Advanced Topics in Weakly Supervised Learning

Lei Feng

School of Computer Science and Engineering

A thesis submitted to Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy (Ph.D.)

2021

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

August 15, 2020

.....
Date

Lei FENG

.....
Lei Feng

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

August 15, 2020

.....
Date

.....
Bo An

Authorship Attribution Statement

This thesis contains material from four papers published in the following peer-reviewed conferences in which I am the first and corresponding author.

Chapter 3 is published as **Lei Feng**, Takuo Kaneko, Bo Han, Gang Niu, Bo An, and Masashi Sugiyama, "Learning with Multiple Complementary Labels". *Proceedings of the 37th International Conference on Machine Learning* (**ICML**), pp.3072-3081, 2020. The contributions of the authors are as follows:

- Takuo Kaneko and I proposed the problem setting.
- I proposed improved solutions, conducted the experiments, and wrote the main manuscript. Takuo Kaneko wrote a part of the manuscript.
- Dr. Gang Niu discussed with me about how to properly organize the manuscript and rewrote the abstract.
- Dr. Bo Han, Dr. Gang Niu, Prof. Bo An, and Prof. Masashi Sugiyama provided insightful comments and carefully revised the manuscript.

Chapter 4 is published as **Lei Feng** and Bo An, "Leveraging Latent Label Distributions for Partial Label Learning". *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (**IJCAI**), pp.2107-2113, 2018. The contributions of the authors are as follows:

- I proposed the idea and the solution.
- I proposed conducted the experiments, and wrote the manuscript. Prof. Bo An discussed with me about how to properly present the key idea of the manuscript.
- Prof. Bo An provided insightful comments and revised the manuscript.

Chapter 5 is published as **Lei Feng** and Bo An, "Partial Label Learning with Self-Guided Retraining". *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (**AAAI**), pp.3542-3549, 2019. The contributions of the authors are as follows:

- I proposed the idea and the solution.
- I proposed conducted the experiments, and wrote the manuscript. Prof. Bo An discussed with me about how to properly present the key idea of the manuscript.
- Prof. Bo An provided critical comments and revised the manuscript.

Chapter 6 is published as **Lei Feng**, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, and Masashi Sugiyama, "Provably Consistent Partial-Label Learning". *Proceedings of the 34th Conference on Neural Information Processing Systems* (**NeurIPS**), accepted, 2020. The contributions of the co-authors are as follows:

- I proposed the idea and the solution.
- Dr. Gang Niu provided critical comments for the methods and I justified the methods;
- I conducted the experiments and wrote the manuscript.
- Jiaqi Lv helped me conduct some experiments.
- Dr. Bo Han, Dr. Miao Xu, and Dr. Gang Niu provided constructive comments and revised the manuscript.
- Prof. Xin Geng, Prof. Bo An, and Prof. Sugiyama provided insightful comments and I revised the manuscript according to their comments.

August 15, 2020 Lei FENG

....

Date Lei Feng

Abstract

Machine learning has achieved great advances in various tasks, especially in supervised learning tasks. However, supervised learning requires all the correct labels of training examples to train an effective model, and collecting training examples with such strong supervision could incur unaffordable monetary or time cost. Therefore, weakly supervised learning, which aims to build predictive models by learning with weak supervision, has attracted increasing attention in recent years.

This doctoral thesis is devoted to investigating some advanced topics in weakly supervised learning, including *complementary-label learning* and *partial-label learning*.

Complementary-label learning solves the problem where each training example is supplied with a single *complementary label* (CL), which only specifies one of the classes that the example does not belong to. Although existing complementary-label learning approaches have provided solid theoretical foundations and achieved promising performance, they are all restricted to the case where each example is associated with a single CL. This case notably limits its potential since our labelers may easily identify *multiple complementary labels* (MCLs) to one example. To address this problem, we propose an extended problem setting to allow MCLs for each example and two ways for learning with MCLs. In the first way, we design two wrappers that decompose MCLs into many single CLs in different manners, so that we could use any method for learning with CLs. However, we find that the supervision information that MCLs hold is conceptually diluted after decomposition. Thus, in the second way, we derive an *unbiased risk estimator*; minimizing it processes each set of MCLs as a whole and possesses an estimation error bound. In addition, we improve the second way into minimizing properly chosen upper bounds for practical implementation. Experiments show that the former way works well for learning with MCLs while the latter is even better.

Partial-label learning solves the problem where each training example is supplied with a set of candidate labels, only one of which is the correct label. By regarding each CL as a non-candidate label, complementary-label learning can be also considered as a special case of partial-label learning. Hence we focus more on partial-label learning in this thesis. We propose several effective methods to alleviate various issues in partial-label learning.

Firstly, most of the existing methods fail to take into account the fact that different candidate labels give different contributions to model training. We formalize the probabilities of different candidate labels being the correct label as latent label distributions, and then propose a novel unified formulation to estimate the latent label distributions while training the model simultaneously.

Secondly, *self-training* is a representative semi-supervised learning strategy that can directly label an unlabeled instance with enough high confidence, while the incorrectly labeled data could have contagiously negative impacts on the final predictions. Therefore, it is still unclear whether the idea of self-training can be used to improve the practical performance of partial-label learning. We provide the first attempt to improve self-training for partial-label learning by presenting a unified formulation with proper constraints to jointly train the desired model and perform pseudo-labeling.

Thirdly, there still lacks a theoretical understanding of the consistency of partial-label learning methods through the lens of the generation process of partially labeled data. We propose the first generation model of partially labeled data, and develop two novel partial-label learning methods that are guaranteed to be provably consistent, i.e., one is *risk-consistent* and the other is *classifier-consistent*.

Extensive experimental results clearly demonstrate the effectiveness of the above our proposed partial-label learning methods.

Acknowledgments

I would like to express tremendous gratitude to people who helped and inspired me during the past three years.

First of all, I am deeply indebted to my supervisor Prof. Bo An. I really appreciate that he provided me the opportunity to pursue my Ph.D. degree under his supervision, which became the change point to my life. He provided an extremely suitable environment for my study and research, and he gave me the greatest freedom to do whatever I wanted to do. He is always patient, polite, gentle, and warmhearted. He has not only taught me how to be professional and responsible and how to properly communicate with others, but also supported and helped me on many things. My deepest gratitude to him is beyond words. I feel very lucky to be his Ph.D. student. I am also eager to be a nice professor like him.

I also feel lucky to share my Ph.D. experience with a group of talented people: Qingyu Guo, Jiarui Gan, Wanyuan Wang, Haipeng Chen, Yanhai Xiong, Jiuchuan Jiang, Youzhi Zhang, Xinrun Wang, Xu He, Aye Phyu, Hongxin Wei, Rundong Wang, Wei Qiu, Yakub Cerny, Runsheng Yu, Yanchen Deng, Shuxin Li, Wanqi Xue, Fei Fei Lin, Pengdeng Li, and Shenggong Ji. Thank you all for those moments of laughs, happiness, and excitement, and your generous help when I needed it. I will never forget the moments for fighting together for deadlines, having internal group meetings, going hiking, and enjoying the group dinner with you.

I would like to thank my Thesis Advisory Committee (TAC) members: Prof. Sinno Jialin Pan and Prof. Xiaohui Bei. They helped me a lot and made my research journey easier in NTU.

I would like to thank these nice people I have met during my internship at RIKEN: Dr. Gang Niu, Prof. Masashi Sugiyama, Dr. Bo Han, Dr. Miao Xu, Jiaqi Lv, Jingfeng Zhang,

Yuting Chou, Chiayou Chen, Nan Lu, Han Bao, Yuko Kuroki, Takuo Kaneko, Nontawat Charoenphakdee, Nana Sekine. Thank you for your guidance/discussion/help/accompany when I was alone in Tokyo.

I would like to thank many of my friends: Senlin Shu, Lingtao Peng, Jie He, Yi Liu, Shuo He, Zhuoyi Lin, Gengyu Lyu, Haobo Wang, Lue Tao, Zhuowei Wang, and so on. Thanks so much to my girlfriend Wenzhuo Yang for accompanying me and giving me much encouragement and patience in these three years.

Most importantly, I would like to thank my families for their unconditional love and support to me. I love you and miss you so much.

Contents

Abstract v
Acknowledgments vii
List of Figures xiv
List of Tables xvii
List of Publications xviii

1 Introduction **1**
1.1 Background 1
1.2 Limitations and Motivations 2
1.3 Major Contributions 3
1.4 Thesis Organization 4

2 Preliminaries **5**
2.1 Supervised Learning 5
2.1.1 Binary Classification 6
2.1.2 Multi-Class Classification 7
2.2 Advanced Topics in Weakly Supervised Learning 8
2.2.1 Complementary-Label Learning 8
2.2.2 Partial-Label Learning 10

3 Learning with Multiple Complementary Labels **13**
3.1 Motivation 13
3.2 Multiple Complementary Labels 14
3.2.1 Data Generation Process 14
3.2.2 Motivation for Data Generation 16

3.3 Learning with Multiple Complementary Labels 19
3.3.1 Wrappers 19
3.3.2 Unbiased Risk Estimator 21
3.3.3 Practical Implementation 26
3.4 Experiments 29
3.4.1 Datasets 29
3.4.2 Experimental Results 33
3.5 Chapter Summary 37

4 Partial-Label Learning with Latent Label Distributions 39
4.1 Motivation 39
4.2 The LALO Method 40
4.3 Alternating Optimization 42
4.3.1 Updating Model Parameters 42
4.3.2 Updating Latent Label Distributions 44
4.4 Experiments 46
4.4.1 Experimental Setup 46
4.4.2 Experimental Results 48
4.4.3 Sensitivity Analysis 50
4.5 Chapter Summary 51

5 Partial-Label Learning with Self-Guided Retraining 52
5.1 Introduction 52
5.2 The Proposed Approach 53
5.3 Optimization 55
5.3.1 Updating Model Parameters 55
5.3.2 Updating Labeling Confidences 56
5.4 Experiments 59
5.4.1 Compared Methods 59
5.4.2 Controlled Datasets 60
5.4.3 Real-World Datasets 63
5.4.4 Further Analysis 64
5.5 Chapter Summary 65

6 Partial-Label Learning with Provably Consistency **67**
6.1 Introduction . . . 67
6.2 Related Formulations . . . 68
6.2.1 Learning with Ordinary Labels . . . 68
6.2.2 Learning with Partial Labels . . . 69
6.2.3 Learning with Multiple Complementary Labels . . . 70
6.3 Data Generation Model . . . 71
6.3.1 Partially Labeled Data Distribution . . . 71
6.3.2 Motivation for Data Generation . . . 73
6.4 Consistent Methods . . . 75
6.4.1 Risk-Consistent Method . . . 76
6.4.2 Classifier-Consistent Method . . . 77
6.5 Experiments . . . 82
6.5.1 Experimental Setup . . . 83
6.5.2 Experimental Results . . . 87
6.5.3 Effectiveness of Generation Model . . . 89
6.6 Chapter Summary . . . 94

7 Conclusions and Future Directions **95**
7.1 Conclusions . . . 95
7.2 Future Directions . . . 97
7.2.1 Online Weakly Supervised Learning . . . 97
7.2.2 Weakly Supervised Metric Learning . . . 97
7.2.3 Weakly Supervised Domain Adaptation . . . 97

Appendices **100**

A Mathematical Proofs in Chapter 3 **100**
A.1 Proof of Theorem 3.4 . . . 100
A.2 Derivations and Boundness of the Used Loss Functions . . . 105
A.2.1 Derivations of the Used Loss Functions . . . 105
A.2.2 Boundness of the Used Loss Functions . . . 105

B Mathematical Proofs in Chapter 6 **107**

B.1 Proof of Theorem 6.4 107

B.2 Proof of Theorem 6.6 109

References **111**

List of Figures

3.1 Mean test accuracy with standard deviation on different datasets with different loss functions. 30
3.2 Mean training accuracy with standard deviation on different datasets with different loss functions. 31

4.1 Classification accuracy on controlled datasets with ϵ ranging from 0.1 to 0.7 ($p = 1, r = 1$). 47
4.2 Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 1$). 48
4.3 Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 2$). 48
4.4 Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 3$). 49
4.5 Parameter sensitivity analysis of LALO on the real-world datasets Lost and MSRCv2. 50

5.1 Classification accuracy on controlled datasets with ϵ ranging from 0.1 to 0.7 ($p = 1, r = 1$). 61
5.2 Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 1$). 62
5.3 Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 2$). 62
5.4 Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 3$). 62
5.5 Parameter sensitivity and convergence analysis of SURE. 65

6.1 Classification accuracy (mean test accuracy with standard deviation of 5 trials) of different methods on different datasets. 89
6.2 Classification accuracy (mean transductive accuracy with standard deviation of 5 trials) of different methods on different datasets. 90
6.3 Heatmaps of different generation processes of candidate label sets. 91

List of Tables

3.1 Two ways for handling a set of MCLs (with size s). 20

3.2 List of the used bounded and unbounded losses, which can be directly inserted to our (unbiased) empirical risk estimator Eq. (3.11). 27

3.3 Summary of the used benchmark datasets with the corresponding models. 29

3.4 Classification accuracy (mean±std of 5 trials) of each method on the four UCI datasets using a linear model. The best performance among all the methods is highlighted in bold. Besides, •/∘ denotes whether the performance of our proposed method (the best of EXP and LOG) is statistically (paired t-test at 0.05 significance level) better/worse than the compared method. 33

3.5 Classification accuracy (mean±std of 5 trials) of each method on the four benchmark datasets using a linear model. The best performance among all the methods is highlighted in bold. Besides, •/∘ denotes whether the performance of our proposed method (the best of EXP and LOG) is statistically (paired t-test at 0.05 significance level) better/worse than the compared method. 34

3.6 Classification accuracy (mean±std of 5 trials) of each method on the five benchmark datasets using neural networks. The best performance among all the methods is highlighted in bold. Besides, •/∘ denotes whether the performance of our proposed method (the best of EXP and LOG) is statistically (paired t-test at 0.05 significance level) better/worse than the compared method. 35

3.7 Classification accuracy (mean±std% of five trials) of each method on Kuzushiji-MNIST using a linear model. The best performance among all the methods is highlighted in bold. 36

3.8 Classification classification (mean±std% of 5 trials) of each method on Kuzushiji-MNIST using a MLP model. The best performance among all the methods is highlighted in bold. 37

4.1 Characteristics of the used datasets. 46
4.2 Win/tie/loss counts on the controlled datasets between LALO and the compared algorithms. 49
4.3 Classification accuracy of each algorithm on the used real-world partially labeled datasets. Furthermore, •/∘ indicates whether LALO is statistically superior/inferior to the compared algorithm. 50

5.1 Characteristics of the controlled datasets. 61
5.2 Win/tie/loss (t-test at 0.05 significance level for two independent samples) counts on the controlled datasets between SURE and the compared algorithms. 63
5.3 Characteristics of real-world partially labeled datasets. 63
5.4 Classification accuracy of each algorithm on the real-world datasets. Furthermore, •/∘ indicates whether SURE is statistically superior/inferior to the compared algorithm (t-test at 0.05 significance level for two independent samples). 63

6.1 Characteristics of the controlled datasets. 83
6.2 Characteristics of the real-world partially labeled datasets. 83
6.3 Test accuracy (mean±std) of each method using neural networks on benchmark datasets. ResNet is trained on CIFAR-10, and MLP is trained on the other three datasets. 84
6.4 Test accuracy (mean±std) of each method using neural networks on benchmark datasets. DenseNet is trained on CIFAR-10, and LeNet is trained on the other three datasets. 84
6.5 Transductive accuracy of each method using neural networks on benchmark datasets. ResNet is trained on CIFAR-10, and MLP is trained on the other three datasets. 85

6.6 Transductive accuracy of each method using neural networks on benchmark datasets. DenseNet is trained on CIFAR-10, and LeNet is trained on the other three datasets. 86
6.7 Test accuracy (mean±std) of each method using linear model on UCI datasets. 87
6.8 Test accuracy (mean±std) of each method using linear model on real-world datasets. 87
6.9 Test accuracy (mean±std%) of the RC method using neural networks on benchmark datasets with different generation models. The best performance is highlighted in bold. 90
6.10 Test accuracy (mean±std%) of the CC method using neural networks on benchmark datasets with different generation models. The best performance is highlighted in bold. 92
6.11 Test accuracy (mean±std) of each method using neural networks on benchmark datasets. DenseNet is trained on CIFAR-10, and LeNet is trained on the other three datasets. Candidate label sets are generated by the generation model in Case 1 (entropy=2.015). 93

List of Publications

International Conferences

1. Hongxin Wei, **Lei Feng**, Rundong Wang, Bo An. MetaInfoNet: Learning Task-Guided Information for Sample Reweighting. *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (**CVPR'21**), submitted.

2. Zhuowei Wang, Jing Jiang, Bo Han, **Lei Feng**, Bo An, Gang Niu, Guodong Long. SemiNLL: A Framework of Noisy-Label Learning by Semi-Supervised Learning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (**CVPR'21**), submitted.

3. **Lei Feng**, Senlin Shu, Nan Lu, Miao Xu, Bo Han, Gang Niu, Bo An, Masashi Sugiyama. Pointwise Binary Classification with Pairwise Confidence Comparisons. *International Conference on Learning Representations* (**ICLR'21**), submitted.

4. **Lei Feng**, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, Masashi Sugiyama. Provably Consistent Partial-Label Learning. *Proceedings of the 34th Conference on Neural Information Processing Systems* (**NeurIPS'20**), accepted.

5. **Lei Feng**, Takuo Kaneko, Bo Han, Gang Niu, Bo An, Masashi Sugiyama. Learning with Multiple Complementary Labels. *Proceedings of the 37th International Conference on Machine Learning* (**ICML'20**), pp.3072-3081.

6. Jiaqi Lv, Miao Xu, **Lei Feng**, Gang Niu, Xin Geng, Masashi Sugiyama. Progressive Identification of True Labels for Partial-Label Learning. *Proceedings of the 37th International Conference on Machine Learning* (**ICML'20**), pp.6500-6510.

7. Hongxin Wei, **Lei Feng**, Xiangyu Chen, Bo An. Combating Noisy Labels by Agreement: A Joint Training Method with Co-Regularization. *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (**CVPR'20**), pp.13726-13735.

8. **Lei Feng**, Senlin Shu, Zhuoyi Lin, Fengmao Lv, Li Li, Bo An. Can Cross Entropy Loss be Robust to Label Noise? *Proceedings of the 29th International Joint Conference on Artificial Intelligence* (**IJCAI'20**), pp.2206-2212.

9. Jun Huang, Linchuan Xu, Jing Wang, **Lei Feng**, Kenji Yamanishi. Discovering Latent Class Labels for Multi-Label Learning. *Proceedings of the 29th International Joint Conference on Artificial Intelligence* (**IJCAI'20**), pp.3058-3064.

10. **Lei Feng**, Bo An. Partial Label Learning by Semantic Difference Maximization. *Proceedings of the 28th International Joint Conference on Artificial Intelligence* (**IJCAI'19**), pp.2294-2300.

11. **Lei Feng**, Bo An, Shuo He. Collaboration based Multi-Label Learning. *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (**AAAI'19**), pp.3550-3557.

12. **Lei Feng**, Bo An. Partial Label Learning with Self-Guided Retraining. *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (**AAAI'19**), pp.3542-3549.

13. Shuo He, **Lei Feng**, Li Li. Estimating Latent Relative Labeling Importances for Multi-Label Learning. *Proceedings of the IEEE International Conference on Data Mining* (**ICDM'18**), pp.1013-1018.

14. **Lei Feng**, Bo An. Leveraging Latent Label Distributions for Partial Label Learning. *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (**IJCAI'18**), pp.2107-2113.

International Journals

1. **Lei Feng**, Jun Huang, Senlin Shu, Bo An. Regularized Matrix Factorization for Multi-Label Learning with Missing Labels. *IEEE Transactions on Cybernetics*, accepted, 2020.

2. **Lei Feng**, Hongxin Wei, Qingyu Guo, Zhuoyi Lin, Bo An. Embedding-Augmented Generalized Matrix Factorization for Recommendation with Implicit Feedback. *IEEE Intelligent Systems*, accepted, 2020.

3. Yan Yan, Shining Li, **Lei Feng**. Partial Multi-Label Learning with Mutual Teaching. *Knowledge-Based Systems*, accepted, 2020.

4. Jianyang Zhang, Fengmao Lv, Guowu Yang, **Lei Feng**, Yufeng Yu, Lixin Duan. Learning Cross-Domain Semantic-Visual Relation for Transductive Zero-Shot Learning. *IEEE Transactions on Cybernetics*, under review, 2020.

5. Hongxin Wei, Renchunzi Xie, **Lei Feng**, Bo Han, Bo An. Deep Learning from Multiple Noisy Annotators as A Union. *IEEE Transactions on Neural Networks and Learning Systems*, under review, 2020.

6. Zhuoyi Lin, **Lei Feng**, Rui Yin, Xingzhi Guo, Chi Xu, Chee-Keong Kwoh. COMET: Convolutional Dimension Interaction for Deep Matrix Factorization. *IEEE Transactions on Cybernetics*, under review, 2020.

7. Zhuoyi Lin, **Lei Feng**, Rui Yin, Chi Xu, and Chee-Keong Kwoh. GLIMG: Global and Local Item Graphs for Recommender Systems. *Information Science*, under review, 2020.

8. Senlin Shu, Fengmao Lv, **Lei Feng**, Yan Yan, Li Li, Shuo He, Jun He. Incorporating Multiple Cluster Centers for Multi-Label Learning. *Information Science*, under review, 2020.

