17) becomes

$$\begin{aligned} &\min_{\mathbf{W},\mathbf{P}} \ \|\mathbf{XW} - \mathbf{YP}\|_{2,1} + \alpha \|\mathbf{W}\|_{2,1} + \beta \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} \mathbf{W}) + \gamma \mathrm{Tr}(\mathbf{P}^\intercal \mathbf{Y}^\intercal \mathbf{L}_y \mathbf{YP}), \\ &\text{s.t. } \mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{D}_x \tilde{\mathbf{X}} \mathbf{W} = \mathbf{P}^\intercal \mathbf{P} = \mathbf{I}. \end{aligned} \tag{7.1}$$

7.2.1 Formulation

Moreover, READER can *complete labels for the unlabeled data* by initializing such label vectors to zeros and update the label matrix in an iterative way. For convenience, we set $\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \\ \mathbf{X}_u \end{bmatrix} \in \mathbb{R}^{N \times M}$ and $\tilde{\mathbf{Y}} = \begin{bmatrix} \mathbf{Y} \\ \mathbf{Y}_u \end{bmatrix} \in \{0,1\}^{N \times L}$, where $\mathbf{X}_u$ and $\mathbf{Y}_u$ denotes the unlabeled feature and label matrix, respectively. Hence, the loss function is rewritten as

$$\sum_{i=1}^{N} c_i \left\|\mathbf{W}^\intercal \mathbf{x}_i - \mathbf{P}^\intercal \mathbf{y}_i\right\|_2 = \left\|\mathbf{C}(\tilde{\mathbf{X}}\mathbf{W} - \tilde{\mathbf{Y}}\mathbf{P})\right\|_{2,1}. \tag{7.2}$$

$\mathbf{C} \in \mathbb{R}^{N \times N}$ is a diagonal score matrix, whose ith diagonal element $\mathbf{C}_{ii} = c_i$ denotes the important of the ith training instance, $\forall i$. Here we simply assume that the labeled training instances should contribute more to the loss function than the unlabeled ones. Based on (7.2), (7.1) becomes,

$$\begin{aligned} &\min_{\mathbf{W},\mathbf{P},\mathbf{Y}_u} \ \left\|\mathbf{C}(\tilde{\mathbf{X}}\mathbf{W} - \tilde{\mathbf{Y}}\mathbf{P})\right\|_{2,1} + \alpha \|\mathbf{W}\|_{2,1} + \beta \mathrm{Tr}(\mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} \mathbf{W}) + \gamma \mathrm{Tr}(\mathbf{P}^\intercal \tilde{\mathbf{Y}}^\intercal \mathbf{L}_y \tilde{\mathbf{Y}} \mathbf{P}), \\ &\text{s.t. } \mathbf{W}^\intercal \tilde{\mathbf{X}}^\intercal \mathbf{D}_x \tilde{\mathbf{X}} \mathbf{W} = \mathbf{P}^\intercal \mathbf{P} = \mathbf{I}. \end{aligned} \tag{7.3}$$

7.2.2 Optimization Algorithm

By replacing the definition in (6.19) by the following,

$$\begin{aligned} \mathbf{A} &\coloneqq \begin{bmatrix} \mathbf{C}\tilde{\mathbf{X}} & -\mathbf{C}\tilde{\mathbf{Y}} \\ \alpha \mathbf{I}_m & \mathbf{0} \end{bmatrix}, \ \mathbf{B} \coloneqq \begin{bmatrix} \beta \tilde{\mathbf{X}}^\intercal \mathbf{L}_x \tilde{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \gamma \tilde{\mathbf{Y}}^\intercal \mathbf{L}_y \tilde{\mathbf{Y}} \end{bmatrix}, \\ \mathbf{E} &\coloneqq \begin{bmatrix} \tilde{\mathbf{X}}^\intercal \mathbf{D}_x \tilde{\mathbf{X}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_L \end{bmatrix}, \ \mathbf{U} \coloneqq \begin{bmatrix} \mathbf{W} \\ \mathbf{P} \end{bmatrix}, \end{aligned} \tag{7.4}$$

[1] round-based decoding maps the component of the vector to the closet value in $\{0,1\}$, denoted by round$(\cdot)$.

it is easy to show that Theorem 4 also holds for the optimization problem in (7.3). Similar with Algorithm 12, we propose Algorithm 14 to obtain the projection matrix $\mathbf{U}$ and the completed label matrix $\tilde{\mathbf{Y}}$.

Algorithm 14 Optimization algorithm for READER-LE

Input: $\tilde{\mathbf{X}} \in \mathbf{R}^{N\times M}$, $\mathbf{Y} \in \{0,1\}^{n\times L}$

Output: $\mathbf{U} = \begin{bmatrix}\mathbf{W}\\ \mathbf{P}\end{bmatrix} \in \mathbb{R}^{(M+N)\times k}$, $\tilde{\mathbf{Y}} = \begin{bmatrix}\mathbf{Y}\\ \mathbf{Y}_u\end{bmatrix} \in \{0,1\}^{N\times L}$

1: Calculate $\mathbf{L}_x$, and $\mathbf{E}$;
2: Set $t = 0$, initialize $\mathbf{H}_t = \mathbf{I}$ and $\tilde{\mathbf{Y}}_t = \begin{bmatrix}\mathbf{Y}\\ \mathbf{0}\end{bmatrix}$;
3: **repeat**
4: Calculate $(\mathbf{L}_y)_t$, $\mathbf{A}_t$ and $\mathbf{B}_t$ according to (7.4);
5: $\mathbf{F}_t \leftarrow \mathbf{A}_t^\intercal \mathbf{H}_t \mathbf{A}_t + \mathbf{B}_t$;
6: Solve $\mathbf{E}\mathbf{U}_t = \mathbf{F}_t\mathbf{U}_t\mathbf{\Lambda}$ with k largest eigenvalues;
7: Calculate $(\mathbf{K}_1)_t$ and $(\mathbf{K}_2)_t$ according to $(\mathbf{L}_y)_t$, $\mathbf{U}_t$ and $\tilde{\mathbf{Y}}_t$;
8: $\tilde{\mathbf{Y}}_{t+1} \leftarrow \begin{bmatrix}\mathbf{Y}\\ (\mathbf{Y}_u)_t\end{bmatrix}$, where $(\mathbf{Y}_u)_t \leftarrow \text{round}\left((\mathbf{K}_1)_t\mathbf{Y} + (\mathbf{K}_2)_t\mathbf{X}_u\mathbf{W}_t\mathbf{P}_t^\intercal\right)$;
9: Calculate $\mathbf{H}_{t+1}$, where its ith diagonal element is $\frac{1}{2\|(\mathbf{A}\mathbf{U}_t)_{i\cdot}\|_2}$;
10: $t \leftarrow t+1$;
11: **until** *Convergence*

7.2.3 Algorithm Analysis

We can show that the iterative algorithm in Algorithm 14 monotonically decreases the objective function in (7.3) in each iteration, and converges to the global optimum.

Theorem 6. *The iterative algorithm in Algorithm 14 (Steps 3 to 10) monotonically decreases the objective function in (7.3) in each iteration.*

Proof. Assume that after t iterations, we have $\mathbf{A}_t$, $\mathbf{B}_t$, $\mathbf{H}_t$, $\tilde{\mathbf{Y}}_t$ and $\mathbf{U}_t$. Note that $\mathbf{A}$ and $\mathbf{B}$ depends on $\tilde{\mathbf{Y}}$, while $\mathbf{H}$ depends on $\mathbf{U}$. Therefore, we verify Algorithm 6 by updating $\mathbf{U}$ and $\tilde{\mathbf{Y}}$, respectively.

In the next iteration, we first update $\mathbf{U}_{t+1}$ by fixing $\mathbf{A}_t$, $\mathbf{B}_t$, $\tilde{\mathbf{Y}}_t$ and $\mathbf{H}_t$. Similar with the proof of Theorem 5, it is easy to obtain the following inequality,

$$\|\mathbf{A}_t\mathbf{U}_{t+1}\|_{2,1} + \text{Tr}(\mathbf{U}_{t+1}^\intercal\mathbf{B}_t\mathbf{U}_{t+1}) \le \|\mathbf{A}_t\mathbf{U}_t\|_{2,1} + \text{Tr}(\mathbf{U}_t^\intercal\mathbf{B}_t\mathbf{U}_t). \tag{7.5}$$

In the similar way, we then update $\tilde{\mathbf{Y}}_{t+1}$, $\mathbf{A}_{t+1}$ and $\mathbf{B}_{t+1}$ by fixing $\mathbf{H}_t$ and $\mathbf{U}_t$. In order to reduce the value of the objective function by $\tilde{\mathbf{Y}}_{t+1}$, we set the derivative of (7.3) w.r.t. $\tilde{\mathbf{Y}}$ to zero, since (7.3) is a convex w.r.t. $\tilde{\mathbf{Y}}$. Thus, we have

$$\mathbf{M}\tilde{\mathbf{Y}}\mathbf{P}\mathbf{P}^\intercal - \mathbf{M}\tilde{\mathbf{X}}\mathbf{W}\mathbf{P}^\intercal + \gamma\mathbf{L}_y\tilde{\mathbf{Y}}\mathbf{P}\mathbf{P}^\intercal = \mathbf{0}, \tag{7.6}$$

where $\mathbf{M}$ is a diagonal matrix with ith diagonal element $(\mathbf{M})_{ii} = \frac{c_i^2}{2\left\|(\tilde{\mathbf{X}}\mathbf{W}-\tilde{\mathbf{Y}}\mathbf{P})_{ii}\right\|_2}$. Based on (7.6) and $\mathbf{P}^\intercal\mathbf{P} = \mathbf{I}$, we obtain $\tilde{\mathbf{Y}} = (\mathbf{M}+\gamma\mathbf{L}_y)^{-1}\mathbf{M}\tilde{\mathbf{X}}\mathbf{W}\mathbf{P}^\intercal$. We assume the label matrix $\mathbf{Y}$ for labeled instances is ground-truth, thereby updating only $\mathbf{Y}_u$ for $\tilde{\mathbf{Y}}$.