Chapter 1

Introduction

1.1 Background

Machine learning [1] aims to automatically detect meaningful patterns in data. It has become a widely used tool in many real-world tasks that require information extraction from large-scale datasets in the past couple of decades. One of the important machine learning tasks is *supervised learning*, which aims to learn a mapping function using the given training data consisting of a set of training examples (input-output pairs). In supervised learning, each training example is composed of a feature vector (*instance*) describing the input/object, and a *label* indicating the ground-truth output. In terms of the meaning of the label, there are two typical tasks in supervised learning, including *classification* and *regression*. In classification, the label specifies the class that the instance belongs to. In regression, the label is a real-value response corresponding to the instance. In this thesis, we focus on classification.

Most of the successful machine learning techniques, such as deep learning [2–4], are demanding for ground-truth labels that are correctly assigned to the training data set. However, in many real-world scenarios, it would be difficult to collect training data with strong supervision information, due to the high cost of the data-labeling process [5]. Therefore, *weakly supervised learning*, which aims to learn with weak supervision, has attracted increasing attention in recent years. To this end, various weakly supervised learning frameworks have been extensively investigated, including *semi-supervised learning* [6–12], *positive-unlabeled learning* [13–21], *noisy-label learning* [22–29], *positive-confidence learning* [30, 31], *similar-unlabeled learning* [32], and *unlabeled-unlabeled learning* [33, 34].

In this thesis, we focus on two advanced weakly supervised learning frameworks including *complementary-label learning* (CLL) and *partial-label learning* (PLL). CLL solves the problem where each training example is provided with a single complementary label, which only specifies an incorrect class of this example. PLL solves the problem where each training example is supplied with a set of candidate labels, only one of which is the correct label. By regarding complementary labels as non-candidate labels, CLL could be considered as an extreme case of PLL. Compared with traditional multi-class classification, it would be much easier to collect training data with weak supervision for CLL or PLL. For instance, there may exist private questions in some survey data [35–37]. In this case, it would be hard for us to easily get the correct answer (label) to the private question. Fortunately, it could be mentally less demanding if the respondent is asked to give additional information instead of the correct answer. Concretely, if we could ask the respondent to give an incorrect answer to a question, then we obtain a complementary label and we can apply CLL. If we ask the respondent to provide multiple possible answers, among which only one is the correct answer, then we obtain a set of candidate labels and we can apply PLL. There are also many real-world applications for PLL or CLL, such as image annotation [38,39], web mining [40], ecoinformatics [41], and medical image segmentation [42].

1.2 Limitations and Motivations

As described above, CLL and PLL are important weakly supervised learning frameworks that have many important applications in real-world problems. Therefore, it is of great importance to develop effective methods to improve the practical performance of CLL and PLL and provide deeper theoretical analyses to advance the understandings of them. Although a number of CLL methods and PLL methods have been proposed, they still have some limitations and there are also some issues that remain unsolved. Specifically, in the previous framework of CLL, the problem setting only allows a single complementary label for each training example. Such setting critically limits the potential of CLL, since the labelers could easily specify multiple complementary labels to a given example in real-world scenarios. In other words, the data distribution of assigning multiple

complementary labels to an example is not explicitly formulated. For PLL, there are several unsolved issues. Firstly, most of the existing methods either focus too much on a single candidate label or treat each candidate label equally. They fail to reflect that different candidate labels give different contributions to model training. Secondly, *self-training* is a representative semi-supervised learning strategy that has both advantages and disadvantages, while it is still unclear whether the idea of self-training can be used to improve the practical performance of PLL. It is expected that by adapting self-training to PLL, we could derive a promising PLL method that has great performance, which would further advance the state of the art of PLL. Thirdly, there still lacks a theoretical understanding of the consistency of PLL methods through the lens of the generation process of partially labeled data. Having an explicit data distribution not only helps us to understand how partially labeled examples are generated, but also enables us to perform effective empirical risk minimization.

This thesis aims to systematically investigate how to effectively solve the aforementioned issues.

1.3 Major Contributions

The major contributions of this thesis are summarized below.

- We propose an extended problem setting to allow multiple complementary labels for each example and two ways for learning with MCLs. In the first way, we design two wrappers that decompose MCLs into many single CLs in different manners, so that we could use any method for learning with CLs. In the second way, we derive an unbiased risk estimator, and improve it into minimizing properly chosen upper bounds for practical implementation. Experiments show that the former way works well for learning with MCLs while the latter is even better.

- We formalize the probabilities of different candidate labels being the correct label as latent label distributions, and propose a novel unified formulation (called LALO) to estimate the latent label distributions while training the model simultaneously. Experimental results demonstrate the effectiveness of LALO.

- We provide the first attempt to improve self-training for partial-label learning by presenting a novel unified formulation (called SURE) with the maximum infinity norm regularization to train an effective model in conjunction with pseudo-labeling. Experimental results demonstrate the effectiveness of SURE.

- We propose the first generation model of partially labeled data, and develop two novel partial-label learning methods that are guaranteed to be provably consistent, i.e., one is risk-consistent and the other is classifier-consistent. Our methods are advantageous, since they are compatible with any deep network or stochastic optimizer. Experiments validate the effectiveness of the proposed consistent methods.

1.4 Thesis Organization

The remaining part of this thesis is organized as follows. Chapter 2 introduces preliminary knowledge and related work of this thesis. Chapter 3 extends the problem setting of CLL to allow multiple complementary labels for each example and presents two ways for learning with MCLs. Chapter 4 presents a novel unified PLL formulation (called LALO) to estimate the latent label distributions while training the model simultaneously. Chapter 5 explores the idea of self-training to improve the practical performance of PLL. Chapter 6 presents a data generation model and two effective methods for PLL. Chapter 7 concludes the thesis and discusses possible directions of future work.

Chapter 2

Preliminaries

2.1 Supervised Learning

In machine learning, it is common that collected data are defined as examples sampled from an unknown distribution, and we usually call it true distribution. Besides, in traditional supervised learning, training data and test data are assumed to be sampled from the true distribution, and we aim at training an effective classifier based on the collected training data for correctly classifying the unseen test data. Of course, there are many real-world scenarios with biased training data [43–45] (i.e., the distribution of obtained training data inconsistent with the true distribution). A number of studies considered learning with biased data, such as covariate shift [46–49], domain adaptation [50–56], and class imbalance [57–60]. In this thesis, we focus on the traditional situation where the training data and test data are both sampled from the true distribution.

Let us formulate the multi-class classification problem. Let the feature space be $\mathcal{X} \in \mathbb{R}^d$ and the label space be $\mathcal{Y} = [k]$ (with k classes) where $[k] := \{1, \ldots, k\}$. Let us clearly define that $\boldsymbol{x}$ denotes an instance and $(\boldsymbol{x}, y)$ denotes an example including an instance $\boldsymbol{x}$ and a label y. When ordinary labels are provided, we usually assume each example $(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}$ is independently sampled from an unknown data distribution with probability density $p(\boldsymbol{x}, y)$. Then, the goal of multi-class classification is to induce a multi-class classifier $f : \mathcal{X} \to \mathbb{R}^k$ that minimizes the following classification risk:

$$R(f) = \mathbb{E}_{p(\boldsymbol{x},y)}\Big[\mathcal{L}(f(\boldsymbol{x}), y)\Big], \tag{2.1}$$

where $\mathbb{E}_{p(\boldsymbol{x},y)}[\cdot]$ denotes the expectation over the joint probability density $p(\boldsymbol{x}, y)$ and $\mathcal{L} : \mathbb{R}^k \times \mathcal{Y} \to \mathbb{R}_+$ is a multi-class loss function that measures how well a classifier estimates a given label. We say that a method is *classifier-consistent* if the learned classifier by the method is infinite-sample consistent to $\arg\min_{f \in \mathcal{F}} R(f)$, and a method is *risk-consistent* if the method possesses a classification risk estimator that is equivalent to $R(f)$ given the same classifier f. It is worth noting that a risk-consistent method is also classifier-consistent [27]. However, a classifier-consistent method may not be risk-consistent.

In general, we do not know the underlying data distribution, while we can approximate the classification risk in Eq. (2.1) through given training examples. Under the conventional assumption that the training examples $\{(\boldsymbol{x_i}, y_i)\}_{i=1}^n$ are identically and independently distributed from the true distribution $p(\boldsymbol{x}, y)$, the following empirical risk estimator is unbiased:

$$\widehat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(\boldsymbol{x}_i), y_i). \tag{2.2}$$

because the following equality holds:

$$\mathbb{E}_{\{(\boldsymbol{x_i},y_i)\}_{i=1}^n \overset{\text{i.i.d}}{\sim} p(\boldsymbol{x},y)} \left[\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(\boldsymbol{x}_i), y_i) \right] = \mathbb{E}_{p(\boldsymbol{x},y)} \Big[\mathcal{L}(f(\boldsymbol{x}), y) \Big].$$

Therefore, we usually resort to *empirical risk minimization*.

However, finding the minimizer $\widehat{f}$ of the empirical risk in Eq. (2.2) is generally very hard when zero-one loss is used because it is hard to be optimized due to it is not convex and not continuous. Therefore, a surrogate loss instead of 0-1 loss is usually used for practical implementation. Which means, we aim to find the minimizer of the risk using the surrogate loss. we will give some examples of such surrogate loss for binary classification and multi-class classification in the following subsection.

2.1.1 Binary Classification

A classification problem is called binary classification when the size of the label space is two, i.e., $|\mathcal{Y}| = 2$. For binary classification, we normally represent the label space $\mathcal{Y}$ as $\{+1, -1\}$ where $+1/-1$ denotes the positive/negative label. Then, we specially introduce

a surrogate loss function ℓ for binary classification and a model $f : \mathcal{X} \to \mathbb{R}$, and the objective is to optimize the following risk:

$$R(f) = \mathbb{E}_{p(\boldsymbol{x},y)}[\ell(yf(\boldsymbol{x}))]. \tag{2.3}$$

We provide several examples of such surrogate loss function ℓ for binary classification. For example, ℓ can be the hinge loss:

$$\ell(z) = \max(0, 1 - z), \tag{2.4}$$

or logistic loss:

$$\ell(z) = \frac{1}{1 + \exp(-z)}. \tag{2.5}$$

The above binary loss functions given are convex and continuous, so it is relatively easier to optimize the empirical risk given these loss functions for binary classification. Finally, a binary classifier $h : \mathcal{X} \to \mathcal{Y}$ could be induced as follows:

$$h(\boldsymbol{x}) = \begin{cases} +1 & \text{if } f(\boldsymbol{x}) \geq 0, \\ -1 & \text{if } f(\boldsymbol{x}) < 0. \end{cases}$$

2.1.2 Multi-Class Classification

A classification problem is called multi-class classification when the size of the label space $\mathcal{Y}$ is larger than 2, i.e., $|\mathcal{Y}| \geq 3$. For multi-class classification, we normally represent the label space as $\mathcal{Y} = \{1, 2, \ldots, k\}$. Then, we specially introduce a surrogate loss function $\mathcal{L}$ for binary classification and a model $f : \mathcal{X} \to \mathbb{R}^k$, and the objective is to optimize the following risk:

$$R(f) = \mathbb{E}_{p(\boldsymbol{x},y)}\Big[\mathcal{L}(f(\boldsymbol{x}), y)\Big]. \tag{2.6}$$

We provide several examples of such surrogate loss function $\mathcal{L}$ for multi-class classification. For example, $\mathcal{L}$ can be the *one-versus-all* (OVA) loss [35, 61]:

$$\mathcal{L}_{\text{OVA}}(f(x), y) = \ell(f_y(\boldsymbol{x})) + \frac{1}{k-1} \sum_{y' \neq y} \ell(-f_{y'}(\boldsymbol{x})), \tag{2.7}$$

or the *pairwise comparison* (PC) loss [35, 61]:

$$\mathcal{L}_{\mathrm{PC}}(f(x), y) = \sum_{y' \neq y} \ell(f_y(\boldsymbol{x}) - f_{y'}(\boldsymbol{x})), \tag{2.8}$$

or the *categorical cross entropy* (CCE) loss:

$$\mathcal{L}_{\mathrm{CCE}}(f(x), y) = -\log \frac{\exp(f_y(\boldsymbol{x}))}{\sum_{y' \in \mathcal{Y}} \exp(f_{y'}(\boldsymbol{x}))}. \tag{2.9}$$

The above multi-class loss functions given are convex and continuous, so it is relatively easier to optimize the empirical risk given these loss functions for multi-class classification. Finally, a multi-class classifier $h : \mathcal{X} \to \mathcal{Y}$ could be induced as follows:

$$h(\boldsymbol{x}) = \mathrm{argmax}_{y \in \mathcal{Y}} f_y(\boldsymbol{x}).$$

2.2 Advanced Topics in Weakly Supervised Learning

In supervised learning, we assume that training examples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ which are composed of pairs of an instance $\boldsymbol{x} \in \mathcal{X}$ and its correct label $y \in \mathcal{Y}$ are identically and independently sampled from the true data distribution $p(\boldsymbol{x}, y)$. However, in many real-world scenarios, it is sometimes difficult to collect the correct label of the instance. The objective of weakly supervised learning is trying to train a classifier from a dataset where the instances do not necessarily have their correct labels.

In this thesis, we focus on two advanced topics in weakly supervised learning, including *complementary-label learning* and *partial-label learning*.

2.2.1 Complementary-Label Learning

Complementary-label learning solves the problem where each training example is supplied with a single complementary label, which only specifies one of the classes that the example does not belong to. Suppose the complementarily labeled examples are denoted by $\{(\boldsymbol{x}_i, \overline{y}_i)\}_{i=1}^n$, where $\overline{y}_i \in \mathcal{Y}$ is a complementary label of the instance $\boldsymbol{x}_i$, and each complementarily labeled example is independently sampled from $\overline{p}(\boldsymbol{x}, \overline{y})$. Since we do not have ordinary labels when only complementarily labeled data are provided, we cannot

follow the general validation method that utilizes the zero-one error or classification accuracy. In this case, an unbiased estimator of the classification risk in Eq. (2.1) allows us to conduct cross validation based on the empirical risk. Besides, having an unbiased risk estimator can also enable us to conduct effective empirical risk minimization.

In order to provide an unbiased estimator of the classification risk in Eq. (2.1) using only complementarily labeled data, it is important to make some assumptions on $\overline{p}(\boldsymbol{x}, \overline{y})$. Previous studies [35, 36] assumed that $\overline{p}(\boldsymbol{x}, \overline{y})$ could be formulated as:

$$\overline{p}(\boldsymbol{x}, \overline{y}) = \frac{1}{k-1} \sum\nolimits_{y \neq \overline{y}} p(\boldsymbol{x}, y). \tag{2.10}$$

This assumption implies that all other labels except the correct label are chosen to be the complementary label with uniform probabilities. This is reasonable as we do not have extra labeling information except a complementary label. Under this assumption, it was first proved by [35] that an unbiased estimator of the classification risk in Eq. (2.1) can be obtained from only complementarily labeled data, when the loss function satisfies certain conditions. Specifically, they used the multi-class loss functions with the one-versus-all strategy and the pairwise comparison strategy [61]:

$$\begin{aligned}
\overline{\mathcal{L}}_{\mathrm{OVA}}\big(f(\boldsymbol{x}), \overline{y}\big) &= \frac{1}{k-1} \sum\nolimits_{y' \neq \overline{y}} \ell\big(f_{y'}(\boldsymbol{x})\big) + \ell\big(-f_{\overline{y}}(\boldsymbol{x})\big), \\
\overline{\mathcal{L}}_{\mathrm{PC}}\big(f(\boldsymbol{x}), \overline{y}\big) &= \sum\nolimits_{y' \neq \overline{y}} \ell\big(f_{y'}(\boldsymbol{x}) - f_{\overline{y}}(\boldsymbol{x})\big),
\end{aligned}$$

where $\ell(z)$ is a binary loss function that satisfies the following symmetric condition:

$$\ell(z) + \ell(-z) = 1.$$

There are some common loss binary loss functions that satisfy the symmetric condition, including *zero-one loss*:

$$\ell_{0\text{-}1}(z) = \begin{cases} 1, & \text{if } z \geq 0, \\ 0, & \text{otherwise,} \end{cases}$$

sigmoid loss:

$$\ell_{\mathrm{S}}(z) = \frac{1}{1 + e^{z}},$$

and *ramp loss*:

$$\ell_{\mathrm{R}}(z) = \frac{1}{2}\max(0, \min(2, 1 - z)).$$

Later, another different assumption was used by [62]. They assumed that all other labels except the correct label are chosen to be the complementary label with different probabilities, and proposed to estimate the class transition probability matrix for model training. Although they showed that the minimizer of their learning objective coincides with the minimizer of the classification risk in Eq. (2.1), they did not provide an unbiased risk estimator.

Recently, a more general unbiased risk estimator [36] was proposed, which does not rely on specific losses or models. Their formulation is as follows:

$$\overline{\mathcal{L}}_{\mathrm{Free}}\big(f(\boldsymbol{x}), \overline{y}\big) = \sum_{y=1}^{k} \mathcal{L}\big(f(\boldsymbol{x}), y\big) - (k-1)\mathcal{L}\big(f(\boldsymbol{x}), \overline{y}\big). \tag{2.11}$$

The above risk estimator is unbiased because it would be equivalent to the classification risk in Eq. (2.6) in expectation: $\mathbb{E}_{\overline{p}(\boldsymbol{x},\overline{y})}[\overline{\mathcal{L}}_{\mathrm{Free}}\big(f(\boldsymbol{x}), \overline{y}\big)] = \mathbb{E}_{p(\boldsymbol{x},y)}[\mathcal{L}(f(\boldsymbol{x}), y)]$, where $(\boldsymbol{x}, \overline{y})$ is a complementarily labeled example and $(\boldsymbol{x}, y)$ is an ordinarily labeled example. For the above formulation, it was shown that due to the existence of the negative term, the empirical risk could be unbounded below, which would lead to over-fitting. In order to alleviate this issue, the authors further proposed modified versions by using the max operator and the gradient ascent strategy. They showed that their risk estimator can be used not only for a learning objective, but also as a validation criterion for the above two methods.

2.2.2 Partial-Label Learning

Partial-label learning [38] aims to learn from examples that are assigned multiple candidate labels, while only one of which is correct. This learning paradigm is also termed as *superset label learning* [41, 63–65] or *ambiguous label learning* [39, 66–69]. By regarding the candidate labels as non-complementary labels, partial-label learning could be viewed as a generalization of complementary-label learning. Since manually labeling the ground-truth label of each instance could incur unaffordable monetary or time cost,

partial-label learning has various application domains, such as web mining [40], image annotation [38, 39], and ecoinformatics [41].

To this end, a number of approaches [41, 65, 70] have been proposed to improve the practical performance of learning from partially labeled data. For example, the Expectation-Maximization (EM) algorithm [71] was adopted to alternatively update the confidence of each candidate label and the model parameters. Inspired by this pioneering study, many followers [67, 72–76] proposed variants of the standard EM algorithm to improve the estimation of the confidence of each candidate label (E step) or improve the model training process (M step). In addition to these EM-based methods, there are also some other parametric methods. For example, the pioneer work called PL-SVM [77] aims maximize the margin between the maximum modeling output from candidate labels and that from non-candidate labels, i.e.,

$$\min_f \max\left(0, 1 - \left[\max_{y \in S_i} f_y(\boldsymbol{x}_i) - \max_{y' \notin S_i} f_{y'}(\boldsymbol{x}_i)\right]\right),$$

where S_i denotes the candidate label set assigned to $\boldsymbol{x}_i$ and $f_y(\boldsymbol{x}_i)$ denotes the y-th element of the model output vector $f(\boldsymbol{x}_i)$. Later, a number of methods [78–81] were proposed to improve the maximum margin technique for PLL. A *convex loss for partial labels* (CLPL) [38] was proposed to modify the one-versus-all multi-class formulation, which aims to maximize the average prediction score of candidate labels and minimize the prediction score of each non-candidate label, i.e.,

$$\min_f \frac{1}{n} \sum_{i=1}^{n} \left(\ell\left(\frac{1}{|S_i|} \sum_{y \in S_i} f_y(\boldsymbol{x}_i)\right) + \sum_{y' \notin S_i} \ell(-f_{y'}(\boldsymbol{x}))\right), \tag{2.12}$$

where ℓ denotes a binary loss function, and the squared hinge loss is used in CLPL [38,82]. CLPL is an upper bound of the ordinary 0-1 loss and holds a generalization bound that guarantees the statistical consistency.

Besides, some non-parametric methods have also been proposed. To name a few, the K-NN method [66] was adapted to deal with candidate labels, and the label propagation strategy [83] was applied to iteratively update the confidence of each candidate label [72, 72]. Although the above methods have achieved promising performance for learning with partial labels, most of them suffer from the inefficiency of optimization due to the

specially designed objective function. Therefore, it is improbable to use them on large-scale datasets, which could be a severe problem in the current era of big data.

On the theoretical side, some researchers have studied the statistical consistency [38] and learnability [63] of partial-label learning. They made the same assumption on the *ambiguity degree*, which describes the maximum probability of an incorrect candidate label appearing with the correct label. Although they assumed that the data distribution for successful partial-label learning should ensure a limited ambiguity degree, it is still unclear what the explicit formulation of the data distribution would be. Besides, the consistency of partial-label learning methods would be hardly guaranteed without modeling the data distribution.

Chapter 3

Learning with Multiple Complementary Labels

3.1 Motivation

In complementary-label learning [35, 36, 62], each training example is supplied with a *complementary label* (CL), which specifies one of the classes that the example does *not* belong to. Compared with ordinary labels, it is obviously easier to collect CLs. Recently, complementary-label learning has been applied to online learning [84] and medical image segmentation [42]. In addition, another potential application of learning with CLs would be data privacy. For example, collecting some survey data may require extremely private questions [35,36]. It may be difficult for us to directly obtain the true answer (label) to the question. Nonetheless, it would be mentally less demanding if we ask the respondent to provide some incorrect answers. Besides, the respondent may provide multiple incorrect answers, rather than exactly one. In this case, *multiple complementary labels* (MCLs) would be more widespread than a single CL.

In this chapter[1], we propose a novel problem setting (Section 3.2.1) that allows MCLs for each example, and provide a real-world example (Section 3.2.2) to justify this problem setting. Although existing complementary-label learning methods [35, 36, 62] have provided solid theoretical foundations and achieved promising performance, they are all restricted to the case where each example is associated with a single CL. To learn with MCLs, we first design two wrappers (Section 3.3.1) that decompose each example with

[1]This chapter has been published in [37].

MCLs into multiple examples, each with a single CL, in different manners. With the two wrappers, we are able to use arbitrary ordinary complementary-label learning methods for learning with MCLs. However, the derived data with many single CLs may not match the assumed data distribution for complementary-label learning [35,36]. In addition, the supervision information would be conceptually diluted after decomposition.

In order to solve the above problems, we further propose an unbiased risk estimator (Section 3.3.2) for learning with MCLs, which processes each set of MCLs as a whole. Our risk estimator is conceptually consistent, and builds a prototype baseline for the new problem setting that may inspire more specially designed methods for this new setting in the future. Then, we theoretically derive an estimation error bound, which guarantees that the empirical risk minimizer converges to the true risk minimizer with high probability as the number of training data approaches infinity. Furthermore, we improve the risk estimator into minimizing properly chosen upper bounds for practical implementation (Section 3.3.3), and we show that they bring benefits to gradient update. Experimental results show that the wrappers work well for learning with MCLs while the (improved) risk estimator is even better on various benchmark datasets.

3.2 Multiple Complementary Labels

In this section, we first introduce our problem setting where each example is associated with MCLs, and then provide a corresponding real-world motivation.

3.2.1 Data Generation Process

Suppose the given dataset for learning with MCLs is represented as $\overline{\mathcal{D}} = \{(\boldsymbol{x}_i, \overline{Y}_i)\}_{i=1}^{n}$, where $\overline{Y}_i$ is a set of complementary labels for the instance $\boldsymbol{x}_i$. It is obvious that learning with MCLs is a generalization of complementary-label learning that learns with a single CL. Specifically, if $\overline{Y}_i$ contains only one complementary label with probability 1, we obtain a complementary-label learning problem. In addition, if $\overline{Y}_i$ contains $k-1$ complementary labels where k denotes the total number of classes, we obtain an ordinary multi-class classification problem. It is easy to know that for all i, $\overline{Y}_i$ cannot be the empty set nor the full label set, hence $\overline{Y}_i \in \overline{\mathcal{Y}}$ where $\overline{\mathcal{Y}} = \{2^{\mathcal{Y}} - \emptyset - \mathcal{Y}\}$ and $|\overline{\mathcal{Y}}| = 2^k - 2$.

It is worth noting that by regarding MCLs as non-candidate labels, learning with MCLs can be considered as partial-label learning [38]. However, there is no unbiased risk estimator provided for partial-label learning. Besides, another significant difference between complementary-label learning and partial-label learning lies in the key assumption. Existing partial-label learning methods usually focus on the dominance relation assumption [38], while complementary-label learning methods generally make an assumption on the data generation process (e.g., Eq. (2.10)). In this chapter, we will make a generalized assumption for the generation process of the examples with MCLs, and show that an unbiased risk estimator for learning with MCLs can be obtained.

For the generation process of training examples with MCLs, we assume that it relies on the size of the set of MCLs. Let us represent the size of the complementary label set by a random variable s, and assume s is sampled from a distribution $p(s)$. In this way, we assume that each training example $(\boldsymbol{x}_i, \overline{Y}_i)$ is drawn from the following data distribution:

$$\overline{p}(\boldsymbol{x}, \overline{Y}) = \sum_{j=1}^{k-1} p(s=j)\overline{p}(\boldsymbol{x}, \overline{Y} \mid s=j), \tag{3.1}$$

where

$$\overline{p}(\boldsymbol{x}, \overline{Y} \mid s=j) := \begin{cases} \frac{1}{\binom{k-1}{j}} \sum_{y \notin \overline{Y}} p(\boldsymbol{x}, y), & \text{if } |\overline{Y}| = j, \\ 0, & \text{otherwise.} \end{cases}$$

It is clear that when $p(s=1)=1$, our introduced distribution reduces to the assumed distribution (e.g., Eq. (2.10)) in ordinary complementary-label learning methods [35,36]. Then, we show that $\overline{p}(\boldsymbol{x}, \overline{Y})$ is a valid probability distribution by the following theorem.

Theorem 3.1 *The following equality holds:*

$$\int_{\overline{\mathcal{Y}}} \int_{\mathcal{X}} \overline{p}(\boldsymbol{x}, \overline{Y}) \mathrm{d}\boldsymbol{x} \ \mathrm{d}\overline{Y} = 1. \tag{3.2}$$

Proof. Firstly, we define the set of all the possible label sets whose size is j as

$$\overline{\mathcal{Y}}_j := \{Y \mid Y \in \overline{\mathcal{Y}}, |Y| = j\}.$$

Then, by the definition of $\overline{p}(\boldsymbol{x}, \overline{Y})$, we can obtain

$$
\begin{aligned}
&\int_{\overline{\mathcal{Y}}} \int_{\mathcal{X}} \overline{p}(\boldsymbol{x}, \overline{Y}) \mathrm{d}\boldsymbol{x} \ \mathrm{d}\overline{Y} \\
&= \int \sum_{\overline{Y} \in \overline{\mathcal{Y}}} \overline{p}(\boldsymbol{x}, \overline{Y}) \mathrm{d}\boldsymbol{x} \\
&= \int \sum_{\overline{Y} \in \overline{\mathcal{Y}}} \sum_{j=1}^{k-1} \Big(\overline{p}(\boldsymbol{x}, \overline{Y} \mid s = j) p(s = j) \Big) \mathrm{d}\boldsymbol{x} \\
&= \int \sum_{j=1}^{k-1} \sum_{\overline{Y} \in \overline{\mathcal{Y}}_j} \Big(\overline{p}(\boldsymbol{x}, \overline{Y} \mid s = j) p(s = j) \Big) \mathrm{d}\boldsymbol{x} \qquad (\because \overline{\mathcal{Y}}_j := \{\overline{Y} \mid \overline{Y} \in \overline{\mathcal{Y}}, |\overline{Y}| = j\}) \\
&= \int \sum_{j=1}^{k-1} \sum_{\overline{Y} \in \overline{\mathcal{Y}}_j} \left(\frac{1}{\binom{k-1}{j}} \sum_{y \notin \overline{Y}} p(\boldsymbol{x}, y) p(s = j) \right) \mathrm{d}\boldsymbol{x} \\
&= \int \sum_{j=1}^{k-1} \left(\frac{1}{\binom{k-1}{j}} \frac{\binom{k}{j}(k-j)}{k} \sum_{y=1}^{k} p(\boldsymbol{x}, y) p(s = j) \right) \mathrm{d}\boldsymbol{x} \qquad \left(\because |\overline{\mathcal{Y}}_j| = \binom{k}{j} \right) \\
&= \int \sum_{j=1}^{k-1} p(\boldsymbol{x}) p(s = j) \mathrm{d}\boldsymbol{x} \\
&= 1,
\end{aligned}
$$

which concludes the proof of Theorem 3.1. ■

3.2.2 Motivation for Data Generation

Here, we provide a motivation for why we adopt the above generation process of training data with MCLs.

Since directly choosing the correct label is hard for labelers, it would be easier if a labeling system can randomly choose a label set and ask labelers whether the correct label is included in the proposed label set or not. Given a pattern $\boldsymbol{x}$, suppose the labeling system first randomly samples the size s of the proposed label set from $p(s)$, and then randomly and uniformly chooses a specific label set with size s from $\overline{\mathcal{Y}}$. In this way, the collected label sets that do not include the correct label precisely follow the same distribution as Eq. (3.1). We will demonstrate this fact in the following.

We start by considering the case where the labeling system has already sampled the size s of the proposed label set. Then we have the following lemma.

Lemma 3.1 *Given the sampled size s of the proposed label set, for any pattern $\boldsymbol{x}$ with its correct label y and any label set $\overline{Y}$ with size s (i.e., $|\overline{Y}| = s$), the following equality holds:*

$$p(y \in \overline{Y} \mid \boldsymbol{x}, s) = \frac{s}{k}. \tag{3.3}$$

Proof. Let us consider the case where the correct label y is a specific label i (i.e., $i \in \{1, 2, \cdots, k\}$), then we have

$$\begin{aligned} p(y \in \overline{Y}, y = i \mid \boldsymbol{x}, s) =& p(y \in \overline{Y} \mid y = i, \boldsymbol{x}, s) p(y = i \mid \boldsymbol{x}, s) \\ =& \sum_{C \in \overline{\mathcal{Y}}} p(y \in \overline{Y}, \overline{Y} = C \mid y = i, \boldsymbol{x}, s) p(y = i \mid \boldsymbol{x}, s). \end{aligned}$$

Here, $p(y = i \mid \boldsymbol{x}, s) = p(y = i \mid \boldsymbol{x})$ since the labeling rule is independent of s. In addition, $\sum_{C \in \overline{\mathcal{Y}}} p(y \in \overline{Y}, \overline{Y} = C \mid y = i, \boldsymbol{x}, s) = \sum_{C \in \overline{\mathcal{Y}}_s} p(y \in \overline{Y}, \overline{Y} = C \mid y = i, \boldsymbol{x})$ since given the size s of the label set, the whole set of all the possible label sets becomes $\overline{\mathcal{Y}}_s$. Then, we can obtain

$$\begin{aligned} p(y \in \overline{Y}, y = i \mid \boldsymbol{x}, s) &= \sum_{C \in \overline{\mathcal{Y}}} p(y \in \overline{Y}, \overline{Y} = C \mid y = i, \boldsymbol{x}, s) p(y = i \mid \boldsymbol{x}, s) \\ &= \sum_{C \in \overline{\mathcal{Y}}_s} p(y \in \overline{Y}, \overline{Y} = C \mid y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) \\ &= \sum_{C \in \overline{\mathcal{Y}}_s} p(y \in \overline{Y} \mid \overline{Y} = C, y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) p(\overline{Y} = C \mid \boldsymbol{x}) \\ &= \sum_{C \in \overline{\mathcal{Y}}_s} p(y \in \overline{Y} \mid \overline{Y} = C, y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) p(\overline{Y} = C), \end{aligned}$$

where the last equality holds due to the fact that for each instance $\boldsymbol{x}$, $\overline{Y}$ is uniformly and randomly chosen. Since $p(\overline{Y} = C) = \frac{1}{|\overline{\mathcal{Y}}_s|}$ if $C \in \overline{\mathcal{Y}}_s$ where $|\overline{\mathcal{Y}}_s| = \binom{k}{s}$, we have

$$\begin{aligned} p(y \in \overline{Y}, y = i \mid \boldsymbol{x}, s) =& \sum_{C \in \overline{\mathcal{Y}}_s} p(y \in \overline{Y} \mid \overline{Y} = C, y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) p(\overline{Y} = C) \\ =& \frac{1}{\binom{k}{s}} \sum_{C \in \overline{\mathcal{Y}}_s} p(y \in \overline{Y} \mid \overline{Y} = C, y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) \\ =& \frac{1}{\binom{k}{s}} |\overline{\mathcal{Y}}^i_s| p(y = i \mid \boldsymbol{x}) \qquad \left(\because \overline{\mathcal{Y}}^i_s := \{\overline{Y} \in \overline{\mathcal{Y}}_s \mid i \in \overline{Y}\} \right) \\ =& \frac{\binom{k-1}{s-1}}{\binom{k}{s}} p(y = i \mid \boldsymbol{x}) \qquad \left(\because |\overline{\mathcal{Y}}^i_s| = \binom{k-1}{s-1} \right) \\ =& \frac{s}{k} p(y = i \mid \boldsymbol{x}). \end{aligned}$$

By further summing up the both side over all the possible i, we can obtain

$$p(y \in \overline{Y} \mid \boldsymbol{x}, s) = \frac{s}{k},$$

which concludes the proof of Lemma 3.1.

Theorem 3.2 *In the above setting, the distribution of collected data where the correct label y ($y \in \mathcal{Y}$) is not included in the label set $\overline{Y}$ ($\overline{Y} \in \overline{\mathcal{Y}}$) is the same as Eq. (3.1), i.e.,*

$$p(\boldsymbol{x}, \overline{Y} \mid y \notin \overline{Y}) = \overline{p}(\boldsymbol{x}, \overline{Y}). \tag{3.4}$$

Proof. Let us express $p(\overline{Y}|y \notin \overline{Y}, \boldsymbol{x}, s)$ as

$$\begin{aligned} p(\overline{Y}|y \notin \overline{Y}, \boldsymbol{x}, s) =& \frac{p(y \notin \overline{Y}, \overline{Y}|\boldsymbol{x}, s)}{p(y \notin \overline{Y}|\boldsymbol{x}, s)} \\ =& \frac{p(y \notin \overline{Y}|\overline{Y}, \boldsymbol{x}, s)p(\overline{Y}|\boldsymbol{x}, s)}{p(y \notin \overline{Y}|\boldsymbol{x}, s)} \\ =& \frac{p(y \notin \overline{Y}|\overline{Y}, \boldsymbol{x}, s)p(\overline{Y}|s)}{p(y \notin \overline{Y}|\boldsymbol{x}, s)}, \end{aligned}$$

where the last equality holds because $\overline{Y}$ is influenced by the size s, and for each instance $\boldsymbol{x}$, $\overline{Y}$ is uniformly and randomly chosen. Note that given s, there are $|\overline{\mathcal{Y}}_s|$ possible label sets, thus $p(\overline{Y}|s) = \frac{1}{|\overline{\mathcal{Y}}_s|}$ where $|\overline{\mathcal{Y}}_s| = \binom{k}{s}$. In this way, we have

$$\begin{aligned} p(\overline{Y}|y \notin \overline{Y}, \boldsymbol{x}, s) =& \frac{p(y \notin \overline{Y}|\overline{Y}, \boldsymbol{x}, s)p(\overline{Y}|s)}{p(y \notin \overline{Y}|\boldsymbol{x}, s)} \\ =& \frac{1}{\binom{k}{s}} \frac{p(y \notin \overline{Y}|\overline{Y}, \boldsymbol{x}, s)}{1 - p(y \in \overline{Y}|\boldsymbol{x}, s)} \\ =& \frac{1}{\binom{k}{s}} \frac{1}{1 - \frac{s}{k}} p(y \notin \overline{Y}|\overline{Y}, \boldsymbol{x}, s) \qquad \left(\because \text{by Lemma 1, } p(y \in \overline{Y}|\boldsymbol{x}, s) = \frac{s}{k} \right) \\ =& \frac{1}{\binom{k}{s}} \frac{k}{k - s} \sum_{y \notin \overline{Y}} p(y|\boldsymbol{x}, s) \\ =& \frac{1}{\binom{k-1}{s}} \sum_{y \notin \overline{Y}} p(y|\boldsymbol{x}). \end{aligned}$$

By multiplying $p(\boldsymbol{x})$ on both side, we have

$$p(\boldsymbol{x}, \overline{Y}|y \notin \overline{Y}, s) = \frac{1}{\binom{k-1}{s}} \sum_{y \notin \overline{Y}} p(\boldsymbol{x}, y).$$

Then by taking into account the variable s, we have

$$\begin{aligned} p(\boldsymbol{x}, \overline{Y} | y \notin \overline{Y}) =& \sum_{j=1}^{k-1} p(s=j) p(\boldsymbol{x}, \overline{Y} | y \notin \overline{Y}, s=j) \\ =& \sum_{j=1}^{k-1} p(s=j) \frac{1}{\binom{k-1}{j}} \sum_{y \notin \overline{Y}} p(\boldsymbol{x}, y) \\ =& \overline{p}(\boldsymbol{x}, \overline{Y}), \end{aligned}$$

which concludes the proof of Theorem 3.2. ■

3.3 Learning with Multiple Complementary Labels

In this section, we first present two wrappers that enable us to use any complementary-label learning method for learning with MCLs. Then, we present an unbiased risk estimator for learning with MCLs as a whole, and establish an estimation error bound.

3.3.1 Wrappers

Since ordinary complementary-label learning methods cannot directly deal with MCLs, it would be natural to ask whether there exist some strategies that can enable us to use any existing complementary-label learning method for learning with MCLs.

Motivated by this, we propose two wrappers that decompose each example with MCLs into multiple examples, each with a single CL. Specifically, suppose a training example with MCLs is given as $(\boldsymbol{x}_i, \overline{Y}_i)$ where $\overline{Y}_i = \{\overline{y}_1, \overline{y}_2\}$. Then ordinary complementary label learning approaches may learn from $(\boldsymbol{x}_i, \overline{y}_1)$ and $(\boldsymbol{x}_i, \overline{y}_2)$. According to whether decomposition is after shuffling the training set, there are two decomposition strategies (wrappers) when we optimize a loss function by a stochastic optimization algorithm:

- **Decomposition after Shuffle.** Given the shuffled training set with MCLs, in each mini-batch, we decompose each example into multiple examples, each with a single CL.

- **Decomposition before Shuffle.** Given the training set with MCLs, we drive a new training set by decomposing each example into multiple examples, each with a single CL. Then, we shuffle the derived training set.

Table 3.1: Two ways for handling a set of MCLs (with size s).

Setting	#TP	#FP	Supervision Purity
Many single CLs	s	$(k-2)s$	$1/(k-1)$
A set of MCLs	1	$k-s-1$	$1/(k-s)$

Both two decomposition strategies enable us to use arbitrary ordinary complementary-label learning methods for learning with MCLs. However, the derived training data with many single CLs may not match the originally assumed data distribution (i.e., Eq. (2.10)) for complementary-label learning, since these CLs are completely derived from MCLs while the data distribution with MCLs is relevant to the size of each set of MCLs. As a consequence, the learning consistency would no longer be guaranteed even if the complementary-label learning method inside the wrappers is originally risk-consistent or classifier-consistent.

Moreover, since ordinary complementary-label learning methods can only learn with a single CL for each example at a time and treat each example independently, the supervision information for each set of MCLs would be conceptually diluted. We demonstrate this issue in Table 3.1. As shown in Table 3.1, there are two settings according to whether to decompose a set of MCLs into many single CLs or not. Since all the non-complementary labels have the possibility to be the correct label, we specially count how many times the correct label serves as a non-complementary label (denoted by #TP), and how many times the other labels except the correct label serve as a non-complementary label (denoted by #FP). Then the supervision purity is calculated by (#TP)/(#TP+#FP).

Clearly, the wrappers follow the setting where a set of MCLs is decomposed into many single CLs. If the size of the set of MCLs is s, then #TP equals s, since the correct label would serve as a non-complementary label for s times after decomposition, and the other labels except the correct label would serve as a non-complementary label for $(k-s-1)s + s(s-1) = (k-2)s$ times, hence the supervision purity would be $s/(s+(k-2)s) = 1/(k-1)$. However, for the setting where the set of MCLs is not decomposed, we can easily know that the correct label serves as a non-complementary label once, and the other labels expect the correct label serve as a non-complementary

label for $k-s-1$ times, hence the supervision purity is $1/(k-s)$. These observations clearly show that the supervision information is diluted after decomposing MCLs ($s \geq 2$), which also motivate us to take a set of MCLs as a whole set. In the following, we will introduce our proposed unbiased risk estimator, which is able to learn with MCLs as a whole.

3.3.2 Unbiased Risk Estimator

The above example has shown that the supervision information is diluted after decomposition. The basic reason lies in that ordinary complementary-label learning methods are designed by only considering the data distribution with a single CL, i.e., $\overline{p}(\boldsymbol{x}, \overline{y})$. However, the data distribution with MCLs $\overline{p}(\boldsymbol{x}, \overline{Y})$ becomes much different, and the wrappers fail to capture such distribution because they do not treat MCLs as a whole for each training example. To solve this problem, we propose an unbiased estimator of the classification risk in Eq. (2.1) for learning with MCLs as a whole.

We derive the unbiased risk estimator based on the assumed data distribution $\overline{p}(\boldsymbol{x}, \overline{Y})$. We first relate the data distribution with ordinary labels to that with MCLs by the following lemma.