$$\begin{bmatrix}\mathbf{Y}\\ \mathbf{Y}_u\end{bmatrix} = \begin{bmatrix}\mathbf{G}_a & \mathbf{G}_b\\ \mathbf{G}_c & \mathbf{G}_d\end{bmatrix}\begin{bmatrix}\mathbf{X}\\ \mathbf{X}_u\end{bmatrix}\mathbf{W}\mathbf{P}^\intercal, \tag{7.7}$$

where $\mathbf{G} = (\mathbf{M}+\gamma\mathbf{L}_y)^{-1}\mathbf{M} = \begin{bmatrix}\mathbf{G}_a & \mathbf{G}_b\\ \mathbf{G}_c & \mathbf{G}_d\end{bmatrix}$. Solving (7.7) produces

$$\mathbf{Y}_u = \mathbf{K}_1\mathbf{Y} + \mathbf{K}_2\mathbf{X}_u\mathbf{W}\mathbf{P}^\intercal, \tag{7.8}$$

where $\mathbf{K}_1 = \mathbf{G}_c\mathbf{G}_a^{-1} \in \mathbf{R}^{(N-n)\times n}$ and $\mathbf{K}_2 = \mathbf{G}_d - \mathbf{G}_c\mathbf{G}_a^{-1}\mathbf{G}_b \in \mathbb{R}^{(N-n)\times(N-n)}$. (7.8) indicates that the predicted labels for the unlabeled data is inferred by a weight linear combination of ground truth label matrix $\mathbf{Y}$ and a mapping $\mathbf{X}_u\mathbf{W}\mathbf{P}^\intercal$ of unlabeled instances in the label space. In addition, we need regularize the updated value of $\mathbf{Y}_u$ by round-based decoding because of $\tilde{\mathbf{Y}} \in \{0,1\}^{N\times L}$. In this way, by updating $\tilde{\mathbf{Y}}_{t+1} \leftarrow \begin{bmatrix}\mathbf{Y}\\ (\mathbf{Y}_u)_t\end{bmatrix}$, where $(\mathbf{Y}_u)_t \leftarrow \text{round}((\mathbf{K}_1)_t\mathbf{Y} + (\mathbf{K}_2)_t\mathbf{X}_u\mathbf{W}_t\mathbf{P}_t^\intercal)$, we can obtain $\mathbf{A}_{t+1}$ and $\mathbf{B}_{t+1}$ according to (7.4). Then the following inequality holds,

$$\|\mathbf{A}_{t+1}\mathbf{U}_t\|_{2,1} + \text{Tr}(\mathbf{U}_t^\intercal\mathbf{B}_{t+1}\mathbf{U}_t) \le \|\mathbf{A}_t\mathbf{U}_t\|_{2,1} + \text{Tr}(\mathbf{U}_t^\intercal\mathbf{B}_t\mathbf{U}_t). \tag{7.9}$$

By integrating (7.5) and (7.9), we arrive at

$$\|\mathbf{A}_{t+1}\mathbf{U}_{t+1}\|_{2,1} + \text{Tr}(\mathbf{U}_{t+1}^\intercal\mathbf{B}_{t+1}\mathbf{U}_{t+1}) \le \|\mathbf{A}_t\mathbf{U}_t\|_{2,1} + \text{Tr}(\mathbf{U}_t^\intercal\mathbf{B}_t\mathbf{U}_t), \tag{7.10}$$

which shows that the value of the objective function in (7.3) monotonically decreases after each iteration. □

We depict the the READER method for LSDR in Algorithm 15.

Algorithm 15 The algorithm of READER-LE

Input: $\tilde{\mathbf{X}} \in \mathbb{R}^{N\times M}$, $\mathbf{X} \in \mathbb{R}^{n\times M}$, $\mathbf{Y} \in \{0,1\}^{n\times L}$, $\hat{\mathbf{x}} \in \mathbb{R}^M$
Output: $\hat{\mathbf{y}} \in \{0,1\}^L$

1: $\mathbf{U} = \begin{bmatrix}\mathbf{W}\\ \mathbf{P}\end{bmatrix} \leftarrow$ <Algorithm 14> $(\tilde{\mathbf{X}}, \mathbf{X}, \mathbf{Y})$;
2: $\mathbf{Z} = \mathbf{W}\mathbf{P}^\intercal$;
3: $\hat{\mathbf{y}} \leftarrow \text{round}(\mathbf{Z}^\intercal\hat{\mathbf{x}})$.

Chapter 8

Instance Space Decomposition

8.1 Related Work

Both FS-DR and LS-DR assume that feature-label relationship can be modeled on the whole training data, which probably contradicts real-world problems, harms classification accuracy and even results in high time complexity. To relax the assumption, the Instance Space Decomposition (ISD) methods are proposed, aiming to solve a complex problem by dividing it into multiple simpler ones. ISD has two advantages. First, simpler problems can be solved by simpler techniques, like transforming a global nonlinear problem into a local linear problem. Second, the training and testing can be more efficient, making the algorithm tractable for large-scale datasets.

8.1.1 Label-Guided Instance Space Decomposition

The ISD methods can be further separated into two groups, label-guided and feature-guided. For the label-guided approaches, the original dataset is decomposed into several subsets according to the hierarchical structure in the label space. Hierarchical Multi-Label Classification (HMC) [74, 5, 96] builds a hierarchy of single-label classifiers. Under the hierarchy constraint, the training data for each classifier is restricted so that it contains only the instances associated with parent labels. However, HMC's applications are limited on particular problems in text categorization and genomics analysis. Applying the same strategy of HMC, HOMER [93] breaks the constraint on the predefined label hierarchy. It builds the label hierarchy by recursively conducting balanced k-means on the label space, transforming the original task into a tree-shaped hierarchy of simpler tasks, each one relevant to a subset of labels. In [62], k sets of label

clustering are generated by applying spectral clustering on the label similarity, based on which the original dataset is divided into k data clusters.

8.1.2 Feature-Guided Instance Space Decomposition

The feature-guided ISD approaches aim to directly find data clusters by conducting clustering analysis in the feature space. CBMLC [59] partitions the original multi-label datasets into multiple small-scale datasets, on which multi-label classifiers are built individually. Given a test instance, it is feed only to the classifier corresponding to the nearest cluster. Different with CBMLC, CLMLC [85] performs data decomposition by applying clustering analysis on a feature subspace, and employs different local models for different data clusters.

In recent years, the extreme MLC problem, where the problem size in instances, features and labels scales to extreme large number, has attracted more and more attentions. Various MLC methods have been proposed to cope with the extreme case, such as MLRF [1], FastXML [66] and SLEEC [7]. Nearly all these extreme MLC methods actually employ the Instance Space Decomposition in their learning process. Multi-label Random Forest (MLRF) [1] learns an ensemble of randomized trees, where nodes are partitioned into a left and a right child by brute force optimization of a multi-label variant of the Gini index over labels. FastXML [66] builds a tree-based multi-label classifier, directly optimizing a novel ranking loss function, nDCG, and efficiently executing its formulation in light of an alternating minimization algorithm. To speed up the kNN classification, SLEEC [7] partitions the original training data into several clusters, learning a local nonlinear embedding per cluster and conducting kNN only within the test sample's nearest cluster.

8.2 Clustering-based Local MLC (CLMLC)

In this study, we put on two assumptions about the locality in MLC setting: (a) meta-labels, i.e. reasonable and strong label combinations, exist implicitly in the label space; (b) only a fraction of features and instances are relevant to a meta-label. These assumptions are supported by several observations. For example, in Enron dataset, 53 labels are categorized into only four meta-labels, and in image annotation, an object typically relates to only a few regions in the high-dimensional feature space.

Hence, we presume that MLC can be tackled by decomposing the original large-scale data into several regular-scale datasets, each of which is relevant to only several meta-labels in a feature subspace with a fraction of training instances. Based on this assumption, a Clustering-based Local MLC (CLMLC) method is proposed in this paper. CLMLC consists of two stages, low-dimensional

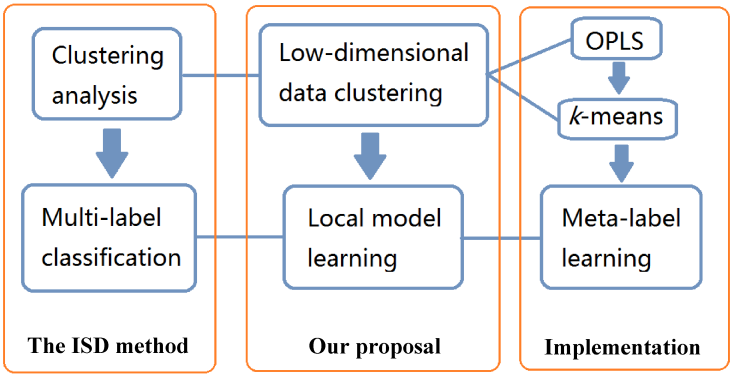


Figure 8.1: The framework of the proposed CLMLC.

data clustering and local model learning. In the first stage, a supervised dimension reduction is firstly conducted to project the original high-dimensional data into a low-dimensional feature subspace, while preserving feature-label correlation. Then clustering analysis is applied to partition the low-dimensional data into several regular-scale datasets. In the second stage, within each data cluster, meta-labels are mined by saving both label similarity and instance locality, and then classifier chains over meta-labels are built as the local MLC model. Given a test instance, prediction is made on the basis of the local model corresponding to its nearest data cluster. To empirically evaluate the performance of CLMLC, extensive experiments on regular/large-scale datasets from various domains are carried out with the state-of-the-art MLC algorithms.

8.2.1 Data Subspace Clustering

We assume that a large-scale dataset could be decomposed into several smaller local sets. To this end, clustering analysis is introduced to find the local clusters. However, directly applying cluster analysis would probably produce unstable outputs and suffer from high computational cost, especially when the dimensionality of the original feature space is relatively high. In this sense, a dimensionality reduction approach is necessary as a pre-processing technique before applying clustering analysis.

Let $\mathbf{X}$ and $\mathbf{Y}$ be already centered so as to $\mathbf{X}^\intercal\mathbf{1} = \mathbf{0}$ and $\mathbf{Y}^\intercal\mathbf{1} = \mathbf{0}$. The Partial Least Squares (PLS) [98] finds the directions of maximum covariance between $\mathbf{X}$ and $\mathbf{Y}$ by Singular Value Decomposition (SVD) as follows:

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{X}^\intercal\mathbf{Y} - \mathbf{U}\mathbf{\Lambda}_m\mathbf{V}^\intercal\|_F^2, \tag{8.1}$$

where $\mathbf{\Lambda}_m$ is a diagonal matrix $(\lambda_1, \lambda_2, ..., \lambda_m)$ with the largest m (the dimensionality of the feature subspace) singular values of $\mathbf{X}^\intercal\mathbf{Y}$, and $\|\cdot\|_F$ denotes the

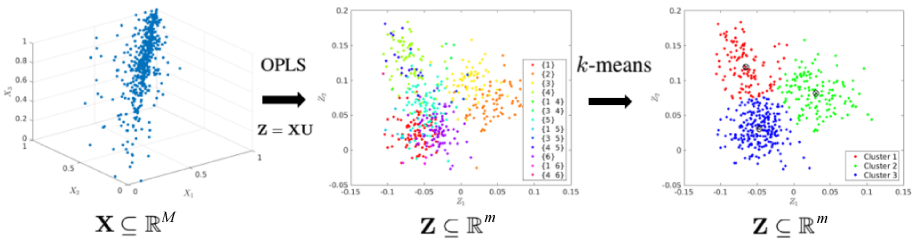


Figure 8.2: An Example of subspace clustering on the Scene dataset.