Lemma 3.2 *The following equality holds:*

$$p(\boldsymbol{x}, y) = 1 - \sum_{j=1}^{k-1} \Big(\frac{k-1}{j} \sum_{\overline{Y} \in \overline{\mathcal{Y}}_j^y} \overline{p}(\boldsymbol{x}, \overline{Y}, s = j) \Big),$$

where $\overline{\mathcal{Y}}_j^y$ *is the set of all the possible label sets with size* j *that include a specific label* $y \in \mathcal{Y}$*, i.e.,*

$$\overline{\mathcal{Y}}_j^y := \{\overline{Y} \in \overline{\mathcal{Y}} \mid y \in \overline{Y}, |\overline{Y}| = j\}.$$

Proof. In terms of our introduced data distribution $\overline{p}(\boldsymbol{x}, \overline{Y})$, we have already obtained

$$\overline{p}(\boldsymbol{x}, \overline{Y} \mid s = j) = \frac{1}{\binom{k-1}{j}} \sum_{y' \notin \overline{Y}} p(\boldsymbol{x}, y').$$

Then, we can obtain the following equality by operating $\sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^y}$ on both the left and the right hand side:

$$\sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^y} \overline{p}(\boldsymbol{x},\overline{Y} \mid s=j) = \frac{1}{\binom{k-1}{j}} \sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^y} \sum_{y'\notin\overline{Y}} p(\boldsymbol{x},y'), \tag{3.5}$$

where

$$\overline{\mathcal{Y}}_j^y := \{\overline{Y}\in\overline{\mathcal{Y}} \mid y\in\overline{Y}, |\overline{Y}|=j\}.$$

In this way, the right hand side of the equality in Eq. (3.5) can be transformed by the following derivations:

$$\begin{aligned}
&\frac{1}{\binom{k-1}{j}} \sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^y} \sum_{y'\notin\overline{Y}} p(\boldsymbol{x},y') \\
&= \frac{1}{\binom{k-1}{j}} \sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^y} \left(1 - \sum_{y'\in\overline{Y}} p(\boldsymbol{x},y')\right) \\
&= \frac{|\overline{\mathcal{Y}}_j^y|}{\binom{k-1}{j}} - \frac{1}{\binom{k-1}{j}} \sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^y} \sum_{y'\in\overline{Y}} p(\boldsymbol{x},y') \\
&= \frac{\binom{k-1}{j-1}}{\binom{k-1}{j}} - \frac{1}{\binom{k-1}{j}} \sum_{y'} \sum_{\overline{Y}'\in\{\overline{Y}'\in\overline{\mathcal{Y}}_j^y \mid y'\in\overline{Y}'\}} p(\boldsymbol{x},y') \qquad \left(\because |\overline{\mathcal{Y}}_j^y| = \tbinom{k-1}{j-1}\right) \\
&= \frac{j}{k-j} - \frac{1}{\binom{k-1}{j}} \left\{\binom{k-1}{j-1} p(\boldsymbol{x},y) + \binom{k-2}{j-2} \sum_{y'\neq y} p(\boldsymbol{x},y')\right\} \\
&= \frac{j}{k-j} - \frac{1}{\binom{k-1}{j}} \left\{\binom{k-1}{j-1} p(\boldsymbol{x},y) + \binom{k-2}{j-2} (1 - p(\boldsymbol{x},y))\right\} \\
&= \frac{j}{k-j} - \frac{1}{\binom{k-1}{j}} \left\{\binom{k-2}{j-2} + \binom{k-2}{j-1} p(\boldsymbol{x},y)\right\} \qquad \left(\because \tbinom{k-2}{j-1} = \tbinom{k-1}{j-1} - \tbinom{k-2}{j-2}\right) \\
&= \frac{j}{k-j} - \frac{j(j-1)}{(k-j)(k-1)} - \frac{j}{k-1} p(\boldsymbol{x},y) \\
&= \frac{j}{k-1} - \frac{j}{k-1} p(\boldsymbol{x},y).
\end{aligned} \tag{3.6}$$

Combing Eq. (3.5) and Eq. (3.6), we obtain

$$p(\boldsymbol{x},y \mid s=j) = p(\boldsymbol{x},y) = 1 - \frac{k-1}{j} \sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^y} \overline{p}(\boldsymbol{x},\overline{Y} \mid s=j). \tag{3.7}$$

In the end, by taking into account the variable s, we have

$$\begin{aligned} p(\boldsymbol{x}, y) &= \sum_{j=1}^{k-1} p(s=j) p(\boldsymbol{x}, y \mid s=j) \\ &= \sum_{j=1}^{k-1} p(s=j)\Big(1 - \frac{k-1}{j} \sum_{\overline{Y} \in \overline{\mathcal{Y}}_j^y} \overline{p}(\boldsymbol{x}, \overline{Y} \mid s=j)\Big) \\ &= 1 - \sum_{j=1}^{k-1} \Big(\frac{k-1}{j} \sum_{\overline{Y} \in \overline{\mathcal{Y}}_j^y} \overline{p}(\boldsymbol{x}, \overline{Y}, s=j)\Big), \end{aligned}$$

which concludes the proof of Lemma 3.2. ■

Based on Lemma 3.2, we derive an unbiased estimator of the ordinary classification risk Eq. (2.1) by the following theorem.

Theorem 3.3 *The ordinary classification risk in Eq. (2.1) can be equivalently expressed as*

$$R(f) = \sum_{j=1}^{k-1} p(s=j) \overline{R}_j(f), \tag{3.8}$$

where

$$\overline{R}_j(f) := \mathbb{E}_{\overline{p}(\boldsymbol{x}, \overline{Y} \mid s=j)} [\overline{\mathcal{L}}_j\big(f(\boldsymbol{x}), \overline{Y}\big)], \tag{3.9}$$

and

$$\overline{\mathcal{L}}_j\big(f(\boldsymbol{x}), \overline{Y}\big) := \sum_{y \notin \overline{Y}} \mathcal{L}\big(f(\boldsymbol{x}), y\big) - \frac{k-1-j}{j} \sum_{y' \in \overline{Y}} \mathcal{L}\big(f(\boldsymbol{x}), y'\big). \tag{3.10}$$

It is intuitive to obtain

$$R(f) = \mathbb{E}_{p(\boldsymbol{x}, y)}\big[\mathcal{L}\big(f(\boldsymbol{x}), y\big)\big] = \sum_{j=1}^{k-1} p(s=j) \mathbb{E}_{p(\boldsymbol{x}, y \mid s=j)}\big[\mathcal{L}\big(f(\boldsymbol{x}), y\big)\big].$$

Then, we express the right hand side for each $j \in \{1, \ldots, k-1\}$ as

$$
\begin{aligned}
&\mathbb{E}_{p(\boldsymbol{x},y|s=j)}\big[\mathcal{L}\big(f(\boldsymbol{x}),y\big)\big] \\
&= \mathbb{E}_{p(\boldsymbol{x}|s=j)}\mathbb{E}_{p(y|\boldsymbol{x},s=j)}\big[\mathcal{L}\big(f(\boldsymbol{x}),y\big)\big] \\
&= \mathbb{E}_{p(\boldsymbol{x}|s=j)}\Big[\sum_{y=1}^{k} p(y|\boldsymbol{x},s=j)\mathcal{L}\big(f(\boldsymbol{x}),y\big)\Big] \\
&= \mathbb{E}_{p(\boldsymbol{x}|s=j)}\Big[\sum_{y=1}^{k}\Big(1-\frac{k-1}{j}\sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^{y}}\overline{p}(\overline{Y}|\boldsymbol{x},s=j)\Big)\mathcal{L}\big(f(\boldsymbol{x}),y\big)\Big] \qquad (\because \text{Eq. (3.7)}) \\
&= \mathbb{E}_{p(\boldsymbol{x}|s=j)}\Big[\sum_{y=1}^{k}\mathcal{L}\big(f(\boldsymbol{x}),y\big)-\frac{k-1}{j}\sum_{y=1}^{k}\sum_{\overline{Y}\in\overline{\mathcal{Y}}_j^{y}}\overline{p}(\overline{Y}|\boldsymbol{x},s=j)\mathcal{L}\big(f(\boldsymbol{x}),y\big)\Big] \\
&= \mathbb{E}_{p(\boldsymbol{x}|s=j)}\Big[\sum_{y=1}^{k}\mathcal{L}\big(f(\boldsymbol{x}),y\big)-\frac{k-1}{j}\sum_{\overline{Y}\in\overline{\mathcal{Y}}_j}\sum_{y'\in\overline{Y}}\overline{p}(\overline{Y}|\boldsymbol{x},s=j)\mathcal{L}\big(f(\boldsymbol{x}),y\big)\Big] \\
&= (*),
\end{aligned}
$$

where $(*)$ can be further expressed as:

$$
\begin{aligned}
(*) &= \mathbb{E}_{p(\boldsymbol{x}|s=j)}\Big[\sum_{y=1}^{k}\mathcal{L}\big(f(\boldsymbol{x}),y\big)-\frac{k-1}{j}\sum_{\overline{Y}\in\overline{\mathcal{Y}}_j}\overline{p}(\overline{Y}|\boldsymbol{x},s=j)\Big(\sum_{y'\in\overline{Y}}\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\Big)\Big] \\
&= \mathbb{E}_{p(\boldsymbol{x}|s=j)}\mathbb{E}_{\overline{p}(\overline{Y}|\boldsymbol{x},s=j)}\Big[\sum_{y=1}^{k}\mathcal{L}\big(f(\boldsymbol{x}),y\big)-\frac{k-1}{j}\sum_{y'\in\overline{Y}}\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\Big] \\
&= \mathbb{E}_{\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\Big[\sum_{y\notin\overline{Y}}\mathcal{L}\big(f(\boldsymbol{x}),y'\big)-\frac{k-1-j}{j}\sum_{y'\in\overline{Y}}\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\Big] \\
&= \mathbb{E}_{\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}[\overline{\mathcal{L}}_j\big(f(\boldsymbol{x}),\overline{Y}\big)] \\
&= \overline{R}_j(f).
\end{aligned}
$$

In this way, we can obtain $R(f) = \sum_{j=1}^{k-1} p(s=j)\overline{R}_j(f)$, which concludes the proof of Theorem 3.3. ∎

It is easy to verify that Eq. (3.8) reduces to Eq. (2.11) when $p(s=1)=1$. Which means, our approach is a generalization of [36]. Furthermore, according to Corollary 2 in [36], our approach is also a generalization of [35].

Given the dataset with MCLs $\overline{\mathcal{D}} = \{(\boldsymbol{x}_i, \overline{Y}_i)\}_{i=1}^{n}$, we can empirically approximate $p(s=j)$ by n_j/n where n_j denotes the number of examples whose complementary label

set size is j. By further taking into account Eqs. (3.8)-(3.10), we can obtain the following empirical approximation of the unbiased risk estimator introduced in Theorem 3.3:

$$\widehat{R}(f) = \frac{1}{n}\sum_{i=1}^{n}\Big(\sum_{y\notin\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}_i), y\big) - \frac{k-1-|\overline{Y}_i|}{|\overline{Y}_i|}\sum_{y'\in\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}_i), y'\big)\Big). \tag{3.11}$$

Estimation Error Bound. Here, we theoretically analyze the estimation error bound of the proposed unbiased risk estimator based on *Rademacher complexity* [85]. We denote by $\mathcal{F} \subset \{f : \mathbb{R}^d \to \mathbb{R}^k\}$ the hypothesis class, $\widehat{f} := \arg\min_{f\in\mathcal{F}} \widehat{R}(f)$ the empirical risk minimizer, and $f^\star = \arg\min_{f\in\mathcal{F}} R(f)$ the true risk minimizer. Besides, we define the functional space $\mathcal{G}_y$ for the label $y \in \mathcal{Y}$ as $\mathcal{G}_y = \{g : \boldsymbol{x} \to f_y(\boldsymbol{x}) \mid f \in \mathcal{F}\}$. Then, we have the following theorem.

Theorem 3.4 *Assume the loss function $\mathcal{L}(f(\boldsymbol{x}), y)$ is ρ-Lipschitz with respect to $f(\boldsymbol{x})$ $(0 < \rho < \infty)$ for all $y \in \mathcal{Y}$. Let $C_{\mathcal{L}} = \sup_{\boldsymbol{x}\in\mathcal{X}, f\in\mathcal{F}, y\in\mathcal{Y}} \mathcal{L}(f(\boldsymbol{x}), y)$ and $\mathfrak{R}_n(\mathcal{G}_y)$ be the Rademacher complexity of $\mathcal{G}_y$ given the sample size n. Then, for any $\delta > 0$, with probability at least $1-\delta$,*

$$R(\widehat{f}) - R(f^\star) \le \sum_{j=1}^{k-1} p(s=j)\Big(\frac{4\sqrt{2}\rho k(k-1)}{j}\sum_{y=1}^{k}\mathfrak{R}_{n_j}(\mathcal{G}_y) + \frac{C_j}{\sqrt{n_j}}\Big), \tag{3.12}$$

where $C_j = (4k-4j-2)C_{\mathcal{L}}\sqrt{\frac{\log\frac{2(k-1)}{\delta}}{2}}$ for all $j \in \{1,\dots,k-1\}$ and n_j denotes the number of examples whose complementary label set size is j.

The definition of Rademacher complexity and the proof of Theorem 3.4 are provided in Appendix A.1. Generally, the Rademacher complexity of $\mathcal{G}_y$, i.e., $\mathfrak{R}_{n_j}(\mathcal{G}_y)$, can be bounded by $C_{\mathcal{G}}/\sqrt{n_j}$ for a positive constant $C_{\mathcal{G}}$ [27,34,86]. Hence Theorem 3.4 shows that the empirical risk minimizer converges to the true risk minimizer with high probability as the number of training data approaches infinity. It is worth noting that this bound is not only related to the Redemacher complexity of the function class, but also s and k. This observation accords with our intuition that the learning task will be harder if the number of classes k increases or the size of the complementary label set s decreases. Besides, if j increases and $p(s=j)$ increases, the estimation error bound would be tighter, since more supervision information would be provided.

3.3.3 Practical Implementation

In this section, we present the practical implementation of our proposed formulation and improvements of the used loss functions. As described above, we have provided a general unbiased risk estimator that is able to use arbitrary loss functions. There arises a question: Can all loss functions work well in our approach? Unfortunately, the answer is negative.

The classification risk in Eq. (2.1) includes an expectation over a non-negative loss $\mathcal{L} : \mathbb{R}^k \times [k] \to \mathbb{R}_+$, hence the expected risk and the empirical approximation are both lower-bounded by zero. However, our proposed risk estimator in Theorem 3.3 contains a negative term. Although the expected risk estimator is unbiased, the empirical risk estimator may become unbounded below if the used loss function is unbounded, thereby leading to over-fitting. Specifically, when the used loss function is unbounded, the optimized loss could be unbounded below due to the negative empirical risk issue. When the optimized loss function is unbounded below, the optimization process becomes pathological and we cannot find a suitable minimizer as the loss could be infinity. Similar issues have also been shown by [16, 36]. The above analysis suggests that a bounded loss is probably better than an unbounded loss, in our empirical risk estimator (i.e., Eq. (3.11)), because the loss would never be negative infinity.

To demonstrate the above conjecture, we would like to insert bounded and unbounded losses into Eq. (3.11), for comparison studies. Note that we assume that the softmax function is absorbed in each loss, and denote by $p_{\boldsymbol{\theta}}(y|\boldsymbol{x}) = \exp(f_y(\boldsymbol{x}))/(\sum_{j=1}^{k} \exp(f_j(\boldsymbol{x})))$ the predicted probability of the instance $\boldsymbol{x}$ belonging to class y, where $\boldsymbol{\theta}$ denotes the parameters of the model f. In this way, we list the compared unbounded and bounded loss functions as follows.

- Categorical Cross Entropy (CCE):

$$\mathcal{L}_{\mathrm{CCE}}(f(\boldsymbol{x}), y) = -\log p_{\boldsymbol{\theta}}(y|\boldsymbol{x}).$$

- Mean Absolute Error (MAE):

$$\mathcal{L}_{\mathrm{MAE}}(f(\boldsymbol{x}), y) = 2 - 2p_{\boldsymbol{\theta}}(y|\boldsymbol{x}).$$

Table 3.2: List of the used bounded and unbounded losses, which can be directly inserted to our (unbiased) empirical risk estimator Eq. (3.11).

Multi-class loss functions	CCE	MAE	MSE	GCE	PHuber-CE
Boundness	Unbounded	Bounded	Bounded	Bounded	Bounded

- Mean Square Error (MSE):

$$\mathcal{L}_{\mathrm{MSE}}(f(\boldsymbol{x}), y) = 1 - 2p_{\boldsymbol{\theta}}(y|\boldsymbol{x}) + \sum_{j=1}^{k} p_{\boldsymbol{\theta}}(j|\boldsymbol{x})^2.$$

- Generalized Cross Entropy (GCE) [87]:

$$\mathcal{L}_{\mathrm{GCE}}(f(\boldsymbol{x}), y) = \frac{(1 - p_{\boldsymbol{\theta}}(y|\boldsymbol{x})^q)}{q},$$

  where $q \in (0, 1]$ is a user-defined hyper-parameter. We set $q = 0.7$, as suggested by [87].

- Partially Huberised Cross Entropy (PHuber-CE) [88]:

$$\mathcal{L}_{\text{PHuber-CE}}(f(\boldsymbol{x}), y) = \begin{cases} -\log p_{\boldsymbol{\theta}}(y|\boldsymbol{x}), & \text{if } p_{\boldsymbol{\theta}}(y|\boldsymbol{x}) \geq \frac{1}{\tau}, \\ -\tau p_{\boldsymbol{\theta}}(y|\boldsymbol{x}) + \log \tau + 1, & \text{else,} \end{cases}$$

  where $\tau > 0$ is a user-defined hyper-parameter. We set $\tau = 10$, because it works well in [88].

The detailed derivations of the above loss functions and their bounds are provided in Appendix A.2. Among these losses, CCE is unbounded while others are bounded. We will experimentally demonstrate (Fig. 3.1) that by inserting the above losses into Eq. (3.11), bounded loss is significantly better than unbounded loss. Furthermore, we conduct a deeper analysis of MAE because MAE has the special property that MAE is not only bounded, but also satisfies the symmetric condition [89], i.e., $\sum_{y=1}^{k} \mathcal{L}_{\mathrm{MAE}}\big(f(\boldsymbol{x}), y\big) = 2k-2$, which means the sum of the losses over all classes is a constant for arbitrary examples. However, is MAE good enough? Previous studies [87, 90] have already shown that MAE suffers from the optimization issue, which would affect its practical performance. To

alleviate this problem, we further improve MAE by proposing two upper-bound surrogate loss functions. Specifically, by using MAE in Eq. (3.11), we obtain

$$\widehat{R}(f) = \frac{k-1}{|\overline{Y}_i|} \sum_{y \notin \overline{Y}_i} \mathcal{L}_{\mathrm{MAE}}\big(f(\boldsymbol{x}_i), y\big) = \frac{2k-2}{|\overline{Y}_i|} \mathcal{L}'_{\mathrm{MAE}}\big(f(\boldsymbol{x}_i), \overline{Y}_i\big) + Z_i, \tag{3.13}$$

where $\mathcal{L}'_{\mathrm{MAE}}\big(f(\boldsymbol{x}_i), \overline{Y}_i\big) := 1 - \sum_{j \notin \overline{Y}_i} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i)$, and Z_i is a constant independent of $f(\boldsymbol{x}_i)$. It is clear that minimizing $\mathcal{L}'_{\mathrm{MAE}}\big(f(\boldsymbol{x}_i), \overline{Y}_i\big)$ is equivalent to minimizing $\sum_{y \notin \overline{Y}_i} \mathcal{L}_{\mathrm{MAE}}\big(f(\boldsymbol{x}, y)\big)$.

Based on this fact, we further introduce two upper-bound surrogate loss functions of $\mathcal{L}'_{\mathrm{MAE}}$:

$$\mathcal{L}_{\mathrm{EXP}}(f(\boldsymbol{x}_i), \overline{Y}_i) = \exp\Big(-\sum_{j \notin \overline{Y}_i} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i)\Big),$$
$$\mathcal{L}_{\mathrm{LOG}}(f(\boldsymbol{x}_i), \overline{Y}_i) = -\log\Big(\sum_{j \notin \overline{Y}_i} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i)\Big).$$

One can easily verify that $\mathcal{L}'_{\mathrm{MAE}}$ is upper bounded by $\mathcal{L}_{\mathrm{EXP}}$ and $\mathcal{L}_{\mathrm{LOG}}$ using the two inequalities $1 - z \leq \exp(-z)$ and $1 - z \leq -\log z$, respectively. By replacing $\mathcal{L}'_{\mathrm{MAE}}$ by $\mathcal{L}_{\mathrm{LOG}}$ and $\mathcal{L}_{\mathrm{LOG}}$ in Eq. (3.13), we obtain two new methods for learning with MCLs. We explain the advantage of $\mathcal{L}_{\mathrm{EXP}}$ and $\mathcal{L}_{\mathrm{LOG}}$ over $\mathcal{L}'_{\mathrm{MAE}}$ by closely examining their gradients:

$$\frac{\partial \mathcal{L}'_{\mathrm{MAE}}}{\partial \boldsymbol{\theta}} = \begin{cases} -\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i) & \text{if } j \notin \overline{Y}_i, \\ 0 & \text{else,} \end{cases}$$
$$\frac{\partial \mathcal{L}_{\mathrm{EXP}}}{\partial \boldsymbol{\theta}} = \begin{cases} -\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i) \cdot w_{\mathrm{EXP}} & \text{if } j \notin \overline{Y}_i, \\ 0 & \text{else,} \end{cases}$$
$$\frac{\partial \mathcal{L}_{\mathrm{LOG}}}{\partial \boldsymbol{\theta}} = \begin{cases} -\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i) \cdot w_{\mathrm{LOG}} & \text{if } j \notin \overline{Y}_i, \\ 0 & \text{else,} \end{cases}$$

where

$$w_{\mathrm{EXP}} = \exp\Big(-\sum_{j \notin \overline{Y}_i} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i)\Big),$$
$$w_{\mathrm{LOG}} = \Big(\sum_{j \notin \overline{Y}_i} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i)\Big)^{-1}.$$

From their gradients, we can clearly observe that $\mathcal{L}'_{\mathrm{MAE}}$ basically treats each example equally, while $\mathcal{L}_{\mathrm{EXP}}$ and $\mathcal{L}_{\mathrm{LOG}}$ give more weights to difficult examples. Concretely, if

Table 3.3: Summary of the used benchmark datasets with the corresponding models.

Dataset	#Train	#Test	#Features	#Classes	Model
MNIST	60000	10000	784	10	MLP (d-500-10), Linear
Kuzushiji-MNIST	60000	10000	784	10	MLP (d-500-10), Linear
Fashion-MNIST	60000	10000	784	10	MLP (d-500-10), Linear
20Newsgroups	16961	1885	1000	20	MLP (d-500-10), Linear
CIFAR-10	50000	10000	3072	10	ResNet, DenseNet
Yeast	1335	149	8	10	Linear
Texture	4950	550	40	11	Linear
Dermatology	329	37	34	6	Linear
Synthetic Control	540	60	60	6	Linear

$\sum_{j \notin \overline{Y}_i} p_{\boldsymbol{\theta}}(j|\boldsymbol{x}_i)$ is small, both w_{EXP} and w_{LOG} would be large. In other words, $\mathcal{L}_{\mathrm{EXP}}$ and $\mathcal{L}_{\mathrm{LOG}}$ pay more attention to hard examples whose sum of the predicted confidences of all the non-complementary labels is small.

3.4 Experiments

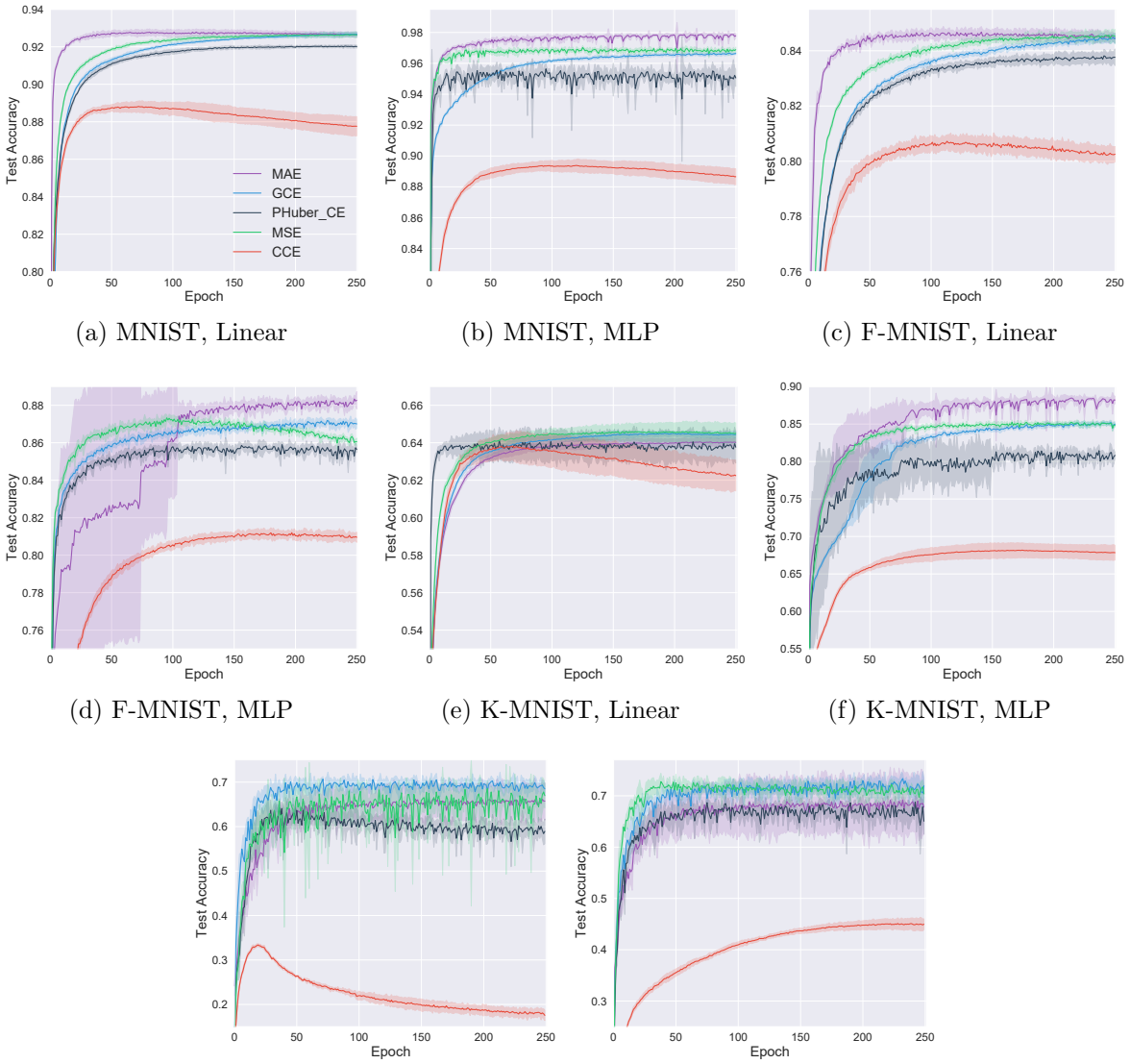
We conduct extensive experiments to evaluate the practical performance of our proposed approaches including the two wrappers, the unbiased risk estimator with various loss functions and the two upper-bound surrogate loss functions.

3.4.1 Datasets

We use five benchmark datasets MNIST [91], Kuzushiji-MNIST [92] (K-MNIST in short), Fashion-MNIST [93] (F-MNIST in short), 20Newsgroups [94], and CIFAR-10 [95], and four datasets from the UCI machine learning repository [96] including Yeast, Texture, Dermatology, and Synthetic Control. We report the characteristics of these datasets as follows:

- MNIST[2] [91]: It is a 10-class dataset of handwritten digits (0 to 9). Each instance is a 28×28 grayscale image.
- Kuzushiji-MNIST[3] [92]: It is a 10-class dataset of cursive Japanese ("Kuzushiji") characters. Each instance is a 28×28 grayscale image.

[2] `http://yann.lecun.com/exdb/mnist/`
[3] `https://github.com/zalandoresearch/fashion-mnist`



(a) MNIST, Linear (b) MNIST, MLP (c) F-MNIST, Linear

(d) F-MNIST, MLP (e) K-MNIST, Linear (f) K-MNIST, MLP

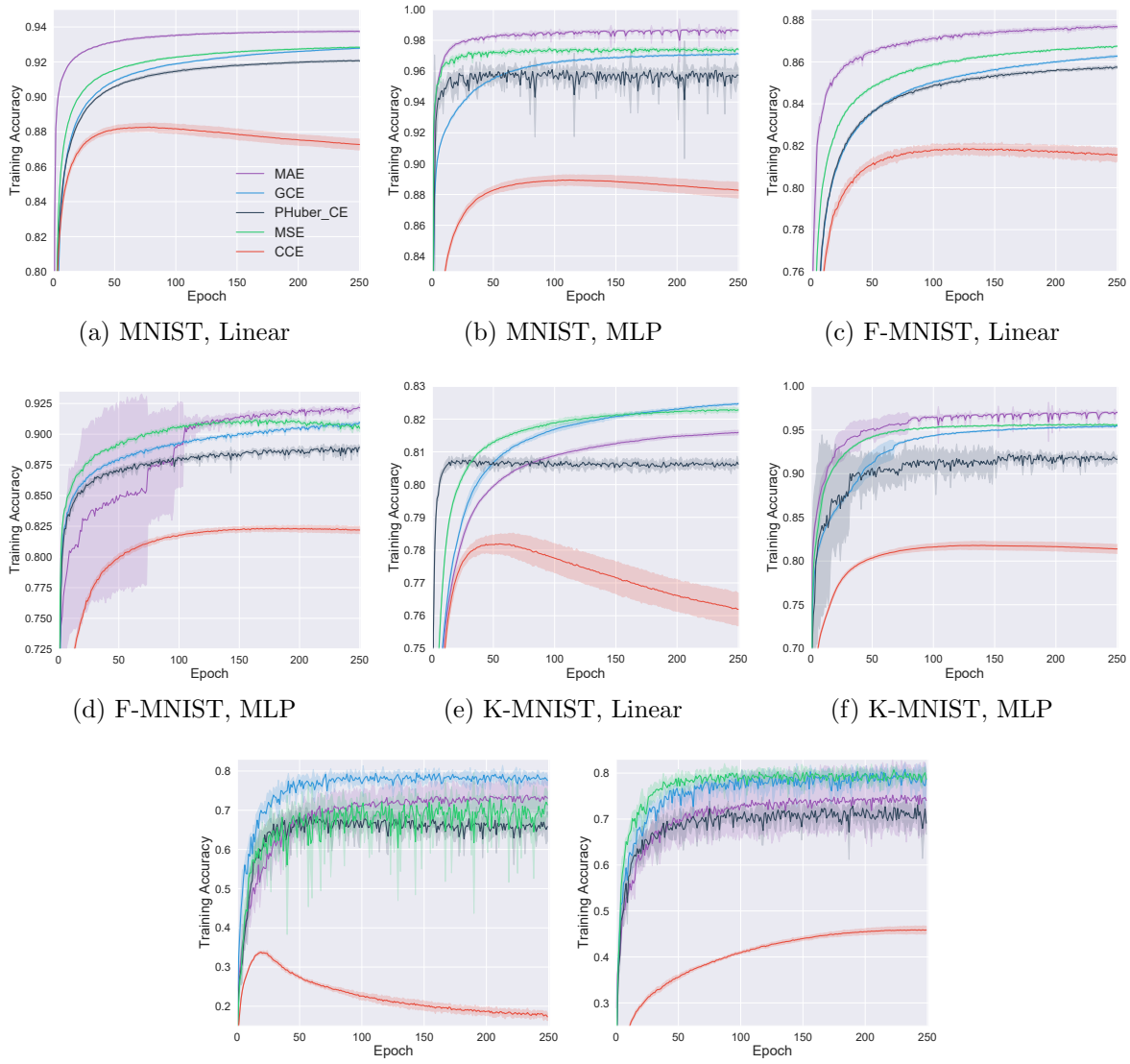
(g) CIFAR-10, ResNet (h) CIFAR-10, DenseNet

Fig. 3.1: Mean test accuracy with standard deviation on different datasets with different loss functions.

- Fashion-MNIST[4] [93]: It is a 10-class dataset of fashion items (T-shirt/top, trouser, pullover, dress, sandal, coat, shirt, sneaker, bag, and ankle boot). Each instance is a 28×28 grayscale image.

- CIFAR-10[5] [95]: It is a 10-class dataset of 10 different objects (airplane, bird,

[4]https://github.com/rois-codh/kmnist
[5]https://www.cs.toronto.edu/~kriz/cifar.html



(a) MNIST, Linear (b) MNIST, MLP (c) F-MNIST, Linear

(d) F-MNIST, MLP (e) K-MNIST, Linear (f) K-MNIST, MLP

(g) CIFAR-10, ResNet (h) CIFAR-10, DenseNet

Fig. 3.2: Mean training accuracy with standard deviation on different datasets with different loss functions.

automobile, cat, deer, dog, frog, horse, ship, and truck). Each instance is a 32×32×3 colored image in RGB format. This dataset is normalized with mean $(0.4914, 0.4822, 0.4465)$ and standard deviation $(0.247, 0.243, 0.261)$.

- 20Newsgroups[6] [94]: It is a 20-class dataset of 20 different topics. We obtained the tf-idf features, and applied TruncatedSVD [97] to reduce the dimension of each

[6] `http://qwone.com/~jason/20Newsgroups/`

examples to 1000. We randomly sample 90% of the examples from the whole dataset to construct the training set, and the rest 10% forms the test set.

- Yeast, Texture, Dermatology, Synthetic Control[7]: They are all the datasets from the UCI Machine Learning Repository. Since they are all regular-scale datasets, we only apply linear model on them. For each dataset, we randomly sample 90% of the examples from the whole dataset to construct the training set, and the rest 10% forms the test set.

We use four base models including linear model, MLP model (d-500-k), ResNet (34 layers) [98], and DenseNet (22 layers) [99]. The summary of these datasets with the corresponding base models are provided in Table 3.3. To generate MCLs, we instantiate $p(s) = \binom{k}{s}/(2^k - 2)$, $\forall s \in \{1, \ldots, k-1\}$, which means $p(s)$ represents the ratio of the number of label sets whose size is s to the number of all possible label sets. For each instance $\boldsymbol{x}$, we first randomly sample s from $p(s)$, and then uniformly and randomly sample a complementary label set $\overline{Y}$ with size s (i.e., $p(\overline{Y}) = 1/\binom{k-1}{s}$).

Compared Methods

We use an unbounded loss CCE and four bounded losses MAE, MSE, GCE [87], and PHuber-CE [88] in our empirical estimator Eq. (3.11). Besides, two upper-bound loss functions LOG and EXP are also inserted into Eq. (3.13). We also compare with a representative partial-label learning approach CLPL [38], which uses a convex formulation with squared hinge loss. In addition, we absorb the following complementary-label learning approaches in the two wrappers (introduced in Section 3.3.1):

- PC [35]: It utilizes the pairwise comparison strategy with sigmoid loss to learn from complementarily labeled data.

- Forward [62]: It conducts forward correction by estimating the latent class transition probability matrix to learn from complementarily labeled data.

[7] `https://archive.ics.uci.edu/ml/datasets.php`

Table 3.4: Classification accuracy (mean±std of 5 trials) of each method on the four UCI datasets using a linear model. The best performance among all the methods is highlighted in bold. Besides, •/∘ denotes whether the performance of our proposed method (the best of EXP and LOG) is statistically (paired t-test at 0.05 significance level) better/worse than the compared method.

Approach		Yeast	Texture	Dermatology	Synthetic Control
Upper-bound Losses	EXP	54.94±1.56%•	97.51±0.09%•	98.89±0.37%	27.87±5.13%•
	LOG	**60.11±1.93%**	98.88±0.43%	**99.46±1.14%**	**90.73±4.41%**
Bounded Losses	MAE	33.07±0.37%•	85.29±7.93%•	85.39±2.58%•	23.50±2.44%•
	MSE	58.17±1.52%•	97.59±0.16%•	97.84±1.21%•	34.20±8.69%•
	GCE	57.56±1.56%•	97.25±0.31%•	97.53±1.81%•	23.67±3.10%•
	Phuber-CE	55.54±1.03%•	94.89±3.28%•	95.14±2.41%•	24.71±3.18%•
Unbounded Loss	CCE	49.50±3.58%•	92.08±1.15%•	83.19±3.65%•	63.47±6.91%•
Decomposition before Shuffle	GA	27.91±5.02%•	90.93±1.34%•	36.05±9.79%•	18.12±1.74%•
	NN	32.73±3.59%•	96.29±0.39%•	61.49±6.83%•	55.12±4.43%•
	FREE	35.50±2.79%•	94.36±1.08%•	86.30±5.62%•	76.95±3.26%•
	PC	53.89±3.53%•	92.68±0.81%•	96.27±3.07%•	72.63±5.86%•
	Forward	58.15±1.54%•	**98.95±0.17%**	99.37±0.85%	38.77±6.06%•
Decomposition after Shuffle	GA	28.21±1.53%•	83.66±2.27%•	42.05±7.94%•	25.46±1.28%•
	NN	36.04±2.24%•	93.91±0.40%•	62.54±9.19%•	59.80±5.14%•
	FREE	43.47±1.36%•	93.94±0.72%•	86.22±6.07%•	73.33±2.17%•
	PC	54.58±2.57%•	94.19±1.21%•	95.73±3.33%•	69.53±9.01%•
	Forward	59.46±1.16%	97.65±0.32%•	99.03±1.33%	43.57±5.83%•
Partial Label Convex Formulation	CLPL	55.39±1.21%•	92.07±0.88%•	99.42±0.54%	63.57±5.46%•

- FREE, NN, GA [36]: These are three approaches for learning from complementarily labeled data. For the FREE method, it is based on an unbiased risk estimator that is loss assumption free. For the NN method, it corrects the negative risk using max operator. For the GA method, it uses a gradient ascent strategy to prevent from overfitting.

For all the approaches, we adopt the same base model for fair comparison. Learning rate and weight decay are selected from $\{10^{-6}, 10^{-5}, \cdots, 10^{-1}\}$. We implement our approach using PyTorch [100], and use the Adam [101] optimization method with mini-batch size set to 256 and epoch number set to 250. Hyper-parameters for all the approaches are selected so as to maximize the accuracy on a validation set (10% of the training set) of complementarily labeled data. All the experiments are conducted on NVIDIA Tesla V100 GPUs.

3.4.2 Experimental Results

Loss Comparison. Fig. 3.1 shows the mean test accuracy with standard deviation of 5 trials, for bounded loss functions MAE, MSE, GCE, PHuber-CE, and unbounded loss

Table 3.5: Classification accuracy (mean±std of 5 trials) of each method on the four benchmark datasets using a linear model. The best performance among all the methods is highlighted in bold. Besides, •/◦ denotes whether the performance of our proposed method (the best of EXP and LOG) is statistically (paired t-test at 0.05 significance level) better/worse than the compared method.

Approach		MNIST	Kuzushiji	Fashion	20Newsgroups
Upper-bound Losses	EXP	**92.67±0.11%**	64.23±0.33%•	**84.56±0.25%**	81.72±0.39%•
	LOG	92.58±0.09%•	**68.89±0.25%**	84.42±0.16%	**84.06±0.57%**
Bounded Losses	MAE	92.66±0.12%	64.03±0.19%•	84.50±0.16%	79.68±1.40%•
	MSE	92.64±0.13%	64.51±0.55%•	84.53±0.20%	81.55±0.52%•
	GCE	92.66±0.12%	64.44±0.17%•	84.44±0.15%	81.78±0.60%•
	Phuber-CE	92.02±0.07%•	63.81±0.75%•	83.76±0.22%•	73.52±1.04%•
Unbounded Loss	CCE	88.23±0.19%•	62.27±0.84%•	80.25±0.29%•	63.78±0.79%•
Decomposition before Shuffle	GA	85.51±0.26%•	55.61±0.24%•	78.64±0.33%•	76.64±0.62%•
	NN	88.09±0.16%•	60.54±0.23%•	80.68±0.07%•	76.00±0.37%•
	FREE	89.35±0.14%•	65.21±0.45%•	81.22±0.11%•	68.34±0.72%•
	PC	88.21±0.23%•	62.76±0.40%•	80.60±0.18%•	66.91±1.20%•
	Forward	92.57±0.05%•	63.51±0.22%•	84.38±0.20%	74.69±1.14%•
Decomposition after Shuffle	GA	83.16±0.22%•	56.31±0.42%•	73.37±0.10%•	66.14±0.79%•
	NN	88.79±0.26%•	63.19±0.12%•	79.77±0.14%•	66.35±0.53%•
	FREE	89.02±0.22%•	64.18±0.18%•	80.11±0.04%•	66.16±0.60%•
	PC	87.76±0.17%•	61.64±0.38%•	80.58±0.17%•	65.64±0.81%•
	Forward	92.54±0.04%•	63.69±0.14%•	84.37±0.17%•	71.98±3.41%•
Partial Label Convex Formulation	CLPL	81.85±0.27%•	55.31±0.23%•	77.26±0.10%•	81.48±0.45%•

function CCE used in our empirical risk estimator in Eq. (3.11). We also record the mean training accuracy with standard deviation (the training set is evaluated with ordinary labels) of 5 trials, and report the results in Fig. 3.2. As can be seen from Fig. 3.1 and Fig. 3.2, all the bounded losses are significantly better than the unbounded loss CCE in our formulation. This observation clearly accords with our discussion on the over-fitting issue in Section 3.3.3. In addition, MAE achieves comparable performance compared with other bounded losses in most cases, while it is sometimes inferior to other bounded losses due to its optimization issue [87]. Both the advantage and disadvantage of MAE motivate us to use the upper-bound loss functions EXP and LOG for improving the practical performance.

Performance Comparison. Table 3.4, Table 3.5, and Table 3.6 show the experimental results of different approaches using a linear model or neural networks on the four UCI datasets and the other five benchmark datasets. In table 3.6, "CIFAR-10 R" and "CIFAR-10 D" mean that we use ResNet and DenseNet on CIFAR-10. Note that CLPL is a convex approach for partial-label learning, which is specially designed with a linear model. Hence CLPL does not appear in Table 3.6. From the three tables, we can find that

Table 3.6: Classification accuracy (mean±std of 5 trials) of each method on the five benchmark datasets using neural networks. The best performance among all the methods is highlighted in bold. Besides, •/∘ denotes whether the performance of our proposed method (the best of EXP and LOG) is statistically (paired t-test at 0.05 significance level) better/worse than the compared method.