Frobenius norm. This is also the solution of the maximization problem:

$$\begin{aligned} \max_{\mathbf{U},\mathbf{V}} \quad & \mathrm{Tr}(\mathbf{U}^\intercal \mathbf{X}^\intercal \mathbf{Y}\mathbf{V}) \\ \text{s.t.} \quad & \mathbf{U}^\intercal\mathbf{U} = \mathbf{V}^\intercal\mathbf{V} = \mathbf{I}_m. \end{aligned} \tag{8.2}$$

One of limitations of PLS is the lack of invariance to arbitrary linear transformations on $\mathbf{X}$ [100].

To overcome this limitation, Orthonormalized PLS (OPLS) [100] is proposed by orthonormalizing $\mathbf{X}$ to $\mathbf{X}(\mathbf{X}^\intercal\mathbf{X})^{-\frac{1}{2}}$ in (8.1), and we have

$$\min_{\mathbf{U},\mathbf{V}} \left\| (\mathbf{X}^\intercal\mathbf{X})^{-\frac{1}{2}}\mathbf{X}^\intercal\mathbf{Y} - \mathbf{U}\mathbf{\Lambda}_m\mathbf{V}^\intercal \right\|_F^2. \tag{8.3}$$

Similar with (8.2), (8.3) can be also rewritten to a maximization problem:

$$\begin{aligned} \max_{\mathbf{U}} \quad & \mathrm{Tr}(\mathbf{U}^\intercal \mathbf{X}^\intercal \mathbf{Y}\mathbf{Y}^\intercal\mathbf{X}\mathbf{U}) \\ \text{s.t.} \quad & \mathbf{U}^\intercal\mathbf{X}^\intercal\mathbf{X}\mathbf{U} = \mathbf{I}. \end{aligned} \tag{8.4}$$

The solution $\mathbf{U}$ consists of eigenvectors $\mathbf{u}$ corresponding to the largest m eigenvalues of a generalized eigenvalue problem

$$(\mathbf{X}^\intercal\mathbf{Y}\mathbf{Y}^\intercal\mathbf{X})\mathbf{u} = \lambda(\mathbf{X}^\intercal\mathbf{X})\mathbf{u}. \tag{8.5}$$

To avoid the singularity of $\mathbf{X}^\intercal\mathbf{X}$ and reduce the model complexity, in practice a regularization term $\gamma\mathbf{I}$ with $\gamma > 0$ is commonly introduced to (8.5), leading to

$$(\mathbf{X}^\intercal\mathbf{Y}\mathbf{Y}^\intercal\mathbf{X})\mathbf{u} = \lambda(\mathbf{X}^\intercal\mathbf{X} + \gamma\mathbf{I})\mathbf{u}. \tag{8.6}$$

In general, directly solving the generalized eigenvalue problem (8.6) suffers from an expensive cost and thus might not scale to large-scale problems. In this study, we use an efficient two-stage approach [78] to address the problem. In the first stage, a penalized least squares problem is solved by regressing the centered feature matrix $\mathbf{X}$ to the centered label matrix $\mathbf{Y}$; after projecting $\mathbf{X}$ into the subspace by the regression, in the second stage, the resulting generalized eigenvalue problem is solved by SVD.

Algorithm 16 The algorithm of data subspace clustering in CLMLC

Input: $\mathbf{X}$: centered data matrix, $\mathbf{Y}$: centered label matrix, m: size of feature subspace, K: number of data clusters

Output: $\mathbf{U}$: projection matrix, $\mathbf{R}, \mathbf{C}$: clustering output

1: Solve the least squares problem:

$$\min_{\mathbf{U}_1} \|\mathbf{X}\mathbf{U}_1 - \mathbf{Y}\|_F^2 + \|\mathbf{U}_1\|_F^2;$$

2: $\mathbf{H} = \mathbf{U}_1^\intercal \mathbf{X}^\intercal \mathbf{Y}$;

3: Decompose $\mathbf{H} = \mathbf{U}_H \Lambda_m \mathbf{U}_H^\intercal$ by SVD;

4: $\mathbf{U} = \mathbf{U}_1 \mathbf{U}_2$, where $\mathbf{U}_2 = \mathbf{U}_H \Lambda_m^{-\frac{1}{2}}$;

5: $[\mathbf{R}, \mathbf{C}] \leftarrow k$-means$(\mathbf{Z}, K)$ by (8.7), where $\mathbf{Z} = \mathbf{X}\mathbf{U}$.

Through (8.6), we find an orthonormal basis $[\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_m]$ to form $\mathbf{U}$. Therefore we can have a low-dimensional expression $\mathbf{z} \in \mathbb{R}^m$ by projection $\mathbf{z} = \mathbf{U}^\intercal \mathbf{x}$, $\mathbf{Z} = \mathbf{X}\mathbf{U}$ as well. Then we conduct clustering on $\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N$ in the light of elimination of most of noisy features. In this paper, k-means is employed, aiming to approximately solve the following optimization problem:

$$\begin{aligned} \min_{\mathbf{R},\mathbf{C}} \quad & \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij} \|\mathbf{z}_i - \mathbf{c}_j\|_2^2 \\ \text{s.t.} \quad & \forall i,\ \|\mathbf{r}_i\|_0 = 1,\ \|\mathbf{r}_i\|_1 = 1, \end{aligned} \tag{8.7}$$

where $\mathbf{R}$ represents the $N \times K$ indicator matrix, indicating the assignment from data points to centroids, while the centroid matrix $\mathbf{C} = [\mathbf{c}_1, ..., \mathbf{c}_K]^\intercal$, whose $\mathbf{c}_j = \sum_i r_{ij}\mathbf{x}_i / \sum_i r_{ij}$. $\|\cdot\|_0$, $\|\cdot\|_1$ and $\|\cdot\|_2$ denote the ℓ_0, ℓ_1 and ℓ_2 norm, respectively. In general, k-means is realized as an iterative algorithm. The pseudo code of data subspace clustering is given in Algorithm 16.

8.2.2 Local Model Learning

In the second stage, we perform local model learning in each cluster. By expecting the existence of meta-labels, We use Laplacian eigenmap to learn meta-labels within each cluster, and then build classifier chains over meta-labels for local model learning. For each data cluster, we construct a graph $\boldsymbol{G} =< \boldsymbol{V}, \boldsymbol{E} >$ in the label space, where $\boldsymbol{V}$ is the vertex/label set, and $\boldsymbol{E}$ is the edge set containing edges between each label pair. Given an appropriate affinity matrix $\mathbf{A}$ on $\boldsymbol{E}$, meta-label learning can be considered as a graph cut problem: cutting the graph $\boldsymbol{G}$ into a set of sub-graphs.

For constructing affinity matrix $\mathbf{A}$, we use two different sources: the label space and the instance space. In this study, we utilize Jaccard index and heat kernel affinity to represent the label similarity and instance locality, respectively.

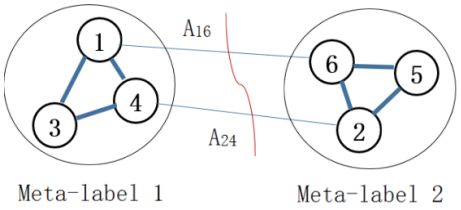


Figure 8.3: Meta-label learning on six labels.

- Label similarity $\mathbf{A}^{(L)} = \{A_{jk}^{(L)}\}_{j,k=1}^{L}$,

$$A_{jk}^{(L)} \coloneqq \frac{\sum_{i=1}^{N} y_{ij} y_{ik}}{\sum_{i=1}^{N} (y_{ij} + y_{ik} - y_{ij} y_{ik})}. \tag{8.8}$$

- Instance locality $\mathbf{A}^{(I)} = \{A_{jk}^{(I)}\}_{j,k=1}^{L}$,

$$A_{jk}^{(I)} \coloneqq \mathrm{e}^{-\|\boldsymbol{\mu}_j - \boldsymbol{\mu}_k\|_2^2}, \text{ where } \boldsymbol{\mu}_j = \frac{\sum_{i=1}^{N} y_{ij} \mathbf{z}_i}{\sum_{i=1}^{N} y_{ij}}. \tag{8.9}$$

By combining these two matrices, we obtain the following affinity matrix $\mathbf{A} = \{A_{jk}\}_{j,k=1}^{L}$,

$$A_{jk} \coloneqq \frac{1}{2}\left(A_{jk}^{(L)} + A_{jk}^{(I)}\right). \tag{8.10}$$

To cut the graph $\mathbf{G}$ into n sub-graphs (n meta-labels) is equivalent to perform k-means on the n smallest eigenvectors $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_n]$ of the generalized eigenvalue problem:

$$\mathbf{L}\mathbf{w} = \lambda \mathbf{D}\mathbf{w}, \tag{8.11}$$

where $\mathbf{D} = (D_{jj}) = (\sum_k A_{jk})$, and $\mathbf{L}$ is the Laplacian matrix, $\mathbf{L} = \mathbf{D} - \mathbf{A}$. Thus, the label assignment to n meta-labels is obtained by applying k-means on the rows of $\mathbf{W}$.

After finding meta-labels, a sophisticated multi-label classifier h could be applied to capture the strong label correlations within each meta-label. On the other hand, to model relatively weak meta-label correlations, a simple MLC method is also necessary in the meta-label space. In this way, label correlations can be well captured without much time cost. To this end, we introduce an efficient classifier chains method [69] over the meta-label space. In general, for each meta-label within a meta-label chain, we expand its training data by taking previous meta-labels as extra features before feeding the data into the classifier h. The outline of local model learning is given in Algorithm 17.

8.2.3 Prediction

Given a test instance $\hat{\mathbf{x}} \in \mathbb{R}^M$, the prediction can be made by two steps. Firstly, $\hat{\mathbf{x}}$ is encoded into the feature subspace by $\hat{\mathbf{z}} = \mathbf{U}^\intercal \hat{\mathbf{x}} \in \mathbb{R}^m$. Secondly, the local

Algorithm 17 The algorithm of local model learning in CLMLC

Input: $\mathbf{Z^c}$: local data matrix, $\mathbf{Y^c}$: local label matrix, n: number of meta-labels,

Output: $\mathbf{h^c}$: local classifier

1: Compute $\mathbf{A}$ according to (8.10);
2: $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D} = \text{diag}(\sum_\ell A_{\ell m})$;
3: Solve $\mathbf{Lw} = \lambda \mathbf{Dw}$ by n smallest eigenvalues;
4: $\mathbf{R^c} \leftarrow k\text{-means}(\mathbf{W}, n)$, where $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_n]$;
5: **for** $k \in \{1, ..., n\}$ **do**
6: $\quad id = \text{find}(\mathbf{R^c} == k)$
7: $\quad h_k^{\mathbf{c}} : \mathbf{Z^c} \mapsto \mathbf{Y^c}(:, id))$;
8: $\quad \mathbf{Z^c} = \mathbf{Z^c} \cup \mathbf{Y^c}(:, id)$;
9: $\mathbf{h^c} \leftarrow \{h_k^{\mathbf{c}}\}_{k=1}^n$.