Approach		MNIST	Kuzushiji	Fashion	CIFAR-10 R	CIFAR-10 D	20Newsgroups
Upper-bound Losses	EXP	97.80±0.06%	**88.25±0.28%**	88.07±0.19%•	72.49±0.84%•	75.53±0.58%	77.22±1.22%
	LOG	**97.86±0.13%**	88.24±0.08%	**88.36±0.26%**	**75.38±0.34%**	**75.80±0.62%**	**79.46±0.94%**
Bounded Losses	MAE	97.81±0.04%	88.11±0.40%	88.13±0.23%	65.57±4.08%•	68.24±5.84%•	49.83±4.01%•
	MSE	96.84±0.08%•	84.97±0.23%•	86.14±0.04%•	63.58±1.19%•	70.89±0.81%•	72.19±0.59%•
	GCE	96.62±0.08%•	85.02±0.26%•	87.03±0.20%•	68.40±1.05%•	71.54±0.83%•	74.96±0.47%•
	Phuber-CE	95.00±0.36%•	80.66±0.32%•	85.52±0.18%•	59.64±1.21%•	66.49±0.67%•	62.63±2.32%•
Unbounded Loss	CCE	88.64±0.50%•	67.86±1.01%•	80.97±0.23%•	18.01±0.63%•	44.94±1.20%•	54.96±0.38%•
Decomposition before Shuffle	GA	96.36±0.05%•	84.35±0.22%•	85.59±0.30%•	69.05±0.83%•	65.38±1.40%•	79.06±0.57%
	NN	96.70±0.08%•	82.21±0.36%•	86.29±0.10%•	63.85±0.74%•	64.80±1.28%•	76.81±0.44%•
	FREE	88.55±0.38%•	70.32±0.80%•	81.17±0.36%•	32.02±1.69%•	39.22±0.43%•	61.22±1.24%•
	PC	92.74±0.17%•	73.18±0.59%•	83.32±0.28%•	43.16±2.21%•	49.53±1.18%•	65.15±2.05%•
	Forward	97.67±0.04%•	87.65±0.24%•	88.08±0.24%•	71.92±1.09%•	71.30±1.16%•	77.19±%0.76•
Decomposition after Shuffle	GA	92.08±0.22%•	74.64±0.67%•	79.73±0.19%•	53.12±0.97%•	56.51±0.89%•	63.37±1.16%•
	NN	92.47±0.14%•	73.88±0.63%•	82.99±0.13%•	36.79±0.78%•	53.78±0.92%•	65.15±0.73%•
	FREE	88.99±0.39%•	70.09±0.74%•	81.74±0.23%•	15.16±2.22%•	47.45±0.98%•	50.86±1.56%•
	PC	92.94±0.05%•	68.60±1.32%•	82.46±0.26%•	33.16±0.92%•	52.23±0.88%•	64.32±0.86%•
	Forward	97.49±0.08%•	86.47±0.39%•	87.56±0.14%•	72.16±0.97%•	75.23±1.02%	79.35±0.82%

equipped with the two wrappers "Decomposition before Shuffle" and "Decomposition after Shuffle", ordinary complementary-label learning approaches work well for learning with MCLs. However, they are significantly outperformed by the upper-bound losses in most cases, which also achieve the best performance among all the approaches on various benchmark datasets. Besides, bounded losses achieve satisfied performance, while they are also inferior to the two upper-bound losses. We have discussed that bounded losses are better than the unbounded loss CCE because of the negative empirical risk issue. Here, we further discuss why upper-bound surrogate losses (e.g., EXP and LOG) are better than bounded losses. Let $f_y(\boldsymbol{x})$ be the score of the classifier for label y of instance $\boldsymbol{x}$ before taking the softmax operation. That is, $p_\theta(y|\boldsymbol{x}) = \exp f_y(\boldsymbol{x})/(\exp f_y(\boldsymbol{x}) + C)$, where $C = \sum_{i \neq y} \exp f_i(\boldsymbol{x})$. If we regard C as a constant, all the introduced bounded losses are a monotonically decreasing function as to $f_y(\boldsymbol{x})$. With this fact and the boundedness, we can infer that the gradient of the loss goes to 0 as $f_y(\boldsymbol{x})$ goes to $-\infty$. This implies, if a loss to an example is large, it would be hard to optimize the loss when we use a gradient-based optimization algorithm. In contrast, the LOG loss (one of the proposed surrogate losses) is unbounded so it would overcome this issue. The EXP loss is bounded, so we conjecture that is why EXP is relatively inferior to LOG and is not so better than

Table 3.7: Classification accuracy (mean±std% of five trials) of each method on Kuzushiji-MNIST using a linear model. The best performance among all the methods is highlighted in bold.

Approach		$s=1$	$s=2$	$s=3$	$s=4$	$s=5$	$s=6$	$s=7$	$s=8$
Upper-bound Losses	EXP	**60.87** (±0.38)	**62.73** (±0.58)	**63.53** (±0.30)	**64.03** (±0.38)	**64.55** (±0.41)	65.06 (±0.15)	65.23 (±0.10)	65.65 (±0.08)
	LOG	60.11 (±0.49)	61.57 (±0.15)	62.71 (±0.32)	63.36 (±0.09)	64.01 (±0.13)	65.68 (±0.27)	**69.35** (±0.22)	**70.10** (±0.18)
Bounded Losses	MAE	60.43 (±0.43)	62.71 (±0.45)	63.51 (±0.10)	63.75 (±0.31)	63.94 (±0.38)	64.61 (±0.19)	64.82 (±0.16)	65.10 (±0.16)
	MSE	58.97 (±0.47)	62.07 (±0.54)	63.05 (±0.38)	63.85 (±0.57)	64.47 (±0.43)	64.80 (±0.34)	65.17 (±0.25)	65.43 (±0.10)
	GCE	60.48 (±0.55)	62.71 (±0.65)	63.13 (±0.30)	63.87 (±0.33)	63.91 (±0.30)	64.28 (±0.07)	64.38 (±0.12)	64.33% (±0.06)
	Phuber-CE	52.69 (±4.22)	56.58 (±3.94)	61.10 (±2.58)	62.32 (±1.50)	64.51 (±0.68)	64.93 (±0.52)	65.96 (±0.37)	65.81 (±0.62)
Unbounded Loss	CCE	51.59 (±0.64)	55.98 (±1.26)	59.15 (±1.18)	61.08 (±0.78)	63.19 (±0.54)	65.05 (±0.51)	66.82 (±0.41)	68.23 (±0.21)
Decomposition before Shuffle	GA	51.72 (±1.04)	53.78 (±1.07)	54.58 (±0.87)	54.78 (±0.58)	55.33 (±0.29)	55.67 (±0.31)	55.91 (±0.42)	56.15 (±0.23)
	NN	55.03 (±1.35)	57.68 (±1.29)	58.87 (±1.19)	59.52 (±0.87)	60.41 (±0.59)	60.89 (±0.53)	61.41 (±0.36)	61.62 (±0.09)
	FREE	57.26 (±0.83)	60.69 (±0.96)	62.77 (±0.79)	63.91 (±0.65)	64.54 (±0.55)	**66.21** (±0.56)	67.00 (±0.28)	67.71 (±0.20)
	PC	54.31 (±1.04)	58.11 (±0.87)	60.15 (±0.79)	61.32 (±0.68)	62.56 (±0.59)	63.55 (±0.43)	64.27 (±0.20)	65.16 (±0.18)
	Forward	60.05 (±0.43)	61.53 (±0.31)	62.43 (±0.26)	62.98 (±0.40)	63.48 (±0.34)	63.95 (±0.29)	64.14 (±0.09)	64.27 (±0.16)
Decomposition after Shuffle	GA	51.72 (±1.05)	53.79 (±1.07)	54.59 (±0.85)	54.83 (±0.58)	55.33 (±0.35)	55.67 (±0.31)	55.90 (±0.41)	56.18 (±0.22)
	NN	55.03 (±1.35)	58.58 (±1.11)	60.43 (±1.00)	61.58 (±0.72)	62.99 (±0.49)	64.00 (±0.48)	65.07 (±0.36)	66.08 (±0.10)
	FREE	57.26 (±0.84)	60.32 (±0.94)	62.11 (±0.64)	62.98 (±0.67)	64.30 (±0.47)	65.18 (±0.45)	66.02 (±0.28)	67.02 (±0.18)
	PC	54.31 (±1.04)	57.32 (±0.76)	58.95 (±0.77)	60.17 (±0.83)	61.47 (±0.45)	62.54 (±0.40)	63.53 (±0.22)	64.74 (±0.22)
	Forward	60.02 (±0.44)	61.75 (±0.25)	62.68 (±0.23)	63.19 (±0.28)	63.59 (±0.19)	63.94 (±0.09)	64.18 (±0.14)	64.32 (±0.15)

other bounded losses.

Further Analysis. We also conduct additional experiments to investigate the influence of the variable s on the Kuzushiji-MNIST dataset using both linear model and MLP model. Specifically, we study the case where the size of each complementary label set s is fixed at j (i.e., $p(s = j) = 1$) while increasing j from 1 to $k - 2$. The detailed experimental results are shown in Table 3.7 (linear model is used) and Table 3.8 (MLP model is used). From the two tables, we can find that the (test) classification accuracy of our proposed methods increases as j increases. This observation is clearly in accordance with our derived estimation error bound (Theorem 3.4), as the estimation error would decrease if j increases. In addition, as shown in the two tables, our proposed upper-bound

Table 3.8: Classification classification (mean±std% of 5 trials) of each method on Kuzushiji-MNIST using a MLP model. The best performance among all the methods is highlighted in bold.

Approach		$s=1$	$s=2$	$s=3$	$s=4$	$s=5$	$s=6$	$s=7$	$s=8$
Upper-bound Losses	EXP	71.66 (±3.48)	**82.51** (±3.08)	84.45 (±0.24)	87.10 (±0.37)	88.35 (±0.18)	**89.61** (±0.33)	90.18 (±0.37)	90.92 (±0.15)
	LOG	**77.07** (±3.00)	82.39 (±0.73)	**85.54** (±0.35)	**87.60** (±0.40)	**88.87** (±0.34)	89.25 (±0.37)	**90.22** (±0.31)	**91.19** (±0.11)
Bounded Losses	MAE	69.87 (±1.04)	73.60 (±5.77)	79.97 (±3.71)	85.34 (±2.78)	86.91 (±3.06)	89.10 (±0.46)	90.32 (±0.31)	91.06 (±0.34)
	MSE	57.56 (±0.92)	71.37 (±0.89)	78.26 (±0.49)	82.97 (±0.41)	85.37 (±0.45)	86.82 (±0.13)	88.03 (±0.11)	88.69 (±0.05)
	GCE	63.85 (±1.27)	74.11 (±2.38)	79.18 (±2.31)	83.65 (±0.15)	85.23 (±0.25)	86.32 (±0.27)	87.12 (±0.20)	87.64 (±0.09)
	Phuber-CE	10.24 (±4.09)	14.76 (±2.11)	26.60 (±1.58)	73.43 (±1.50)	81.41 (±0.58)	83.00 (±0.42)	84.69 (±0.47)	85.59 (±0.52)
Unbounded Loss	CCE	56.17 (±0.64)	60.89 (±0.61)	64.18 (±0.77)	66.57 (±0.41)	69.14 (±0.49)	71.63 (±0.31)	74.55 (±0.31)	78.22 (±0.22)
Decomposition before Shuffle	GA	70.25 (±0.24)	76.50 (±0.47)	79.77 (±0.32)	82.03 (±0.22)	84.05 (±0.64)	85.58 (±0.32)	86.40 (±0.24)	87.49 (±0.15)
	NN	65.33 (±0.51)	71.34 (±0.53)	75.46 (±0.31)	78.67 (±0.58)	81.40 (±0.28)	84.08 (±0.16)	86.56 (±0.39)	88.61 (±0.12)
	FREE	53.90 (±1.05)	60.32 (±1.14)	63.98 (±0.85)	66.79 (±0.64)	69.31 (±0.73)	71.65 (±0.73)	74.43 (±0.28)	76.61 (±0.33)
	PC	56.36 (±0.56)	62.37 (±0.50)	66.09 (±0.44)	69.51 (±0.47)	72.46 (±0.35)	75.18 (±0.33)	78.50 (±0.52)	82.40 (±0.38)
	Forward	75.40 (±2.02)	83.19 (±0.61)	85.18 (±0.48)	86.63 (±0.38)	87.51 (±0.29)	88.29 (±0.29)	88.96 (±0.26)	89.41 (±0.25)
Decomposition after Shuffle	GA	70.25 (±0.24)	75.91 (±1.37)	78.46 (±2.84)	80.60 (±3.35)	82.14 (±4.51)	83.48 (±4.92)	84.01 (±5.35)	84.65 (±6.28)
	NN	63.73 (±0.97)	67.26 (±0.82)	69.46 (±0.74)	71.25 (±0.62)	73.15 (±0.45)	74.82 (±0.35)	77.09 (±0.17)	79.39 (±0.21)
	FREE	55.33 (±0.89)	60.81 (±0.97)	64.65 (±0.89)	67.01 (±0.70)	69.60 (±0.78)	71.63 (±0.46)	74.22 (±0.40)	77.16 (±0.50)
	PC	56.68 (±1.28)	61.07 (±0.99)	63.86 (±0.67)	65.61 (±0.44)	68.03 (±0.64)	69.74 (±0.65)	72.49 (±0.37)	75.17 (±0.46)
	Forward	66.09 (±0.49)	73.20 (±3.05)	75.76 (±2.61)	82.53 (±2.60)	86.27 (±0.65)	88.05 (±0.27)	89.24 (±0.22)	90.22 (±0.20)

losses outperform other approaches in most cases, even though different base models are employed. This observation also demonstrates the effectiveness of our proposed upper-bound losses.

3.5 Chapter Summary

In this chapter, we proposed a novel problem setting called *learning with multiple complementary labels* (MCLs), which is a generalization of *complementary-label learning* [35,36]. To solve this learning problem, we first designed two wrappers that enable us to use arbitrary complementary-label learning approaches for learning with MCLs. However, we found that the supervision information that MCLs hold is conceptually diluted after de-

composition. Therefore, we further proposed an unbiased risk estimator for learning with MCLs, which processes each set of MCLs as a whole. Then, we theoretically derived an estimation error bound, which guarantees the learning consistency. Although our risk estimator does not rely on specific models or loss functions, we showed that bounded loss is generally better than unbounded loss in our empirical risk estimator. In addition, we improved the risk estimator into minimizing properly chosen upper bounds for practical implementation. Extensive experiments clearly demonstrated the effectiveness of the proposed approaches.

Chapter 4

Partial-Label Learning with Latent Label Distributions

4.1 Motivation

Partial-label learning is a specific type of weakly supervised learning [38], in which each instance is associated with a set of candidate labels. However, only one of the candidate labels is the ground-truth label, which is concealed in the training process.

To learn from partially labeled examples, the key is how to properly deal with the candidate labels. So far, there exist two major strategies, including the average-based strategy and the identification-based strategy. The average-based strategy basically treats each candidate label equally in the model training [38, 102]. The identification-based strategy considers the correct label as a latent variable hidden in the indiscriminate candidate label set [41, 67, 71, 77, 103]. They all make predictions by aggregating the modeling outputs of the candidate labels without discrimination, while the confidence of each candidate label being the ground-truth label is regrettably ignored. As a consequence, these methods may be suboptimal, since each candidate label normally makes different contributions to the model training.

In this chapter[1], we formalize the different labeling confidence levels of the candidate labels as the latent label distributions, and propose the LALO (partial-label learning with LAtent Label distributiOns) approach. LALO first introduces a novel unified framework that estimates the latent label distributions while training the model simultaneously, and

[1]This chapter has been published in [74].

then presents a biconvex formulation with constrained local consistency, finally adopts an alternating method to solve this optimization problem. On the one hand, the inductive model is discriminatively trained by minimizing the least squares loss of fitting the latent label distributions. On the other hand, the latent label distributions are regularized by the modeling outputs via a constrained label propagation procedure specifically for the PL properties. Through the mutual promotion of the model training and the label propagation, the ground-truth label can be identified by optimally estimating the label distributions. The effectiveness of LALO is validated by experiments on 4 controlled datasets and 5 real-world partially labeled datasets.

4.2 The LALO Method

For each training example, we receive a feature vector $\boldsymbol{x}_i \in \mathbb{R}^d$ and its corresponding label vector $\boldsymbol{y}_i \in \{0,1\}^k$ with k labels. Suppose n denotes the number of training examples, $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{Y} \in \{0,1\}^{n\times k}$ are the instance matrix and label matrix, respectively. In the setting of partial-label learning, $y_{ij} = 1$ means the i-th training sample is assigned the j-th (candidate) label.

Existing partial-label learning frameworks indiscriminately train the model with noise-corrupted label matrix $\boldsymbol{Y} \in \{0,1\}^{n\times k}$, in which the labeling confidence of each candidate label is not discriminated. However, each candidate normally makes different contributions to the model training. To capture the labeling confidence (relative importance) of each candidate label, we propose to train the model with the latent label distributions. Specifically, for a training example $\boldsymbol{x}_i \in \mathbb{R}^d$, its latent label distribution is denoted by $\boldsymbol{p}_i \in [0,1]^k$. By arranging the label distributions of n training examples, we form the label distribution matrix $\boldsymbol{P} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_n]^\top \in [0,1]^{n\times k}$. By substituting $\boldsymbol{Y} \in \{0,1\}^{n\times k}$ with $\boldsymbol{P} \in [0,1]^{n\times k}$, we thus propose a novel unified framework that estimates the latent label distributions while training the model simultaneously:

$$\min \sum_{i=1}^{n} \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{p}_i, f) + \lambda\Omega(f) + \mu\Psi(\boldsymbol{P}), \tag{4.1}$$

where $\mathcal{L}$ denotes the prescribed loss function, Ω controls the complexity of the model f, Ψ tries to obtain an accurate estimation of the label distribution matrix $\boldsymbol{P}$, and λ, μ are the hyper-parameters trading off the three terms.

Unlike the average-based framework and the identification-based framework, our proposed framework naturally treats the modeling outputs of the candidate labels in a discriminative manner due to the label distribution matrix $\boldsymbol{P}$, which can indicate the different contributions of the candidate labels. To optimally estimate $\boldsymbol{P}$, we assume it should have the following property: *local consistency*, i.e., The label distributions of neighboring (similar) instances are probably similar. Specifically, if the i-th instance $\boldsymbol{x}_i$ is similar to the j-th instance $\boldsymbol{x}_j$, their corresponding label distributions $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ should also be similar. In order to characterize the similarity between instances, we construct the similarity matrix $\boldsymbol{S} = [s_{ij}]_{n\times n}$ by the symmetry-favored K-NN graph [104], i.e.,

$$s_{ij} = \begin{cases} \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2}{\sigma^2}\right) & \text{if } j \in \mathcal{N}_i, \\ 0 & \text{otherwise.} \end{cases}$$

Here, the set $\mathcal{N}_i$ saves the indices of the k-nearest neighbors of $\boldsymbol{x}_i$, and the hyper-parameter σ is set to $\sigma = \sum_{i=1}^{n} \left\|\boldsymbol{x}_i - \boldsymbol{x}_{i_K}\right\|_2 / n$ where $\boldsymbol{x}_{i_K}$ denotes the K-th nearest neighbor of $\boldsymbol{x}_i$. To ensure that the similarity matrix $\boldsymbol{S}$ is symmetric, we further set $\boldsymbol{S} = \boldsymbol{S} + \boldsymbol{S}^\top$. In this way, we define $\Psi(\boldsymbol{P})$ as follows:

$$\begin{aligned} \Psi(\boldsymbol{P}) &= \sum_{i,j}^{n} s_{ij} \left\| \frac{\boldsymbol{p}_i}{\sqrt{d_{ii}}} - \frac{\boldsymbol{p}_j}{\sqrt{d_{jj}}} \right\|_2^2 \\ \text{s.t.} \quad & \textstyle\sum_j p_{ij} = 1, \quad \forall i \in [n], \\ & \boldsymbol{0}_{n\times k} \le \boldsymbol{P} \le \boldsymbol{Y}, \end{aligned} \tag{4.2}$$

where $d_{ii} = \sum_j s_{ij}$ is the degree of the vertex $\boldsymbol{x}_i$ in the graph corresponding to the similarity matrix, and $\boldsymbol{0}_{n\times k} \in \{0\}^{n\times k}$, and where $\boldsymbol{0}_{n\times k} \le \boldsymbol{P} \le \boldsymbol{Y}$ means $0 \le p_{ij} \le y_{ij}, \forall i \in [n], \forall j \in [k]$. The above regularization is adapted from manifold regularization [105], which promotes that similar examples should have similar label distributions. The first constraint formalizes the labeling confidence levels of all the labels as label distributions. The second one guarantees that the correct label is always a candidate label, and the confidence of each non-candidate label is always fixed at zero. Following the above settings, we propose to train the model by minimizing the least squares loss of fitting the label distributions:

$$L(\boldsymbol{x}_i, \boldsymbol{p}_i, f) = \left\|\boldsymbol{W}^\top \phi(\boldsymbol{x}_i) + \boldsymbol{b} - \boldsymbol{p}_i\right\|_2^2, \tag{4.3}$$

where $\phi(\cdot) : \mathbb{R}^d \to \mathbb{R}^h$ is a feature mapping that maps the feature space to some higher (maybe infinite) dimensional Hilbert space with h dimensions, $\boldsymbol{W} \in \mathbb{R}^{h\times k}$ and $\boldsymbol{b} \in \mathbb{R}^k$ are the model parameters. It is worth noting that the feature mapping is introduced to enables a kernel extension for dealing with the non-linearity conveyed by the data. For the regularization term to control the model complexity, we adopt the widely used squared Frobenius norm:

$$\Omega(f) = \|\boldsymbol{W}\|_F^2 . \tag{4.4}$$

Finally, we present the whole formulation as a constrained optimization problem:

$$\begin{aligned} \min_{\boldsymbol{W},\boldsymbol{b},\boldsymbol{P}} \sum_i \|\boldsymbol{e}_i\|_2^2 + \lambda \|\boldsymbol{W}\|_F^2 + \mu \sum_{i,j}^n s_{ij} \left\| \frac{\boldsymbol{p}_i}{\sqrt{d_{ii}}} - \frac{\boldsymbol{p}_j}{\sqrt{d_{jj}}} \right\|_2^2 \\ \text{s.t.} \quad \boldsymbol{p}_i = \boldsymbol{W}^\top \boldsymbol{z}_i + \boldsymbol{b} + \boldsymbol{e}_i, \quad \forall i \in [n], \\ \textstyle\sum_j p_{ij} = 1, \quad \forall i \in [n], \\ \boldsymbol{0}_{n\times k} \le \boldsymbol{P} \le \boldsymbol{Y}, \end{aligned} \tag{4.5}$$

where $\boldsymbol{z}_i = \phi(\boldsymbol{x}_i)$. In this formulation, we aim to estimate the latent label distributions while training the target model simultaneously.

4.3 Alternating Optimization

Obviously, the optimization problem (4.5) is a biconvex problem [106], and we solve this problem in an alternating way. Specifically, we first optimize the objective function with respective to $\boldsymbol{W}$ and $\boldsymbol{b}$ when $\boldsymbol{P}$ is fixed, and then optimize the objective function with respective to $\boldsymbol{P}$ when $\boldsymbol{W}$ and $\boldsymbol{b}$ are both fixed. Such optimization process is repeated until convergence.

4.3.1 Updating Model Parameters

When $\boldsymbol{P} \in \mathbb{R}^{n\times k}$ is fixed, the optimization problem (4.5) with respective to $\boldsymbol{W}$ and $\boldsymbol{b}$ can be stated as follows:

$$\begin{aligned} \min_{\boldsymbol{W},\boldsymbol{b}} \operatorname{tr}(\boldsymbol{\Xi}^\top \boldsymbol{\Xi}) + \lambda \cdot \operatorname{tr}(\boldsymbol{W}^\top \boldsymbol{W}) \\ \text{s.t.} \quad \boldsymbol{P} = \boldsymbol{Z}\boldsymbol{W} + \boldsymbol{1}_n \boldsymbol{b}^\top + \boldsymbol{\Xi}, \end{aligned} \tag{4.6}$$

where $\boldsymbol{\Xi} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_n]^\top \in \mathbb{R}^{n\times k}$, $\mathrm{tr}(\cdot)$ is the trace norm operator with the property $\mathrm{tr}(\boldsymbol{W}^\top\boldsymbol{W}) = \|\boldsymbol{W}\|_F^2$, and $\mathbf{1}_n = [1, 1, \cdots, 1]^\top \in \mathbb{R}^n$. Then, the Lagrangian of this problem can be expressed as:

$$\mathcal{L}(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\Xi}, \boldsymbol{A}) = \mathrm{tr}(\boldsymbol{\Xi}^\top\boldsymbol{\Xi}) + \lambda \cdot \mathrm{tr}(\boldsymbol{W}^\top\boldsymbol{W}) - \mathrm{tr}(\boldsymbol{A}^\top(\boldsymbol{Z}\boldsymbol{W} + \mathbf{1}_n\boldsymbol{b}^\top + \boldsymbol{\Xi} - \boldsymbol{P})), \tag{4.7}$$

where $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_n]^\top \in \mathbb{R}^{n\times k}$ is the matrix that stores the Lagrange multipliers. In this way, the following equations will be induced according to the KKT conditions:

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial\boldsymbol{\Xi}} = 0 &\Rightarrow \boldsymbol{A} = 2\boldsymbol{\Xi},\\
\frac{\partial\mathcal{L}}{\partial\boldsymbol{A}} = 0 &\Rightarrow \boldsymbol{Z}\boldsymbol{W} + \mathbf{1}_n\boldsymbol{b}^\top + \boldsymbol{\Xi} = \boldsymbol{P},\\
\frac{\partial\mathcal{L}}{\partial\boldsymbol{W}} = 0 &\Rightarrow \boldsymbol{W} = \frac{1}{2\lambda}\boldsymbol{Z}^\top\boldsymbol{A},\\
\frac{\partial\mathcal{L}}{\partial\boldsymbol{b}} = 0 &\Rightarrow \boldsymbol{A}^\top\mathbf{1}_n = \mathbf{0}_k.
\end{aligned}$$

The above linear equations can be solved by following steps:

$$\begin{aligned}
\boldsymbol{Z}\boldsymbol{W} + \mathbf{1}_n\boldsymbol{b}^\top + \boldsymbol{\Xi} &= \boldsymbol{P}\\
\frac{1}{2\lambda}\boldsymbol{Z}\boldsymbol{Z}^\top\boldsymbol{A} + \mathbf{1}_n\boldsymbol{b}^\top + \frac{1}{2}\boldsymbol{A} &= \boldsymbol{P}.
\end{aligned} \tag{4.8}$$

Here, we define the positive definite matrix $\boldsymbol{H}$ by $\boldsymbol{H} = \frac{1}{2\lambda}\boldsymbol{K} + \frac{1}{2}\boldsymbol{I}_{n\times n}$ and $\boldsymbol{K} = \boldsymbol{Z}\boldsymbol{Z}^\top \in \mathbb{R}^{n\times n}$ is given by its elements $k_{ij} = \boldsymbol{\phi}(\boldsymbol{x}_i)\boldsymbol{\phi}(\boldsymbol{x}_j)^\top = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, where $\mathcal{K}(\cdot,\cdot)$ is the kernel function. For LALO, Gaussian kernel $\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2/(2\sigma'^2))$ is employed with σ' set to the average distance of all pairs of training examples. The matrix $\boldsymbol{I}_{n\times n}$ is an $n \times n$ identity matrix. Then we can obtain:

$$\begin{aligned}
\boldsymbol{H}\boldsymbol{A} + \mathbf{1}_n\boldsymbol{b}^\top &= \boldsymbol{P}\\
\boldsymbol{A} + \boldsymbol{H}^{-1}\mathbf{1}_n\boldsymbol{b}^\top &= \boldsymbol{H}^{-1}\boldsymbol{P}\\
\mathbf{1}_n^\top\boldsymbol{H}^{-1}\mathbf{1}_n\boldsymbol{b}^\top &= \mathbf{1}_n^\top\boldsymbol{H}^{-1}\boldsymbol{P}\\
\boldsymbol{b}^\top &= \frac{\mathbf{1}_n^\top\boldsymbol{H}^{-1}\boldsymbol{P}}{\mathbf{1}_n^\top\boldsymbol{H}^{-1}\mathbf{1}_n}.
\end{aligned}$$

For computational convenience, we define $\boldsymbol{s} = \mathbf{1}_n^\top\boldsymbol{H}^{-1} \in \mathbb{R}^{1\times n}$, and the results are reported as follows:

$$\boldsymbol{b}^\top = \frac{\boldsymbol{s}\boldsymbol{P}}{\boldsymbol{s}\mathbf{1}_n}, \tag{4.9}$$

$$\boldsymbol{A} = \boldsymbol{H}^{-1}(\boldsymbol{P} - \mathbf{1}_n\boldsymbol{b}^\top). \tag{4.10}$$

4.3.2 Updating Latent Label Distributions

When $\boldsymbol{W}$ and $\boldsymbol{b}$ are fixed, the modeling output matrix $\boldsymbol{Q} \in \mathbb{R}^{n\times k}$ can be denoted by $\boldsymbol{Q} = \boldsymbol{Z}\boldsymbol{W} + \mathbf{1}_n\boldsymbol{b}^\top = \frac{1}{2\lambda}\boldsymbol{K}\boldsymbol{A} + \mathbf{1}_n\boldsymbol{b}^\top$, then $\boldsymbol{\Xi} = \boldsymbol{P} - \boldsymbol{Q}$. By eliminating $\boldsymbol{\Xi}$, we can obtain:

$$\begin{aligned} \min_{\boldsymbol{P}} \|\boldsymbol{P}-\boldsymbol{Q}\|_F^2 + \mu \sum_{i,j}^{n} s_{ij} \left\| \frac{\boldsymbol{p}_i}{\sqrt{d_{ii}}} - \frac{\boldsymbol{p}_j}{\sqrt{d_{jj}}} \right\|_2^2 \\ \text{s.t.} \quad \textstyle\sum_j p_{ij} = 1, \quad \forall i \in [n], \\ \mathbf{0}_{n\times k} \le \boldsymbol{P} \le \boldsymbol{Y}. \end{aligned} \tag{4.11}$$

Here, this optimization problem is actually a constrained label propagation problem [83], where μ specifies the relative amount of labeling information from the neighbor points and the modeling outputs. The first constraint guarantees that a label distribution is consistently assigned to each instance in the process of label propagation. The second constraint guarantees that labels are only propagated among candidate labels. While in semi-supervised settings [7], labels are normally propagated from labeled examples to unlabeled examples. In addition, traditional label propagation problems normally treat the observed label matrix $\boldsymbol{Y}$ as the initial label matrix. In contrast, since the observed label matrix $\boldsymbol{Y}$ is a noise-corrupted version in partial-label learning, we take the modeling output matrix $\boldsymbol{Q}$ as the initial label matrix for each optimization iteration, thereby adjusting the confidence level of each candidate label iteratively. The optimization problem (4.11) can be reformulated as a standard Quadratic Programming (QP) problem, which can be solved by any off-the-shelf QP tools.

Concretely, in order to solve problem (4.11), we introduce $\widetilde{\boldsymbol{p}} = \text{vectorize}(\boldsymbol{P}) \in [0,1]^{nk}$ where vectorize$(\cdot)$ aims to transform the matrix to a vector. Similarly, $\widetilde{\boldsymbol{q}} = \text{vectorize}(\boldsymbol{Q}) \in \mathbb{R}^{nk}$ and $\widetilde{\boldsymbol{y}} = \text{vectorize}(\boldsymbol{Y}) \in \{0,1\}^{nk}$. To deal with the equality constraint using $\widetilde{\boldsymbol{p}}$, we pick up the indices of $\widetilde{\boldsymbol{p}}$ by defining a set $\mathcal{C} = \{\mathcal{C}_0, \mathcal{C}_1, \cdots, \mathcal{C}_{n-1}\}$ as follows:

$$j \in \mathcal{C}_i \quad \text{if} \quad j\%n = i, \forall j \in [nk],$$

Using these notations, problem (4.11) can be written as:

$$\begin{aligned} \min_{\widetilde{\boldsymbol{p}}} \frac{1}{2}\widetilde{\boldsymbol{p}}^\top \widetilde{\boldsymbol{H}} \widetilde{\boldsymbol{p}} + \widetilde{\boldsymbol{f}}^\top \widetilde{\boldsymbol{p}} \\ \text{s.t.} \quad \textstyle\sum_{j\in\mathcal{C}_i} \widetilde{p}_j = 1, \quad \forall \mathcal{C}_i \subseteq \mathcal{C}, \\ \mathbf{0}_{nk} \le \widetilde{\boldsymbol{p}} \le \widetilde{\boldsymbol{y}}, \end{aligned} \tag{4.12}$$

Algorithm 1 The LALO Algorithm

Inputs:

$\mathcal{D}$: the partially labeled training set
k: the number of nearest neighbors used for the similarity matrix
λ, μ: the hyper-parameters trading off each term in the loss function

Process:

construct the similarity matrix by the symmetry-favored K-NN graph;
calculate the kernel matrix $\boldsymbol{K} = [\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{n\times n}$;
initialize $\boldsymbol{P}$ according to Eq. (4.13);
repeat
    update $\boldsymbol{b}$ and $\boldsymbol{A}$ according to Eq. (4.9) and Eq. (4.10);
    update $\boldsymbol{Q} = \frac{1}{2\lambda}\boldsymbol{K}\boldsymbol{A} + \mathbf{1}_n\boldsymbol{b}^\top$;
    calculate $\widetilde{\boldsymbol{p}}$ by solving (4.12) with a general QP procedure;
    update $\boldsymbol{P}$ by reshaping $\widetilde{\boldsymbol{p}} \in \mathbb{R}^{nk}$ into $\boldsymbol{P} \in \mathbb{R}^{n\times k}$;
until convergence.
return the predicted label y according to (4.14).

Output:

y: the predicted label of a test instance

where $\widetilde{\boldsymbol{f}} = -2\widetilde{\boldsymbol{q}}$, and $\widetilde{\boldsymbol{H}} \in \mathbb{R}^{nk\times nk}$ is defined as follows:

$$\widetilde{\boldsymbol{H}} = \begin{bmatrix} \widetilde{\boldsymbol{T}} & \mathbf{0}_{n\times n} & \cdots & \mathbf{0}_{n\times n} \\ \mathbf{0}_{n\times n} & \widetilde{\boldsymbol{T}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0}_{n\times n} \\ \mathbf{0}_{n\times n} & \cdots & \mathbf{0}_{n\times n} & \widetilde{\boldsymbol{T}} \end{bmatrix}.$$

Here, $\widetilde{\boldsymbol{T}}$ is a square matrix defined by $\widetilde{\boldsymbol{T}} = 2((\mu+1)\boldsymbol{I}_{n\times n} - \mu\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{S}\boldsymbol{D}^{-\frac{1}{2}}) \in \mathbb{R}^{n\times n}$ where $\boldsymbol{D}$ is a diagonal matrix with its diagonal element defined by $d_{ii} = \sum_j s_{ij}$. In this way, the optimization problem (4.12) can be efficiently tackled by any existing QP toolbox.

At the beginning of the alternating optimization, we initialize the label distribution matrix $\boldsymbol{P} = [p_{ij}]_{n\times k}$ as follows:

$$p_{ij} = \begin{cases} \frac{1}{\sum_j y_{ij}} & \text{if } y_{ij} = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{4.13}$$

After $\widetilde{\boldsymbol{p}}$ is obtained, we can easily obtain the label distribution matrix $\boldsymbol{P}$ by reshaping $\widetilde{\boldsymbol{p}} \in \mathbb{R}^{nk}$ into $\boldsymbol{P} \in \mathbb{R}^{n\times k}$. After the completion of the optimization process, the predicted

Table 4.1: Characteristics of the used datasets.

	Controlled Datasets				Real-World Datasets				
Dataset	glass	usps	letter	deter	Lost	FG-NET	MSRCv2	Soccer Player	Yahoo! News
Examples	214	9298	20000	358	1122	1002	1758	17472	22991
Features	10	256	16	23	108	262	48	279	163
Classes	5	10	26	6	16	78	32	171	219
Avg. CLs	-	-	-	-	2.23	7.48	3.16	2.09	1.91

label $\hat{y}$ of the test instance $\boldsymbol{x}$ would be

$$\hat{y} = \arg\max_{j \in [k]} \sum_{i=1}^{n} a_{ij} \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_i) + b_j. \tag{4.14}$$

The pseudo code of LALO is presented in Algorithm 1. Since the proposed formulation (4.5) is biconvex, it can be solved by the alternating optimization method with guaranteed convergence [106], and we set the maximum number of iterations as 50.

4.4 Experiments

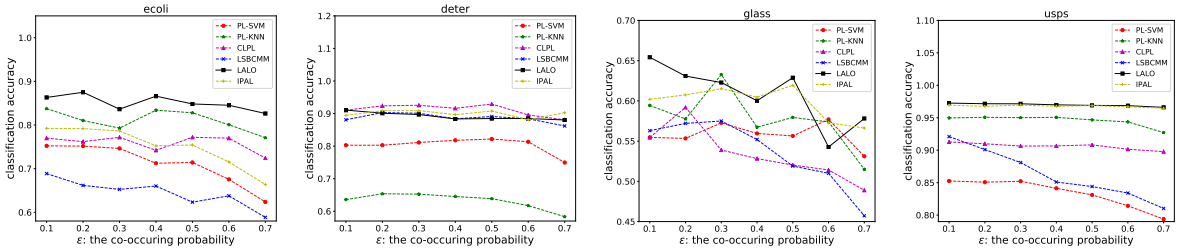
In this section, we conduct extensive experiments on artificial (controlled) and real-world datasets to validate the effectiveness of LALO. The detailed information of the used datasets is reported in Table 4.1.

4.4.1 Experimental Setup

Following the widely-used controlling protocol [38, 41, 67, 70, 72, 102, 103], we also use three controlling parameters p, r and ϵ to generate artificial PL datasets. Concretely, p controls the proportion of training examples that are partially labeled, r decides the number of incorrect labels in each candidate label set, and ϵ controls the probability of an incorrect candidate label appearing with the correct label. We use four benchmark datasets including glass, ecoli, deter, and usps. The first three datasets are collected from the UCI repository [96], and the last dataset is collected from the website of the late Sam Roweis[2].

In addition, we have also collected 5 real-world partially labeled datasets, including Soccer Player [39], Lost [38], Yahoo! News [107], FG-NET [108], and MSCRCv2 [41].

[2] `https://cs.nyu.edu/~roweis/data.html`



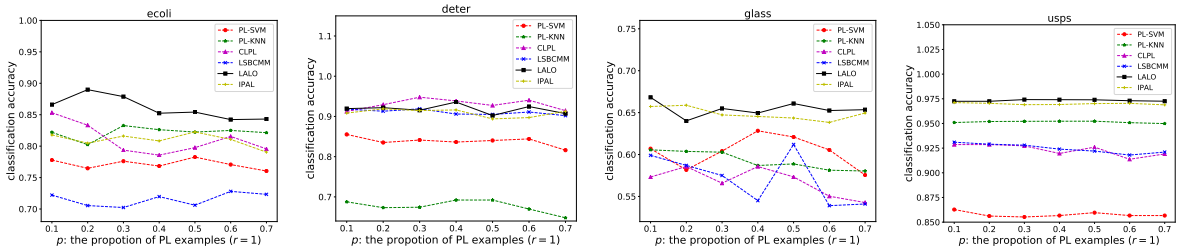
(a) ecoli (b) deter (c) glass (d) usps

Fig. 4.1: Classification accuracy on controlled datasets with ϵ ranging from 0.1 to 0.7 ($p = 1, r = 1$).

These real-world datasets come from several application domains. For *automatic face naming* (Lost, Soccer Player and Yahoo! News), each face cropped from an image or a video frame is considered as an instance and the names extracted from the corresponding image captions or video subtitles work as candidate labels. For *objective classification* (MSRCv2), image segments are considered as instances and objects appearing in the same image work as candidate labels. For *facial age estimation* (FG-NET), each human face is represented as an instance, and the age annotations obtained by crowdsouring are candidate labels. Besides, the average number of the candidate labels (Avg. CLs) for each real-world dataset is also recorded in Table 4.1.

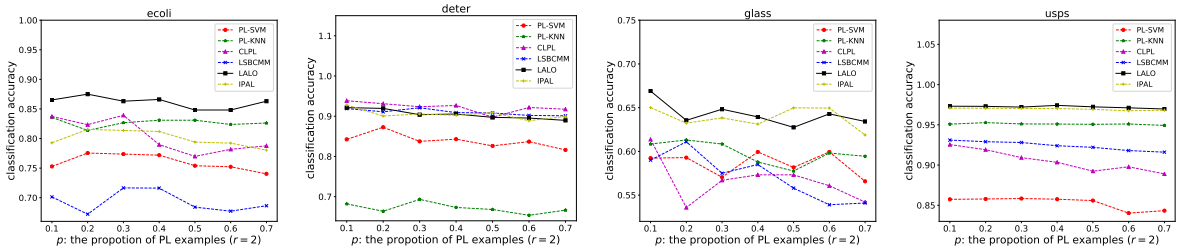
The performance of LALO is compared with five state-of-the-art partial-label learning algorithms, each configured with recommended parameters according to the respective papers:

- PL-KNN [66]: an K-nearest neighbor approach following the average-based framework. (Recommended configuration: $K = 10$).
- CLPL [38]: a parametric approach following the average-based framework. (Recommended configuration: SVM with squared hinge loss).
- IPAL [72]: an instance-based approach following the average-based framework. (Recommended configuration: $\alpha = 0.95, K = 10, T = 100$).
- PL-SVM [77]: a maximum margin approach following the identification-based framework. (Recommended configuration: $\lambda \in \{10^{-3}, 10^{-2}, \cdots, 10^{3}\}$).



(a) ecoli (b) deter (c) glass (d) usps

Fig. 4.2: Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 1$).



(a) ecoli (b) deter (c) glass (d) usps

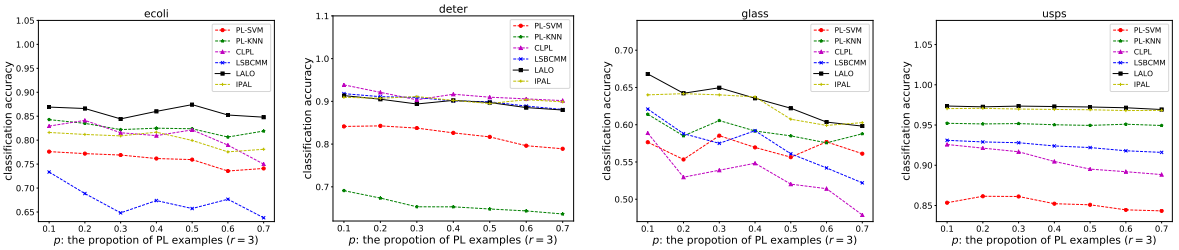
Fig. 4.3: Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 2$).

- LSB-CMM [41]: a maximum likelihood approach following the identification-based framework. (Recommended configuration: k mixture components).

The hyper-parameters employed by LALO are set as $K = 10, \lambda = 0.05, \mu = 0.005$. We present sensitivity analysis of LALO in Subsection 5.3. On each artificial and real-world dataset, ten runs of 50%/50% random train/test splits are performed, and the averaged accuracy (with standard deviation) is reported for each method. In addition, we use the t-test at 0.05 significance level for two independent samples to investigate whether LALO is significantly superior/inferior to the compared algorithms for all experiments.