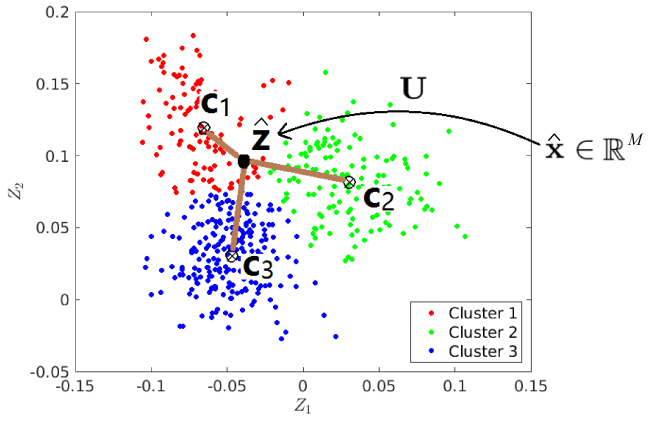


Figure 8.4: Prediction on a test instance by activating the classifier corresponding to its nearest cluster $\mathbf{c}_1$.

classifier $\mathbf{h^c}$ corresponding to $\hat{\mathbf{z}}$'s nearest cluster $\mathbf{c}$ such as,

$$\mathbf{c} = \arg\min_{\mathbf{c} \in \mathbf{C}} \|\hat{\mathbf{z}} - \mathbf{c}\|_2^2 , \tag{8.12}$$

is activated to predict the label assignment by $\hat{\mathbf{y}} = \mathbf{h^c}(\hat{\mathbf{z}})$. Note that $\mathbf{C}$ in (8.12) is the centroid matrix obtained according to (8.7).

Remarks

The complete procedure of CLMLC, including training (Steps 1 to 5) and testing (Steps 6 to 8), is outlined in Algorithm 18. It is worth noting that CLMLC is able to serve as a *meta-strategy* for large-scale MLC problems. For example, other dimension reduction or clustering analysis techniques could be used to replace the OPLS or k-means in Algorithm 16, in order to handle specific problem

Algorithm 18 The algorithm of CLMLC

Input: $\mathbf{X}$: centered data matrix, $\mathbf{Y}$: centered label matrix, $\hat{\mathbf{x}}$: test instance, m: size of feature subspace, K: number of data clusters, n: number of meta-labels

Output: $\hat{\mathbf{y}}$: predicted label set

Training:

1: $[\mathbf{U}, \mathbf{R}, \mathbf{C}] \leftarrow$ <Algorithm 16> $(\mathbf{X}, \mathbf{Y}, m, K)$;
2: $\mathbf{Z} = \mathbf{X}\mathbf{U}$;
3: **for** $\mathbf{c} \in \mathbf{C}$ **do**
4: Find local dataset $[\mathbf{Z^c}, \mathbf{Y^c}]$ by $\mathbf{R}$;
5: $\mathbf{h}^c \leftarrow$ <Algorithm 17> $(\mathbf{Z^c}, \mathbf{Y^c}, n)$;

Testing:

6: $\hat{\mathbf{z}} = \mathbf{U}^\intercal \hat{\mathbf{x}}$;
7: Find $\hat{\mathbf{z}}$'s nearest cluster $\mathbf{c}$ by (8.12);
8: $\hat{\mathbf{y}} \leftarrow \mathbf{h^c}(\hat{\mathbf{z}})$;

settings or data patterns. Similarly, any MLC method can be directly applied for local model learning in Algorithm 17. It shows the high flexibility of CLMLC to address various MLC problems.

8.3 Experimental Results

8.3.1 Comparison with State-of-the-Art

The proposed **CLMLC** method was compared with four state-of-the-art MLC methods:

- **ECC** [69]: an ensemble of classifier chains, where chain orders are generated randomly. Each classifier of a single CC is trained by taking previously assigned labels as extra attributes.
- **MLHSL** [79]: an FS-DR MLC method. A dataset is encoded by mapping features into a subspace, and then an MLC method is built on the basis of the encoded dataset.
- **CPLST** [14]: an LS-DR MLC method. The label space is encoded by a feature-aware principal label space transformation, and the round-based decoding [14] is used to predict the label set.
- **CBMLC** [59]: a first attempt on applying clustering analysis on the dataset before feeding the data to a multi-label classifier.

ECC is adopted due to its superior performance compared with other MLC decomposition methods, such as BR [8] and CC [69], as shown in [69]. As global

MLC methods, MLHSL is chosen as a representative of FS-DR methods, while CPLST is chosen by its performance advantage, especially in Hamming-Score, over several LS-DR methods, such as Compressive Sensing, PLST and orthogonally constraint CCA, as shown in [14]. As a local MLC method, CBMLC is selected for comparison in cluster analysis. Note that SLEEC [7] is excluded from the comparing methods, although it employs the similar local strategy with CLMLC. This is because SLEEC focuses on extreme MLC [66], where standard multi-label evaluation metrics like our four metrics are not appropriate.

In the experiments, *5-fold cross validation* was performed to evaluate the classification performance. For fair comparison, CC with ridge regression was used as the baseline classifier for CBMLC, MLHSL, CPLST and CLMLC. In parameter setting, for CLMLC, we set the dimensionality of feature subspace m by $\min\{L, 30\}$, and the number of clusters K by 20/100 for regular/large-scale datasets, respectively. For a cluster $\mathbf{c}$, the number of meta-labels n was set to $\lceil L^{\mathbf{c}}/5 \rceil$. CLMLC employed an ensemble of 2 CCs as the meta-label classifier. ECC used an ensemble of 10 CCs. In addition, in order to scale up ECC, random sampling was applied to randomly select 75% of instances and 50% of features for building each CC in ECC, as recommended in [69]. CBMLC and MLHSL shared the same value of K and m with CLMLC, respectively. For CPLST, we set the ratio for LS-DR by 0.8/0.6 for regular/large-scale datasets, respectively. Note that the parameters were chosen for the comparing methods in order to balance the classification accuracy and execution time, according to the experimental results on conducting grid search in the parameter spaces (detailed discussion will be made in Section 8.3.2). We obtained the MATLAB codes of CPLST and MLHSL from the authors, and implemented the MATLAB codes of ECC[3], CBMLC[3] and CLMLC[3] by ourselves. Experiments were performed in a computer configured with an Intel Quad-Core i7-4770 CPU at 3.4GHz with 4GB RAM.

Experimental results of five comparing MLC methods on benchmark datasets are reported in Table 8.1, where the averaged rank of each method over all datasets is shown in the last row of each metric. For each evaluation metric, the larger the value, the better the performance. Among the five comparing methods, the best performance is highlighted in boldface.

[3] https://github.com/futuresun912/CLMLC.git

Table 8.1: Experimental results (mean±std.) on eighteen multi-label datasets in four evaluation metrics.

Method	Birds	Genbase	Medical	Enron	Scene	Yeast	Corel5k	Rcv1s1	Rcv1s2	Rcv1s3	Rcv1s4	Bibtex	Corel16k1	Corel16k2	Corel16k3	Corel16k4	Corel16k5	Delicious
Exact-Match																		
CC	.507±.014	.980±.006	.604±.012	.102±.013	.592±.014	.197±.009	.008±.001	.133±.005	.242±.008	.247±.006	.350±.007	.165±.002	.011±.001	.009±.003	.009±.001	.009±.001	.009±.002	.002±.000
MLHSL	**.524±.013**	**.982±.005**	.676±.010	.120±.009	.596±.014	.196±.008	.004±.001	.114±.003	.220±.009	.219±.009	.320±.006	.119±.003	.009±.001	.008±.003	.007±.001	.007±.001	.008±.002	.002±.000
CPLST	.502±.015	.980±.005	.583±.011	.092±.010	.479±.010	.149±.009	.005±.000	.063±.003	.176±.005	.173±.005	.290±.007	.148±.003	.007±.001	.006±.001	.006±.001	.006±.001	.007±.001	.001±.000
CBMLC	.375±.017	.973±.004	.578±.016	.101±.006	.553±.009	.163±.005	.012±.001	.163±.007	.255±.008	.248±.006	.326±.009	.164±.006	.016±.001	.014±.001	.017±.001	.017±.001	.015±.001	**.012±.001**
CLMLC	**.524±.009**	.979±.004	**.688±.014**	**.147±.008**	**.627±.012**	**.205±.009**	**.025±.003**	**.224±.003**	**.316±.004**	**.319±.007**	**.400±.003**	**.172±.003**	**.030±.002**	**.029±.002**	**.030±.001**	**.028±.001**	**.030±.001**	.004±.000
Hamming-Score																		
CC	.949±.002	**.999±.000**	.987±.000	.911±.002	.875±.005	.786±.003	.990±.000	**.973±.000**	.976±.000	**.977±.000**	.981±.000	.987±.000	**.981±.000**	.982±.000	**.982±.000**	.982±.000	**.982±.000**	.981±.000
MLHSL	**.954±.002**	**.999±.000**	**.990±.000**	.936±.001	.875±.005	.786±.003	.990±.000	**.973±.000**	**.977±.000**	**.977±.000**	.981±.000	.986±.000	**.981±.000**	.982±.000	**.982±.000**	.982±.000	**.982±.000**	.981±.000
CPLST	.950±.002	**.999±.000**	.986±.000	.911±.002	**.887±.002**	**.797±.004**	**.991±.000**	**.973±.000**	**.977±.000**	**.977±.000**	**.982±.000**	**.988±.000**	**.981±.000**	**.983±.000**	**.982±.000**	**.983±.000**	**.982±.000**	**.982±.000**
CBMLC	.887±.005	**.999±.000**	.987±.001	.930±.001	.869±.004	.750±.002	.988±.000	.966±.000	.971±.001	.969±.001	.976±.001	.986±.000	.972±.001	.976±.000	.975±.000	.975±.001	.975±.000	.976±.000
CLMLC	**.954±.002**	**.999±.000**	.988±.000	**.940±.001**	.885±.004	.779±.003	.986±.000	.969±.000	.973±.000	.973±.000	.979±.000	.986±.000	.977±.000	.979±.000	.978±.000	.979±.000	.979±.000	.979±.000
Macro-F1																		
CC	.316±.015	**.767±.015**	**.381±.021**	.192±.003	.654±.014	.361±.008	.017±.001	.139±.008	.126±.003	.131±.005	.120±.002	.215±.001	.018±.001	.018±.001	.020±.001	.015±.001	.016±.001	.037±.001
MLHSL	.302±.007	**.767±.015**	.354±.009	.160±.007	.648±.013	.354±.009	.010±.001	.104±.006	.096±.004	.091±.004	.089±.001	.095±.002	.014±.001	.014±.001	.014±.001	.010±.001	.012±.001	.025±.001
CPLST	.287±.011	.761±.012	.374±.015	.167±.004	.639±.008	.351±.007	.016±.001	.125±.003	.110±.003	.108±.004	.108±.003	.186±.001	.015±.001	.018±.001	.021±.002	.013±.000	.015±.001	.048±.000
CBMLC	.188±.012	.738±.019	.312±.007	**.195±.006**	.655±.011	**.418±.006**	.032±.001	.204±.009	.195±.005	.185±.004	.176±.005	**.257±.004**	**.068±.002**	**.063±.001**	**.058±.002**	**.070±.000**	**.060±.002**	**.143±.001**
CLMLC	**.369±.009**	.761±.017	.358±.016	.153±.005	**.689±.009**	.400±.010	**.041±.001**	**.215±.002**	**.210±.004**	**.198±.003**	**.189±.003**	.210±.002	.056±.002	.057±.002	.053±.002	.051±.003	.047±.001	.067±.002
Micro-F1																		
CC	.457±.013	.991±.003	.770±.006	.413±.004	.645±.014	.626±.008	.163±.009	.356±.010	.370±.006	.373±.013	.441±.009	.392±.003	.107±.013	.102±.008	.086±.010	.107±.010	.106±.011	.107±.004
MLHSL	.452±.016	**.992±.002**	**.812±.003**	**.483±.006**	.640±.013	.627±.009	.140±.012	.310±.010	.330±.010	.317±.013	.392±.009	.279±.003	.089±.014	.084±.010	.065±.010	.085±.010	.090±.011	.063±.003
CPLST	.450±.014	**.992±.002**	.756±.003	.414±.005	.635±.007	.631±.009	.106±.003	.349±.004	.371±.008	.365±.007	.440±.009	.382±.003	.070±.002	.078±.001	.079±.002	.070±.002	.073±.001	.194±.001
CBMLC	.265±.011	.988±.002	.740±.014	.463±.005	.641±.012	.581±.006	.151±.005	.371±.005	.387±.003	.376±.008	.426±.007	.393±.003	.163±.003	.157±.002	.154±.002	**.161±.003**	**.156±.003**	**.268±.002**
CLMLC	**.474±.009**	.987±.004	.782±.007	.480±.009	**.676±.011**	**.632±.007**	**.192±.004**	**.401±.002**	**.423±.005**	**.422±.006**	**.472±.003**	**.396±.002**	**.164±.004**	**.164±.004**	**.160±.005**	.157±.003	.148±.002	.214±.006

For all the 72 configurations (18 datasets $\times$ 4 evaluation metrics), CLMLC ranked 1st among five comparing MLC methods at 37.8% cases, ranked 2nd at 18.9% cases, and ranked 5th at only 3.3% cases, which was remarkably better than the other methods. Specifically, CLMLC outperformed the other methods in Exact-Match (ranked 1st at 88.9% cases) and Micro-F1 (ranked 1st at 55.6% cases), and was competitive in terms of Macro-F1 (ranked 1st/2nd at 33.3%/61.1% cases). It demonstrates the effectiveness of the clustering-based local strategy adopted in CLMLC. The similar instances with similar label sets can be grouped together by CLMLC, leading to its strong capability on modeling label correlations and thus superior performance in Exact-Match. However, such grouped local data sometimes weaken the influence of minority labels, resulting in the worse performance of CLMLC in Hamming-Score (ranked 4nd at 66.7% cases). CPLST and ECC performed better than the other methods in Hamming-Score (ranked 1st at 38.9% and 16.7% cases, respectively), since it is designed to be optimized in Hamming-Score, according to the theoretical analysis in [14]. In Hamming-Score, MLHSL ranked in 1st/2nd place at 11.1%/61.1% cases, but performed worse in other metrics, especially on large-scale datasets. It is probably because large-scale datasets typically need a sufficient number of instances for training, while FS-DR tends to remove too many features. CBMLC outperformed other methods except CLMLC in Exact-Match (ranked 1st/2nd at 5.6%/55.5% cases), Macro-F1 (ranked 1st/2nd at 44.4%/33.3% cases) and Micro-F1 (ranked 1st/2nd at 16.7%/44.4% cases), but worked worst in Hamming-Score (ranked 5th at 72.2% cases). In addition, CBMLC worked worse than CLMLC on the average in all the four metrics, indicating that cluster analysis should be applied after appropriate feature dimension reduction. Note that the two local MLC methods, CLMLC and CBMLC, worked remarkably better than ECC, MLHSL and CPLST in terms of Exact-Match, Macro-F1 and Micro-F1 on the twelve large-scale datasets, demonstrating the superiority of local MLC strategy on real-world problems.

The execution time on seven large-scale datasets, including both training and prediction time, is reported in Table 8.2. The least time cost is highlighted in boldface. Among all the methods, MLHSL needed the least execution time on the average due to the low-dimensional feature subspace induced by FS-DR. CLMLC consumed the second least time on the average. Note that, CLMLC paid only slightly higher time cost than MLHSL on the Corel16k datasets. On datasets with large number of labels (large values in L), like delicious, CLMLC consumed more execution time than MLHSL and CPLST. Benefiting from LS-DR, CPLST cost the third least execution time, which was significantly less than ECC and CBMLC. But such superiority of CPLST decreased as the number of features increased (large values in D), like Bibtex. Due to its clustering analysis directly applied on high-dimensional datasets, CBMLC consumed the

Table 8.2: Execution time (10^3sec) of comparing methods over seven large-scale datasets.

	Corel5k	Rcv1s1	Rcv1s2	Bibtex	Corel16k1	Corel16k2	Delicious
ECC	0.353	0.190	0.187	1.285	0.229	0.252	6.042
MLHSL	0.018	**0.004**	**0.004**	0.015	**0.008**	**0.009**	**0.528**
CPLST	0.042	0.045	0.036	0.223	0.042	0.042	0.558
CBMLC	0.097	0.112	0.127	1.002	0.131	0.151	1.916
CLMLC	**0.005**	**0.004**	**0.004**	**0.014**	0.010	0.010	0.567

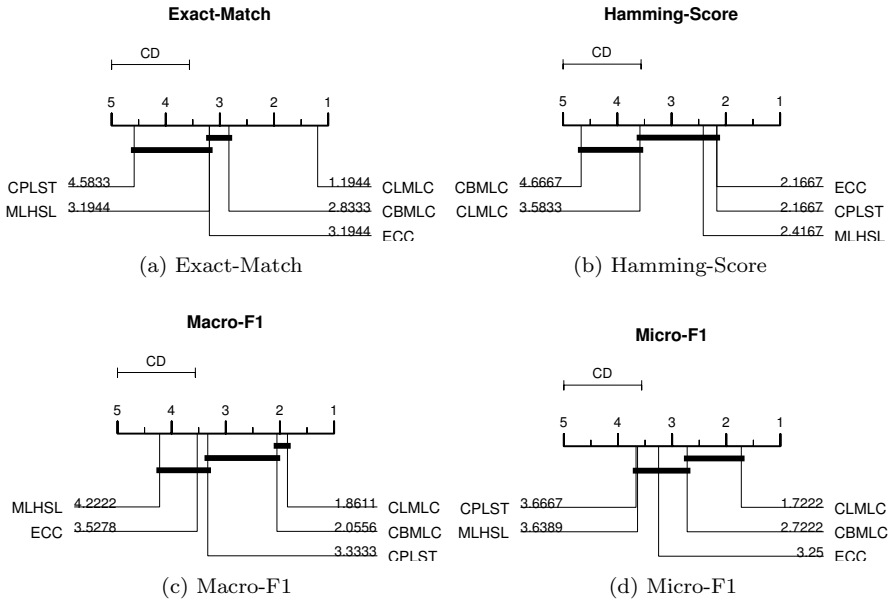
second largest time on all the datasets. ECC consumed the largest time on all the seven datasets, resulting from the ensemble strategy. In summary, the proposed CLMLC is one of the best choices for MLC in the balance of performance and execution time, especially when Exact-Match or Macro/Micro-F1 is the principal goal and the practical processing speed is required in a large-scale problem.

To derive a more objective insistence on the experimental results, we conducted *Friedman test* [21] with significance level 0.05 (5 methods, 18 datasets). The results are shown in Table 8.3. Since the values of the Friedman Statistic F_F in terms of all metrics were higher than the Critical Value, the null hypothesis of equal performance was rejected. Then, we proceeded to a *Nemenyi testing* to confirm the difference between any two methods. According to [21], the performance of two methods is regarded as significantly different if their average ranks differ by at least the Critical Difference (CD). Figure 8.5 shows the CD diagrams for four evaluation metrics at 0.05 significance level. In each subfigure, the value of CD is given as a rule above the axis, where the averaged rank is marked. In Figure 8.5, the algorithms which are not significantly different are connected by a thick line. In summary, among 90 comparisons (5 methods $\times$ 18 datasets), CLMLC achieved statistically superior performance than all the other methods in terms of Exact-Match. In Macro/Micro-F1, CLMLC achieved statistically comparable performances with CBMLC, and statistically superior performances than ECC, MLHSL and CPLST. Such observation demonstrates the competing performance of the proposed CLMLC in Exact-Match and Macro/Micro-F1, compared with the state-of-the-art MLC methods.

Table 8.4 reports the reduced sizes of training datasets in CLMLC, which are averaged by 5-fold cross validation. Here "std." shows the standard deviation of the values from K clusters. As shown in Table 8.4, consistently with our previous assumptions, there is strong locality in datasets, especially on datasets in text domain, like Medical, Rcv1 and Bibtex, where $\overline{L^{\mathbf{c}}} \ll L$ in each data cluster $\mathbf{c}$. Indeed the problem sizes in terms of N, M and L have been significantly reduced. For example, in Bibtex, the average problem size ($m \times \overline{N^{\mathbf{c}}} \times \overline{L^{\mathbf{c}}}$) in each cluster $\mathbf{c}$ has been reduced to nearly 1/30000 by CLMLC compared with

Table 8.3: Results of the Friedman Statistics F_F (5 methods, 18 datasets) and the Critical Value (0.05 significance level). The null hypothesis as the equal performance is rejected, if the values of F_F in terms of all metrics are higher than the Critical Value.

Friedman Test	Exact Match	Hamming Score	Macro F1	Micro F1
F_F	24.166	15.992	11.680	6.051
Critical Value		2.507		



(a) Exact-Match (b) Hamming-Score

(c) Macro-F1 (d) Micro-F1

Figure 8.5: CD diagrams (0.05 significance level) of five comparing methods.

Table 8.4: Problem sizes of training datasets in CLMLC. The values were averaged by 5-fold cross validation. Here "std." denotes the standard deviation.

Dataset	Original size			Reduced size			
	N	M	L	$\overline{N^{\mathbf{c}}} \pm$ std.	m	$\overline{L^{\mathbf{c}}} \pm$ std.	K
Birds	516	260	19	25.80±45.36	19	7.24±2.93	20
Genbase	530	1186	27	26.48±45.28	27	2.78±2.72	20
Medical	782	1449	45	39.12±39.47	30	4.68±2.97	20
Enron	1362	1001	53	68.08±66.81	30	23.85±6.54	20
Scene	1926	294	6	96.28±30.61	6	4.75±1.33	20
Yeast	1934	103	14	96.68±18.49	14	13.31±0.69	20
Corel5k	4000	499	374	40.00±18.18	30	54.08±25.19	100
Rcv1s1	4800	944	101	48.00±28.92	30	19.54±12.62	100
Rcv1s2	4800	944	101	48.00±31.34	30	18.51±11.78	100
Rcv1s3	4800	944	101	48.00±30.69	30	18.13±12.00	100
Rcv1s4	4800	944	101	48.00±33.80	30	14.36±9.94	100
Bibtex	5916	1836	159	59.16±39.44	30	29.95±20.41	100
Corel16k1	11013	500	164	110.13±42.59	30	71.31±22.18	100
Corel16k2	11009	500	164	110.09±48.86	30	71.32±24.37	100
Corel16k3	11008	500	154	110.08±44.93	30	69.11±22.39	100
Corel16k4	11070	500	162	110.70±48.46	30	70.73±22.91	100
Corel16k5	11078	500	160	110.78±46.55	30	72.82±23.64	100
Delicious	12884	500	983	128.84±155.55	30	333.48±200.60	100

the original set, bringing the fastest execution time on Bibtex, as shown in Table 8.2.

8.3.2 Parameter sensitivity analysis

To evaluate the potentiality of CLMLC, a parameter sensitivity analysis was conducted. First, the parameters m and K were dealt with the Rcv1s1 and Bibtex datasets, where m controls the dimensionality of the feature subspace, and K is the number of data clusters. In this experiment, we kept the value of n by $\lceil L^{\mathbf{c}}/5 \rceil$, and increased m from 5 to 100 by step 5, and K from 10 to 200 by step 10. Figure 8.6 shows the experimental results in terms of four evaluation metrics, whose values are averaged by 5-fold cross validation. In Figure 8.6, the warmer the color, the better the performance. We observe that as the values of m and K increased, its performance in Exact-Match and Macro/Micro-F1 upgraded, and then became stable once m and K reached 30 and 100, respectively. In contrast, as the values of m and K increased, its performance in Hamming-Score degraded, although the change was very slight (within 0.5%).

To optimize the parameters of MLHSL, CPLST and CBMLC, another set of

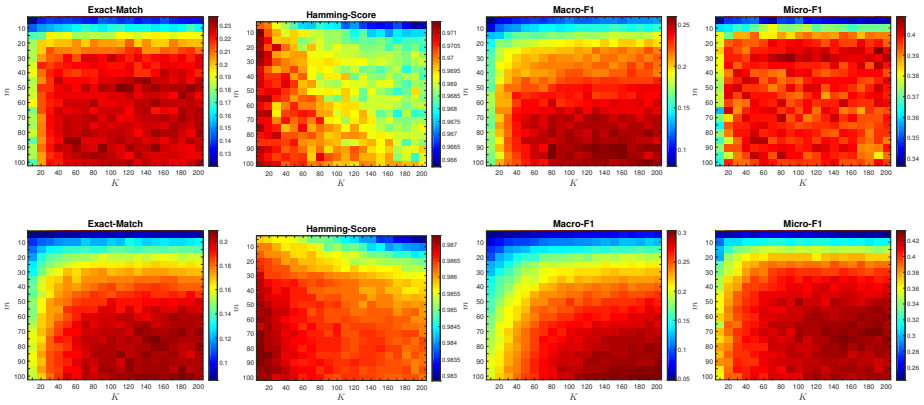


Figure 8.6: Parameter sensitivity analysis over the dimensionality m of feature subspace and the number K of clusters on Rcv1s1 (the first row) and Bibtex (the second row) ($n = \lceil L^{\mathbf{c}}/5 \rceil$). The size of m/K was increased from 5/10 to 100/200 by step 5/10.

parameter sensitivity analysis has been performed individually. Specifically, for MLHSL, m shared the similar tendency with CLMLC. For CPLST, the ratio of LS-DR remarkably influenced the experimental results. As the ratio increased, its performance upgraded. As the ratio approached 0.8/0.6 on regular/large-scale datasets, the performance became stable, while execution time increased dramatically. For CBMLC, as the number of cluster K increased, the values of evaluation metrics, except Hamming-Score, increased and became stable as K approached 100. Such observations validate the effectiveness of parameter configurations.

Part III

Conclusion

Chapter 9

Summary and Future Work

The thesis is concluded in this Chapter. Section 9.1 summarizes this thesis and briefly discusses the proposed methods. Section 9.2 presents the contributions achieved by the research work in this thesis. Finally, Section 9.3 discusses the future work motivated by the thesis.

9.1 Summary of the thesis

This thesis focuses on the problem of Multi-Label Classification (MLC) and Multi-Label Dimension Reduction (ML-DR), and proposes several approaches in order to overcome the limitations of existing MLC and ML-DR methods. According to the organization of this thesis, we summarizes it in the following four parts.

- **Introduction**. In Chapter 1, we introduce the background of our research on MLC by showing the difference between traditional classification and MLC. Then we discuss the current problems facing with the existing MLC methods, leading to the concerns of this thesis and the motivation of our research, i.e., to improve the classification performance by modeling label correlations and conducting dimension reduction. For the convenience of following statement, the notations frequently used in this thesis are listed. In the last section of this chapter, we show the structure of this thesis.
- **Part I**. This part concentrates on handling MLC problems via Classifier Chains (CC) based methods, which has been demonstrated the efficiency on capturing label correlations. There are three chapters in this part.
  - Chapter 2 gives a detailed introduction on MLC with mathematical definition via risk minimization, and presents the main idea of CC with a brief discussion on the related works.

- Motivated by CC, Chapter 3 proposes the Polytree-Augmented Classifier Chains (PACC) method, which aims to capture label correlations more precisely and avoid the problem of error propagation. To improve the performance and reduce time complexity on classification, Chapter 3 develops a two-stage feature selection framework for PACC, termed PACC-LDF, and demonstrates its efficiency of both PACC and PACC-LDF compared with their counterparts of CC-based methods by extensive empirical evidences and statistical tests.
- From a viewpoint of conditional likelihood minimization, Chapter 4 generalizes the existing CC-based methods as well as several information theoretic multi-label feature selection approaches. Several corollaries have been induced from the theorems proposed here, which helps to illustrate the experimental results obtained by previous papers on CC and guides the further research on CC. Based on the proposed MLC framework, Optimized CC (OCC) is proposed by selecting relevant parent labels (label correlation modeling) and mining label-specific features (feature selection).

- **Part II**. This part focuses on ML-DR, and presents our research achievements in terms of Feature Space Dimension Reduction (FS-DR), Label Space Dimension Reduction (LS-DR) and Instance Space Decomposition (ISD). Specifically, this part consists of four chapters.
  - Chapter 5 makes an introduction on ML-DR, and categorizes the existing methods into three groups: FS-DR, LS-DR and ISD, corresponding to performing dimension reduction on the feature space, the label space and the instance space, respectively. The main idea of each group of methods is presented by algorithms.
  - Chapter 6 focuses on FS-DR, and proposes two methods, MLC with Meta-Label-Specific Features (MLSF) and Robust sEmi-supervised multi-lAbel DimEnsion Reduction (READER), to demonstrate the effectiveness of conducting FS-DR on MLC. MLSF decomposes FS-DR into two stages, meta-label learning and specific feature selection, in order to capture label correlations and select label-specific features, respectively. In contrast, READER treats FS-DR as the problem of empirical risk minimization. Benefiting from the $\ell_{2,1}$-norm and manifold learning, READER selects discriminative features across labels in a semi-supervised way. To optimize its objective function, an efficient algorithm is developed with convergence property.
  - Chapter 7 presents our research work on LS-DR. The proposed method READER-LE can be viewed as an extension of READER. The main idea is to linearly approximate the nonlinear label embedding in the

original READER, which enables us to simply decode both training and testing instances. Based on a similar optimization algorithm of READER, the label encoding matrix can be obtained with proved convergence to its optimum. Note that READER-LE is the first method on conducting LS-DR in a semi-supervised manner.

  - Chapter 8 proposes a Clustering-based Local MLC (CLMLC) method by following the ISD strategy. Specifically, CLMLC consists of two stages, subspace clustering and local model learning. In the first stage, instances are first projected into a feature subspace found by a supervised FS-DR approach. Then similar instances associated with similar labels are grouped together by performing k-means on the feature subspace. In local model learning, under the assumption on the existence of meta-labels, we treat meta-label learning as a graph cut problem, which can be solved by a generalized eigenvalue problem. Extensive experiments conducted on real-world benchmark datasets verified our assumption and demonstrated the performance superiority of CLMLC compared with state-of-the-art ML-DR methods.

- **Part III**. In this part, we conclude this thesis in terms of contents and contributions. In addition, we discuss the future work motivated by our research work introduced in this thesis.

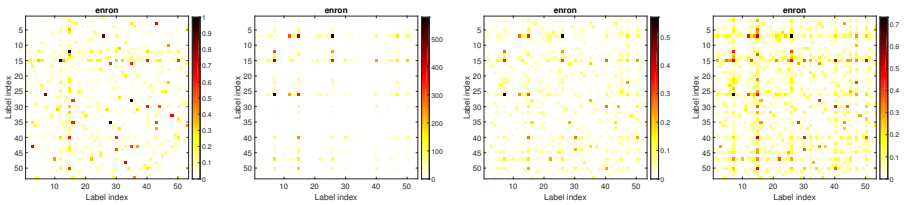
9.2 Contributions

The contributions of this thesis can be cast into five folds:

- In PACC (Chapter 3), we introduce a novel polytree structure for CC to capture label correlations and avoid the problems of error propagation and poorly ordered chain in CC. In addition, a two-stage label-specific feature selection framework is developed which can be directly applied on the existing CC-based methods to improve their performance;

- A unified MLC framework via conditional likelihood maximization is proposed in OCC (Chapter 4). According to the framework, existing CC-based methods and some popular multi-label feature selection approaches can be considered as the special cases of OCC. In addition, several important theorems with corollaries are developed on the basis of the unified framework, which helps us to better understand the experimental phenomenons in previous papers on CC;

- In Chapter 5, it is the first time to introduce the ISD strategy for ML-DR besides the traditional FS-DR and LS-DR. We further separate the existing ISD methods into two categories: label-guided ISD and feature-guided



(a) Normalize mutual information (b) Label kernel (c) Jaccard index (d) Cosine similarity

Figure 9.1: Visualization of label correlations in four criterions on Enron

ISD. Following the latter strategy, CLMLC is proposed in Chapter 8 to group similar instances with similar labels together, and achieves performance superiority compared with the state-of-the-art ML-DR methods;

- Two novel FS-DR methods, MLSF and READER, are proposed in Chapter 6. By $\ell_{2,1}$-norm based empirical risk minimization and manifold learning, READER enables to select features across labels in a robust and semi-supervised way. We show that the optimization problem of READER is equivalent to a generalized eigenvalue problem, and develop an efficient alternating optimization algorithm with convergence property to optimize its objective function.

- A novel LS-DR method, READER-LE, is proposed in Chapter 7 by extending READER. Based on the low rank assumption on the label matrix, most of existing LS-DR methods embed labels in a linear way, which violates various real-world applications. In contrast, READER-LE nonlinearly embeds the label space, saving local information among labels. In addition, it is the first time to conduct LS-DR in a semi-supervised manner, which enables to utilize a large amount of unlabeled data to improve its performance.

- The experimental codes of the proposed methods in this thesis have been released in my GitHub page: https://github.com/futuresun912.

9.3 Future Work

In this thesis, we successfully overcome several limitations in existing MLC and ML-DR methods by proposed methods. However, it is worth noting that there are still some limitations and open problems existed in the proposed methods. Hence, it is necessary to conduct our future research to address these problems, thereby summarizing the future work corresponding to the limitations.

- Precisely modeling label correlations. The current works on MLC attempts to model label correlations from a special point of view, without precisely quantitative definition on label correlations. Fig. 9.1 shows the different patterns of label correlations modeled by four different approaches. It probably misguides the multi-label classifier by modeling insufficient or redundant label correlations, hence reducing the generalization capability. One possible future research direction is to capture label correlations, which helps to directly minimize the empirical risk over the training instances.

- Reducing computational complexity in ML-DR. Although one of the major motivations of ML-DR is to reduce the time cost of MLC on large-scale problems, in practice it typically increases the time complexity in the learning phase, especially for the FS-DR methods. In contrast, LS-DR enables to dramatically decrease the time cost in most cases, but LS-DR always harms the classification performance in several evaluation metrics, like Exact-Match, Macro/Micro-F1, etc. Hence it is necessary to develop a novel ML-DR method to reduce time complexity without performance loss.

- Handling extreme MLC at a tractable resource cost. In recent years, extreme MLC [66, 7, 101, 4] has attracted a lot of attentions from researchers on MLC. However, it is impossible to directly apply the methods proposed in this thesis to handle the extreme MLC problems, where the number of instances, features and labels probably exceeds thousands or even millions. To develop the simple optimization algorithm and utilize distributed computing is a possible solution.

- Semi-supervised MLC. More and more datasets merges as the boom of information technology, which is intractable and too expensive for human beings to annotate each data point with appropriate label subset. Thus, the semi-supervised MLC methods aiming to conduct MLC based on a limited number of labeled instances would help us to fully utilize a huge amount of real-world data. In fact, several semi-supervised MLC methods, such as READER and its extension READER-LE, have been proposed. However, further improvement in terms of time complexity and prediction accuracy is required.

Bibliography

[1] 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 22Nd International Conference on World Wide Web*, pages 13–24, New York, NY, USA, 2013. ACM. 96

[2] A. Alessandro, G. Corani, D. Mauá, and S. Gabaglio. An ensemble of bayesian networks for multilabel classification. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 1220–1225, 2013. 3, 16

[3] K. Aoki and M. Kudo. Decision tree using class-dependent feature subsets. *Structural, Syntactic, and Statistical Pattern Recognition*, 2396:761–769, 2002. 23

[4] R. Babbar and B. Schölkopf. Dismec – distributed sparse machines for extreme multi-label classification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM 2017)*, 2017. 116

[5] Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, 2006. 95

[6] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, 2003. 67

[7] K. Bhatia, H. Jain, P. Kar, M. Varma, and P. Jain. Sparse local embeddings for extreme multi-label classification. In *Advances in NIPS 28*, pages 730–738, 2015. 96, 103, 116

[8] M. Boutell, J. Luo, X. Shen, and C. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004. 2, 3, 9, 16, 17, 48, 50, 81, 84, 102

[9] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, 2011. 67, 69

[10] Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *Journal of Machine Learning Research*, 13:27–66, January 2012. 45

[11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. 81

[12] Xiaojun Chang, Feiping Nie, Yi Yang, and Heng Huang. A convex formulation for semi-supervised multi-label feature selection. In *Proceedings of the 28th AAAI*, pages 1171–1177. 2014. 83

[13] F. Charte, A.J. Rivera, M.J. del Jesus, and F. Herrera. Addressing imbalance in multilabel classification: Measures and random resampling algorithms. *Neurocomputing*, 163:3–16, 2015. 25

[14] Y. Chen and H. Lin. Feature-aware label space dimension reduction for multi-label classification. In *Advances in NIPS 25*, pages 1529–1537. 2012. 61, 64, 74, 91, 102, 103, 106

[15] Weiwei Cheng and Eyke Hüllermeier. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76(2-3):211–225, 2009. 54

[16] C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968. 19, 20

[17] F.D. Comite, R. Gilleron, and M. Tommasi. Learning multi-label alternating decision trees from texts and data. In *Proceedings of the 3rd International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 35–49, 2003. 3, 15, 16

[18] K. Dembczynski, W. Waegeman, W.W. Cheng, and E. Hüllermeier. On label dependence and loss minimization in multi-label classification. *Machine Learning*, 88(1-2):5–45, 2012. 21, 55, 82

[19] 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 Twenty-Seventh International Conference on Machine Learning*, pages 279–286. Omnipress, 2010. 18, 37, 42, 50

[20] Krzysztof Dembczyński, Willem Waegeman, and Eyke Hüllermeier. An analysis of chaining in multi-label classification. In *Proceedings of the 2012 European Conference on Artificial Intelligence*, volume 242, pages 294–299. IOS Press, 2012. 42

[21] J. Demšar. Statistical comparisons of classifiers over multiple sata sets. *The Journal of Machine Learning Research*, 7:1–30, 2006. 32, 34, 54, 107

[22] Kai-Bo Duan and S. Sathiya Keerthi. *Which Is the Best Multiclass SVM Method? An Empirical Study*, pages 278–285. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. 2

[23] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, New York, 2 edition, 2001. 83

[24] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems 14*, pages 681–687, 2001. 3, 15, 16

[25] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. 51, 84

[26] R. M. Fano. Transmission of information: statistical theory of communications. *IEEE Transactions on Information Theory*, 1961. 40

[27] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(7):179–188, 1936. 2

[28] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, November 1997. 44

[29] K. Fukunaga and P. M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE Transactions on Computers*, C-24(7):750–753, July 1975. 2

[30] Johannes Fürnkranz, Eyke Hüllermeier, Eneldo Loza Mencía, and Klaus Brinker. Multilabel classification via calibrated label ranking. *Machine Learning*, 73(2):133–153, November 2008. 3, 16, 70

[31] N. Ghamrawi and A. McCallum. Collective multi-label classification. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 195–200, 2005. 3, 16

[32] Yuhong Guo and Suicheng Gu. Multi-label classification using conditional dependency networks. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, IJCAI'11, pages 1300–1305. AAAI Press, 2011. 3, 12, 16

[33] M.A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 359–366, 2000. 25

[34] Xiaofei He and Partha Niyogi. Locality preserving projections. In *In Advances in Neural Information Processing Systems 16*. MIT Press, 2003. 51, 81

[35] R. Hecht-Nielsen. Theory of the backpropagation neural network. In *International 1989 Joint Conference on Neural Networks*, pages 593–605 vol.1, 1989. 1

[36] David Heckerman, Dan Geiger, and David M. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995. 1

[37] Guang-Bin Huang, Dian Hui Wang, and Yuan Lan. Extreme learning machines: a survey. *International Journal of Machine Learning and Cybernetics*, 2(2):107–122, 2011. 2

[38] Jun Huang, Guorong Li, Qingming Huang, and Xindong Wu. Learning label specific features for multi-label classification. In *2015 IEEE International Conference on Data Mining, 2015*, pages 181–190, 2015. 12, 51, 66, 81

[39] Ling Jian, Jundong Li, Kai Shu, and Huan Liu. Multi-label informed feature selection. In *Proceedings of the 25th IJCAI*, pages 1627–1633, 2016. 83

[40] Asha Gowda Karegowda, AS Manjunath, and MA Jayaram. Comparative study of attribute selection using gain ratio and correlation based feature selection. *International Journal of Information Technology and Knowledge Management*, 2(2):271–277, 2010. 35

[41] Akisato Kimura, Masashi Sugiyama, Takuho Nakano, Hirokazu Kameoka, Hitoshi Sakano, Eisaku Maeda, and Katsuhiko Ishiguro. Semicca: Efficient semi-supervised learning of canonical correlations. *Information and Media Technologies*, 8(2):311–318, 2013. 84

[42] Keigo Kimura, Mineichi Kudo, and Lu Sun. Dimension reduction using nonnegative matrix tri-factorization in multi-label classification. In

Proceedings of the 21st International Conference on Parallel and Distributed Processing Techniques and Applications: Workshop on Mathematical Modeling and Problem Solving, pages 250–255, 2015. 61, 66

[43] Keigo Kimura, Mineichi Kudo, and Lu Sun. Simultaneous nonlinear label-instance embedding for multi-label classification. In *Proceedings of the joint IAPR International Workshops on Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition*, 2016. 61, 91

[44] Keigo Kimura, Mineichi Kudo, Lu Sun, and Sadamori Koujaku. Fast random k-labelsets for large-scale multi-label classification. In *Proceedings of the 23rd International Conference on Pattern Recognition*, 2016. 16

[45] Kenji Kira and Larry A Rendell. The feature selection problem: Traditional methods and a new algorithm. In *AAAI*, volume 2, pages 129–134, 1992. 35

[46] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2):273–324, 1997. 24, 26, 35

[47] M. Kudo and J. Sklansky. Classifier-independent feature selection for two-stage feature selection. *Advances in Pattern Recognition*, 1451:548–554, 1998. 23

[48] S. Kullback and R.A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951. 39

[49] Abhishek Kumar, Shankar Vembu, Aditya Krishna Menon, and Charles Elkan. Learning and inference in probabilistic classifier chains with beam search. In *Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part I*, ECML PKDD'12, pages 665–680, Berlin, Heidelberg, 2012. Springer-Verlag. 50

[50] Ernst Kussul and Tatiana Baidyk. Improved method of handwritten digit recognition tested on {MNIST} database. *Image and Vision Computing*, 22(12):971 – 981, 2004. Proceedings from the 15th International Conference on Vision Interface. 2

[51] J. Langford, T. Zhang, D. Hsu, and S. Kakade. Multi-label prediction via compressed sensing. In *Advances in Neural Information Processing Systems 22*, pages 772–780. 2009. 91

[52] David D. Lewis. Feature selection and feature extraction for text categorization. In *Proceedings of the Workshop on Speech and Natural Language*, pages 212–217, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics. 48

[53] Z. Lin, G. Ding, M. Hu, and J. Wang. Multi-label classification via feature-aware implicit label space encoding. In *Proceedings of the 31st International Conference on Machine Learning*, pages 325–333, 2014. 61, 91

[54] D.C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989. 22

[55] H. Liu and H. Motoda. *Feature selection for knowledge discovery and data mining*. Kluwer Academic Publishers, Norwell, MA, USA, 1998. 24

[56] Z. Ma, F. Nie, Y. Yang, J. R. R. Uijlings, and N. Sebe. Web image annotation via subspace-sparsity collaborated feature selection. *IEEE Trans. on Multimedia*, 14(4):1021–1030, Aug 2012. 84

[57] Z. Ma, F. Nie, Y. Yang, J. R. R. Uijlings, N. Sebe, and A. G. Hauptmann. Discriminating joint feature analysis for multimedia data understanding. *IEEE Trans. on Multimedia*, 14(6):1662–1672, Dec 2012. 84

[58] Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, and SašO Deroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104, September 2012. 54

[59] G. Nasierding, G. Tsoumakas, and A. Kouzani. Clustering based multi-label classification for image annotation and retrieval. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 4514–4519, 2009. 62, 64, 96, 102

[60] Peter Nemenyi. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, New Jersey, USA, 1963. 32

[61] Feiping Nie, Heng Huang, Xiao Cai, and Chris H. Ding. Efficient and robust feature selection via joint l2,1-norms minimization. In *Advances in NIPS 23*, pages 1813–1821. 2010. 72, 83

[62] Batzaya Norov-Erdene, Mineichi Kudo, Lu Sun, and Keigo Kimura. Locality in multi-label classification problems. In *Proceedings of the 23rd International Conference on Pattern Recognition*, 2016. 62, 95

[63] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. 22

[64] Peter Peduzzi, John Concato, Elizabeth Kemper, Theodore R. Holford, and Alvan R. Feinstein. A simulation study of the number of events per variable in logistic regression analysis. *Journal of Clinical Epidemiology*, 49(12):1373 – 1379, 1996. 1

[65] Hanchuan Peng, Fuhui Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, Aug 2005. 46, 48, 56

[66] Y. Prabhu and M. Varma. Fastxml: a fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 263–272, 2014. 96, 103, 116

[67] G.J. Qi, X.S. Hua, Y. Rui, J.H. Tang, T. Mei, and H.J. Zhang. Correlative multi-label video annotation. In *Proceedings of the 15th International Conference on Multimedia*, pages 17–26, 2007. 3, 16

[68] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. 1

[69] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359, 2011. 2, 4, 12, 16, 17, 37, 48, 50, 51, 52, 67, 70, 101, 102, 103

[70] Jesse Read, Luca Martino, and David Luengo. Efficient monte carlo methods for multi-dimensional learning with classifier chains. *Pattern Recognition*, 47(3):1535 – 1546, 2014. Handwriting Recognition and other {PR} Applications. 50

[71] Jesse Read, Peter Reutemann, Bernhard Pfahringer, and Geoff Holmes. MEKA: A multi-label/multi-target extension to Weka. *Journal of Machine Learning Research*, 17(21):1–5, 2016. 9, 28

[72] G. Rebane and J. Pearl. The recovery of causal polytrees from statistical data. In *Proceedings of the 3rd Conference on Uncertainty in Artificial Intelligence*, pages 222–228, 1987. 19

[73] E.S. Robert and Y. Singer. Boostexter: a boosting-based system for text categorization. *Machine Learning*, 39:135–168, 2000. 3, 16

[74] Juho Rousu, Craig Saunders, Sandor Szedmak, and John Shawe-Taylor. Learning hierarchical multi-category text classification models. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 744–751, 2005. 95

[75] D. W. Ruck, S. K. Rogers, M. Kabrisky, M. E. Oxley, and B. W. Suter. The multilayer perceptron as an approximation to a bayes optimal discriminant function. *IEEE Transactions on Neural Networks*, 1(4):296–298, Dec 1990. 2

[76] Karl-Michael Schneider. A comparison of event models for naive bayes anti-spam e-mail filtering. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 1*, EACL '03, pages 307–314, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. 2

[77] Konstantinos Sechidis, Nikolaos Nikolaou, and Gavin Brown. *Information theoretic feature selection in multi-label data through composite likelihood*, volume 8621, pages 143–152. 8 2014. 48, 56

[78] L. Sun, B. Ceran, and J. Ye. A scalable two-stage approach for a class of dimensionality reduction techniques. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 313–322, 2010. 98

[79] L. Sun, S. Ji, and J. Ye. Hypergraph spectral learning for multi-label classification. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 668–676, 2008. 66, 102

[80] L. Sun, S. Ji, and J. Ye. Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):194–200, 2011. 51, 61, 62, 66

[81] L. Sun and M. Kudo. Polytree-augmented classifier chains for multi-label classification. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 3834–3840, 2015. 4, 16, 19, 21, 41

[82] Lu Sun and Mineichi Kudo. Multi-label classification by polytree-augmented classifier chains with label-dependent features. *Pattern Analysis and Applications*, June 2016. under review. 4

[83] Lu Sun and Mineichi Kudo. Optimization of classifier chains via conditional likelihood maximization. *Pattern Recognition*, June 2016. under review. 5, 37

[84] Lu Sun, Mineichi Kudo, and Keigo Kimura. Multi-label classification with meta-label-specific features. In *Proceedings of the 23rd International Conference on Pattern Recognition*, 2016. 5, 61, 66

[85] Lu Sun, Mineichi Kudo, and Keigo Kimura. A scalable clustering-based local multi-label classification method. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, pages 261–268, 2016. 3, 5, 62, 96

[86] Lu Sun, Mineichi Kudo, and Keigo Kimura. Reader: Robust semi-supervised multi-label dimension reduction. *IEICE Transactions on Information and Systems*, October 2017. 5, 71

[87] J.A.K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3):293–300, 1999. 1

[88] Farbound Tai and Hsuan-Tien Lin. Multilabel classification with principal label space transformation. *Neural Computing*, 24(9):2508–2542, September 2012. 61, 64

[89] L. Tang, S. Rajan, and V.K. Narayanan. Large scale multi-label classification via metalabeler. In *Proceedings of the 18th International Conference on World Wide Web*, pages 211–220, 2009. 12

[90] Robert Tibshirani. Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(3):273–282, 2011. 67, 69

[91] J.T. Tomás, N. Spolaôr, E.A. Cherman, and M.C. Monard. A framework to generate synthetic multi-label datasets. *Electronic Notes in Theoretical Computer Science*, 302:155–176, 2014. 28

[92] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3:1–13, 2007. 2, 3, 11, 15

[93] G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proceedings of ECML/PKDD 2008 Workshop on Mining Multidimensional Data*, 2008. 30, 31, 62, 65, 95

[94] G. Tsoumakas, I. Katakis, and L. Vlahavas. Random k-labelsets for multilabel classification. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1079–1089, 2011. 3, 16, 30, 70

[95] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas. Mulan: A java library for multi-label learning. *Journal of Machine Learning Research*, 12:2411–2414, 2011. 9, 28, 83

[96] Celine Vens, Jan Struyf, Leander Schietgat, Sašo Džeroski, and Hendrik Blockeel. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185–214, 2008. 95

[97] H. Wang, C. Ding, and H. Huang. Multi-label linear discriminant analysis. In *Proceedings of the 11th European Conference on Computer Vision*, volume 6316, pages 126–139. 2010. 61, 63, 66

[98] D. Watkins. *Chemometrics, mathematics and statistics in chemistry*. Reidel Publishing Company, Dordrecht, N etherlands, 1984. 98

[99] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 2764–2770, 2011. 91

[100] K. Worsley, J. Poline, K. Friston, and A. Evans. Characterizing the response of PET and fMRI data using multivariate linear models. *Neuroimage*, 6(4):305–319, 1997. 98

[101] Chang Xu, Dacheng Tao, and Chao Xu. Robust extreme multi-label learning. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1275–1284, New York, NY, USA, 2016. ACM. 116

[102] Howard Hua Yang and John Moody. Data visualization and feature selection: New algorithms for nongaussian data. In *in Advances in Neural Information Processing Systems*, pages 687–693. MIT Press, 1999. 48

[103] Y. Yang and J.O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420, 1997. 24, 35

[104] H. Yu, P. Jain, P. Kar, and S. Dhillon. Large-scale multi-label learning with missing labels. In *Proceedings of the 31st International Conference on Machine Learning*, pages 593–601, 2014. 91

[105] Kai Yu, Shipeng Yu, and Volker Tresp. Multi-label informed latent semantic indexing. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 258–265, 2005. 9, 51, 66, 81

[106] L. Yu and H. Liu. Feature selection for high-dimensional data: a fast correlation-based filter solution. In *Proceedings of the 20th International Conference on Machine Learning*, pages 856–863, 2003. 24

[107] J.H. Zaragoza, L.E. Sucar, E.F. Morales, C. Bielza, and P. Larra naga. Bayesian chain classifiers for multidimensional classification. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 2192–2197, 2011. 18, 21, 37, 48, 51

[108] M. Zhang and L. Wu. Lift: multi-label learning with label-specific features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1):107–120, 2015. 51, 66, 81

[109] M. L. Zhang and Z. H. Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, Aug 2014. 2

[110] M.L. Zhang and K. Zhang. Multi-label learning by exploiting label dependency. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 999–1008, 2010. 21

[111] M.L. Zhang and Z.H. Zhou. Multi-label neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 18:1338–1351, 2006. 3, 15, 16

[112] M.L. Zhang and Z.H. Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40:2038–2048, 2007. 3, 9, 15, 16, 30

[113] Yin Zhang and Zhi-Hua Zhou. Multilabel dimensionality reduction via dependence maximization. *ACM Transactions on Knowledge Discovery from Data*, 4(3):14:1–14:21, October 2010. 51, 66, 81

[114] Jidong Zhao, Ke Lu, and Xiaofei He. Locality sensitive semi-supervised feature selection. *Neurocomputing*, 71(10–12):1842–1849, 2008. 84