4.4.2 Experimental Results

Controlled Datasets. Fig. 4.1 reports the classification accuracy of each algorithm as ϵ ranges from 0.1 to 0.7 with step size 0.1 when p and r are both fixed at 1. For each



(a) ecoli (b) deter (c) glass (d) usps

Fig. 4.4: Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 3$).

Table 4.2: Win/tie/loss counts on the controlled datasets between LALO and the compared algorithms.

	PL-KNN	CLPL	IPAL	PL-SVM	LSB-CMM
vary ϵ $(p, r = 1)$	25/2/1	21/2/5	16/12/0	26/2/0	23/5/0
vary p $(r = 1)$	26/2/0	24/1/3	13/15/0	27/1/0	23/5/0
vary p $(r = 2)$	26/2/0	21/4/3	11/16/1	28/0/0	21/5/2
vary p $(r = 3)$	24/4/0	21/2/5	23/12/3	27/1/0	21/7/0
In Total	**101/10/1**	**87/9/16**	**63/65/4**	**108/4/0**	**88/22/2**

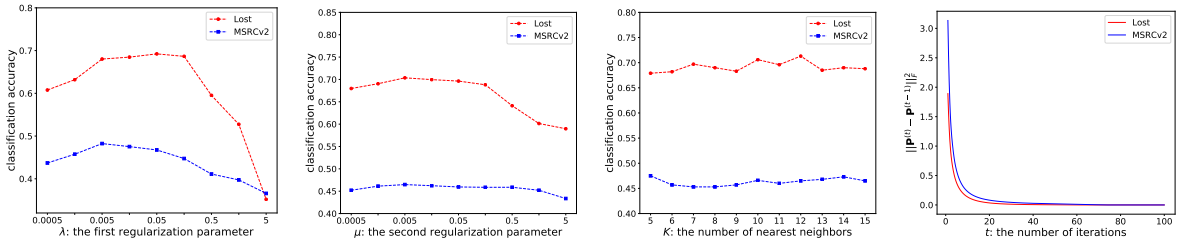
ground-truth label $y \in \mathcal{Y}$, one extra label $y' \neq y$ is selected as the coupled label that co-occurs with y in the candidate label set with probability ϵ, and any other label is chosen to be the false positive label with the probability $1 - \epsilon$. Fig. 4.2 reports the classification accuracy of each algorithm as p ranges from 0.1 to 0.7 with step size 0.1 when r is set to 1. In this setting, r labels are randomly selected as incorrect candidate labels of each partially labeled example. In addition, we also do experiments on controlled datasets as p ranges from 0.1 to 0.7 with r set to 2 and 3, and the experimental results are reported in Fig. 4.3 and Fig. 4.4.

As shown in the above four figures, LALO outperforms the compared algorithms in most cases. Furthermore, Table 4.2 reports the concrete counts of win/tie/loss between LALO and other compared algorithms. As shown in this table, out of the 112 results (with 4 UCI datasets and 28 configurations), we can find that LALO can achieve superior or at least comparable performance against all compared algorithms in most cases, and lose to them in only a few cases.

Real-World Datasets. The test accuracy of each algorithm on real-world datasets is recorded in Table 4.3. It is worth noting that Avg. CLs of FG-NET is somewhat big,

Table 4.3: Classification accuracy of each algorithm on the used real-world partially labeled datasets. Furthermore, •/◦ indicates whether LALO is statistically superior/inferior to the compared algorithm.

	LALO	PL-KNN	CLPL	IPAL	PL-SVM	LSB-CMM
Lost	0.693±0.024	0.332±0.030•	0.670±0.024	0.576±0.035•	0.639±0.056•	0.591±0.019•
MSRCv2	0.465±0.013	0.417±0.012•	0.375±0.020•	0.476±0.019	0.417±0.027•	0.431±0.008•
Soccer Player	0.523±0.005	0.494±0.004•	0.347±0.004•	0.525±0.006	0.430±0.004•	0.506±0.006•
Yahoo! News	0.613±0.004	0.403±0.004•	0.457±0.005•	0.565±0.004•	0.615±0.002	0.594±0.007•
FG-NET	0.073±0.006	0.037±0.008•	0.047±0.017•	0.054±0.006•	0.058±0.010•	0.056±0.008•
FG-NET(MAE3)	0.424±0.011	0.284±0.035•	0.240±0.045•	0.347±0.021•	0.343±0.022•	0.344±0.026•
FG-NET(MAE5)	0.569±0.020	0.438±0.033•	0.343±0.055•	0.512±0.020•	0.473±0.016•	0.478±0.025•



(a) Vary λ (b) Vary μ (c) Vary K (d) Convergence curves

Fig. 4.5: Parameter sensitivity analysis of LALO on the real-world datasets Lost and MSRCv2.

hence the predictive performance of each method is quite bad. For properly evaluating the performance of the facial age estimation task, we further utilize the popular mean absolute error (MAE) [102] to provide other two ways of evaluation. Specifically, for FG-NET (MAE3/MAE5), a test instance is considered correctly classified if the MAE between the predicted age and the true age is no more than 3/5 years. From Table 4.3, we can observe that LALO is statistically better than other compared algorithms on all the real-world datasets except CLPL on Lost, PL-SVM on Yahoo! News, and IPAL on MSRCv2 and Soccer Player, and LALO is never significantly outperformed by any compared algorithm.

4.4.3 Sensitivity Analysis

Here, we present sensitivity analysis of LALO. There are three hyper-parameters λ, μ and K. Fig. 4.5 shows the performance of LALO under different parameter configurations. From Fig. 4.5, we can easily find that the parameter configuration specified for LALO in

Subsection 5.1 ($\lambda = 0.05$, $\mu = 0.005$, $K = 10$) naturally follows the sensitivity curves. In addition, Fig. 4.5 reports the difference of the label distribution matrix $\boldsymbol{P}$ between two successive iterations. We can easily observe that $\left\|\boldsymbol{P}^{(t)} - \boldsymbol{P}^{(t-1)}\right\|_F^2$ gradually decreases to 0 as t increases. Therefore, the convergence of LALO is demonstrated.

4.5 Chapter Summary

In this chapter, we proposed a novel unified partial-label learning framework and presented a biconvex formulation to leverage the latent label distributions for the model training. Extensive experimental results validated the effectiveness of the proposed approach named LALO. Since LALO serves as a bridge between the model training and label propagation, this work could be naturally extended to inductive semi-supervised learning based on label propagation. Besides, it would be also interesting to explore the consistency of the feature space and the label space in other manners.

Chapter 5

Partial-Label Learning with Self-Guided Retraining

5.1 Introduction

In partial-label learning, each training example is represented by a single instance (feature vector) while associated with a set of candidate labels, only one of which is the ground-truth label. Obviously, the available labeling information in the partially labeled training set is ambiguous, as the ground-truth label is concealed in the candidate label set. Hence the critical issue of learning from partially labeled examples is how to effectively differentiate the correct label from the candidate label set. Based on the employed strategy, existing approaches can be roughly grouped into two categories. The first category adopts the average-based strategy while the second category adopts the identification-based strategy. To be specific, the average-based strategy normally considers that all the candidate labels make equal contributions in the model training process, and average their model outputs to produce the final predicted outputs [38, 66, 74, 102]. The identification-based strategy considers the unknown correct label as a hidden variable that could be identified by an iterative refining procedure [41, 71, 77, 103]. Although some of the above approaches are able to extract the relative labeling confidence of each candidate label, they fail to reflect the mutually exclusive relationships among different candidate labels.

Motivated by self-training that takes into account such mutually exclusive relationships by directly labeling an unlabeled instance with enough high confidence, this chapter

aims to give the first attempt to adopt this strategy for learning with partially labeled examples. A straightforward method is to first apply a multi-output model on the partially labeled examples, then pick up the candidate label with enough high confidence as the ground-truth label, finally retrain the model on the resulting data. This process is repeated until no partially labeled examples exist, or no partially labeled examples can be picked up as the ground-truth label. Although this method is intuitive, the model learned from partially labeled examples is probably hard to directly identify the ground-truth label in accordance with the modeling outputs, as candidate label sets exist. Furthermore, the mistakenly identified labels could contagiously mislead the final predictions.

In this chapter[1], we propose a novel partial-label learning approach named SURE (Self-gUided REtraining) to address the above problem. To be specific, we propose a novel unified formulation with the maximum infinity norm regularization to train an effective model in conjunction with pseudo-labeling. Instead of adopting the conventional self-training strategy that simply selects the correct label with high predicting confidence, we impose the maximum infinity norm regularization in the model training process to perform effective pseudo-labeling. In this way, the pseudo labels are decided by balancing the minimum approximation loss and the maximum infinity norm. To optimize the objective function, a convex-concave problem is encountered, as a result of the maximum infinity norm regularization. We demonstrate that the optimization of the convex-concave problem would be consistent with the optimization of multiple quadratic programming problems. In order to speedup the optimization process, we further propose a surrogate objective for practical implementation. Experimental results on a number of controlled and real-world datasets clearly validate the advantage of our proposed approach.

5.2 The Proposed Approach

Let us first formally define the partial-label learning problem with some introduced notations. We denote by $\mathcal{X} \in \mathbb{R}^d$ the feature space with d dimensions and $\mathcal{Y} = \{1, 2, \cdots, k\}$ the corresponding label space with k labels. The given partially labeled dataset is denoted by $\mathcal{D} = \{\boldsymbol{x}_i, S_i\}_{i=1}^n$ where $\boldsymbol{x}_i \in \mathcal{X}$ is an instance and S_i denotes its candidate label

[1]This chapter has been published in [75].

set. The key assumption of partial-label learning lies in that the ground-truth label for $\boldsymbol{x}_i$ is concealed in its candidate label set S_i. The task of partial-label learning is to learn a function $f : \mathcal{X} \to \mathcal{Y}$ from the partially labeled training set $\mathcal{D}$, to correctly predict the label of a test instance. For convenience, we denote by $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n \times d}$ the instance matrix and $\boldsymbol{Y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_n]^\top \in \{0,1\}^{n \times k}$ the annotation matrix of $\boldsymbol{X}$, where y_{ij} denotes whether the i-th instance $\boldsymbol{x}_i$ have the j-th candidate label. By adopting the identification-based strategy, we also regard the ground-truth label as latent variable, and denote by $\boldsymbol{P} = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n]^\top \in [0,1]^{n \times k}$ the confidence matrix where p_{ij} means the probability (confidence) of that the j-th label is the correct label of the i-th instance.

Unlike self-training that takes into account the mutually exclusive relationships among the candidate labels by performing deterministic pseudo-labeling, we introduce the maximum infinity norm regularization to automatically achieve this consideratum. A unified formulation with proper constraints is proposed as follows:

$$\begin{aligned} \min \sum_{i=1}^{n} (\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{p}_i, f) - \lambda \left\| \boldsymbol{p}_i \right\|_\infty) + \beta \Omega(f) \\ \text{s.t.} \quad 0 \le p_{ij} \le y_{ij}, \ \forall i \in [n], \ \forall j \in [k], \\ \sum\nolimits_{j=1}^{l} p_{ij} = 1, \quad \forall i \in [n], \end{aligned}$$

where $[n] := \{1, 2, \cdots, n\}$, $\mathcal{L}$ is the employed loss function, Ω controls the complexity of model f, and λ, β are the tradeoff parameters. In this unified formulation, the model is learned from the pseudo label matrix $\boldsymbol{P}$, rather than the original noisy label matrix $\boldsymbol{Y}$. Besides, unlike the way of traditional self-training to perform deterministic pseudo-labeling by picking up the label with enough high confidence, the confidence of the ground-truth label is differentiated and enlarged by trading off the loss and the maximum infinity norm. Intuitively, only within the allowable range of loss, the candidate label with enough high confidence can be identified as the ground-truth label. In this way, the negative effects of self-training is alleviated by training the model and performing pseudo-labeling jointly. In addition, the first constraint plays two roles: the confidence of each candidate label should be larger than 0, but no more than 1; the confidence of each non-candidate label should be strictly 0. The second constraint guarantees that each confidence vector $\boldsymbol{p}_i$ will always be in the probability simplex, i.e., $\{\boldsymbol{p}_i \in [0,1]^k : \sum_j p_{ij} =$

$1 \mid i \in [n]\}$. This constraint also implicitly takes into consideration the mutually exclusive relationships among the candidate labels, as the confidence of certain one candidate label is enlarged by the maximum infinity norm regularization, the confidences of other candidate labels will be naturally reduced.

To instantiate the above formulation, we adopt the widely used squared loss, i.e., $\mathcal{L}(\boldsymbol{x}_i, \boldsymbol{p}_i, f) = \left\|f(\boldsymbol{x}_i) - \boldsymbol{p}_i\right\|_2^2$. Besides, we employ the simple linear model $f(\boldsymbol{x}_i) = \boldsymbol{W}^\top \boldsymbol{x}_i + \boldsymbol{b}$ where $\boldsymbol{W}, \boldsymbol{b}$ are model parameters. A kernel extension for the general nonlinear case will be introduced in the later section. To control the model parameter, we simply adopt the common squared Frobenius norm of $\boldsymbol{W}$, i.e., $\|\boldsymbol{W}\|_F^2$. To sum up, the final optimization problem is presented as follows:

$$\begin{aligned} \min_{\boldsymbol{P},\boldsymbol{W},\boldsymbol{b}} \sum_{i=1}^{n} (\left\|\boldsymbol{W}^\top \boldsymbol{x}_i + \boldsymbol{b} - \boldsymbol{p}_i\right\|_2^2 - \lambda \left\|\boldsymbol{p}_i\right\|_\infty) + \beta \|\boldsymbol{W}\|_F^2 \\ \text{s.t.} \quad 0 \le p_{ij} \le y_{ij}, \ \forall i \in [n], \ \forall j \in [l], \\ \sum\nolimits_{j=1}^{l} p_{ij} = 1, \quad \forall i \in [n]. \end{aligned} \tag{5.1}$$

5.3 Optimization

Problem (5.1) could be tackled by alternating optimization. This optimization method aims to update one variable while keeping others fixed. The optimization process is repeated until convergence.

5.3.1 Updating Model Parameters

With $\boldsymbol{P}$ fixed, problem (5.1) with respective to $\boldsymbol{W}$ and $\boldsymbol{b}$ can be compactly stated as follows:

$$\min_{\boldsymbol{W},\boldsymbol{b}} \left\|\boldsymbol{X}\boldsymbol{W} + \boldsymbol{1}\boldsymbol{b}^\top - \boldsymbol{P}\right\|_F^2 + \beta \|\boldsymbol{W}\|_F^2, \tag{5.2}$$

where $\boldsymbol{1}$ denotes the vector with all components set to 1. Setting the gradient with respect to $\boldsymbol{W}$ and $\boldsymbol{b}$ to 0, we obtain the following closed-form solutions:

$$\begin{aligned} \boldsymbol{W} &= (\boldsymbol{X}^\top \boldsymbol{X} + \beta \boldsymbol{I} - \frac{\boldsymbol{X}^\top \boldsymbol{1}\boldsymbol{1}^\top \boldsymbol{X}}{n})^{-1} (\boldsymbol{X}^\top \boldsymbol{P} - \frac{\boldsymbol{X}^\top \boldsymbol{1}\boldsymbol{1}^\top \boldsymbol{P}}{n}), \\ \boldsymbol{b} &= \frac{1}{n} (\boldsymbol{P}^\top \boldsymbol{1} - \boldsymbol{W}^\top \boldsymbol{X}^\top \boldsymbol{1}). \end{aligned}$$

Kernel Extension. To deal with the nonlinear case, the above linear learning model can be easily extended to a kernel-based nonlinear model. To achieve this, we utilize a feature mapping $\phi(\cdot) : \mathbb{R}^d \to \mathbb{R}^{\mathcal{H}}$ to map the original feature space $\boldsymbol{x} \in \mathbb{R}^d$ to some higher (maybe infinite) dimensional Hilbert space $\phi(\boldsymbol{x}) \in \mathbb{R}^{\mathcal{H}}$. By representor theorem [109], $\boldsymbol{W}$ could be expressed as a weighted combination of input variables, that is, $\boldsymbol{W} = \phi(\boldsymbol{X})^\top \boldsymbol{A}$ where $\boldsymbol{A} \in \mathbb{R}^{n\times k}$ stores the combination weights of instances. Hence $\phi(\boldsymbol{X})\boldsymbol{W} = \boldsymbol{K}\boldsymbol{A}$ where $\boldsymbol{K} \in \phi(\boldsymbol{X})\phi(\boldsymbol{X})^\top \in \mathbb{R}^{n\times n}$ is the kernel matrix with each element defined by $k_{ij} = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j) = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and $\mathcal{K}(\cdot,\cdot)$ denotes the kernel function. Here, Gaussian kernel function $\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|_2^2/(2\sigma^2))$ is used, where the hyper-parameter σ is set to the average pairwise distance between two examples. By incorporating such kernel extension, problem (5.2) can be presented as follows:

$$\min_{\boldsymbol{A},\boldsymbol{b}} \left\|\boldsymbol{K}\boldsymbol{A} + \boldsymbol{1}\boldsymbol{b}^\top - \boldsymbol{P}\right\|_F^2 + \beta \cdot \mathrm{tr}(\boldsymbol{A}^\top \boldsymbol{K}\boldsymbol{A}),$$

where $\mathrm{tr}(\cdot)$ denotes the trace operator. By further setting the gradient with respect to $\boldsymbol{A}$ and $\boldsymbol{b}$ to 0, we obtain the following closed-form solutions:

$$\boldsymbol{A} = (\boldsymbol{K} + \beta\boldsymbol{I} - \frac{\boldsymbol{1}\boldsymbol{1}^\top \boldsymbol{K}}{n})^{-1}(\boldsymbol{P} - \frac{\boldsymbol{1}\boldsymbol{1}^\top \boldsymbol{P}}{n}), \tag{5.3}$$

$$\boldsymbol{b} = \frac{1}{n}(\boldsymbol{P}^\top \boldsymbol{1} - \boldsymbol{A}^\top \boldsymbol{K}^\top \boldsymbol{1}). \tag{5.4}$$

By adopting this kernel extension, we choose to update the parameters $\boldsymbol{A}$ and $\boldsymbol{b}$ throughout this chapter.

5.3.2 Updating Labeling Confidences

With $\boldsymbol{A}$ and $\boldsymbol{b}$ fixed, the modeling output matrix $\boldsymbol{Q} = [\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n]^\top \in \mathbb{R}^{n\times k}$ is denoted by $\boldsymbol{Q} = \phi(\boldsymbol{X})\boldsymbol{W} + \boldsymbol{1}\boldsymbol{b}^\top = \boldsymbol{K}\boldsymbol{A} + \boldsymbol{1}\boldsymbol{b}^\top$, problem (5.1) reduces to:

$$\begin{aligned} \min_{\boldsymbol{P}} \sum_{i=1}^{n} (\left\|\boldsymbol{p}_i - \boldsymbol{q}_i\right\|_2^2 - \lambda \left\|\boldsymbol{p}_i\right\|_\infty) \\ \text{s.t.} \quad 0 \le p_{ij} \le y_{ij},\ \forall i \in [n],\ \forall j \in [k], \\ \textstyle\sum_{j=1}^{l} p_{ij} = 1, \quad \forall i \in [n]. \end{aligned} \tag{5.5}$$

Obviously, we can solve problem (5.5) by solving n independent problems, one for each example. We further denote by OP the minimum loss of the problem for the i-th example:

$$\begin{aligned} OP = \min_{\boldsymbol{p}_i} \left\|\boldsymbol{p}_i - \boldsymbol{q}_i\right\|_2^2 - \lambda \left\|\boldsymbol{p}_i\right\|_\infty \\ \text{s.t.} \quad \mathbf{1}^\top \boldsymbol{p}_i = 1, \\ \mathbf{0} \leq \boldsymbol{p}_i \leq \boldsymbol{y}_i. \end{aligned} \tag{5.6}$$

Here, problem (5.6) is a constrained convex-concave problem, as the first term is convex while the second term is concave. Instead of using traditional time-consuming convex-concave procedure [110] to solve this problem, we propose a new optimization solution to this problem. Specifically, we demonstrate that the optimization of this problem would be consistent with the optimization of k independent QP problems, each for one label. We denoted by $OPI(j)$ the minimum loss of the problem for the j-th label:

$$\begin{aligned} OPI(j) = \min_{\boldsymbol{p}_i} \left\|\boldsymbol{p}_i - \boldsymbol{q}_i\right\|_2^2 - \lambda p_{ij} \\ \text{s.t.} \quad p_{ik} \leq p_{ij}, \ \forall k \in [k], \\ \mathbf{1}^\top \boldsymbol{p}_i = 1, \\ \mathbf{0} \leq \boldsymbol{p}_i \leq \boldsymbol{y}_i. \end{aligned} \tag{5.7}$$

Theorem 5.1 *The equality* $OP = \min_{j \in [k]} OPI(j)$ *holds.*

Proof. It is obvious that there must exist $j \in [k]$ such that $p_{ij} = \left\|\boldsymbol{p}_i\right\|_\infty$ and the optimum loss OP of problem (5.6) can be obtained. In addition, if $p_{ij} = \left\|\boldsymbol{p}_i\right\|_\infty$ coincidentally holds, then $OPI(j) = OP$, as in such case, problem (5.7) is equivalent to problem (5.6). While if $p_{ij} \neq \left\|\boldsymbol{p}_i\right\|_\infty$, then $OPI(j) > OP$. Hence $OP = \min_{j \in [k]} OPI(j)$. ∎

Theorem 5.1 gives us a motivation to solve problem (5.6) by selecting the minimum loss from k independent quadratic programming problems. However, this may be time-consuming, as the label space could be very large. Thus, we propose a surrogate objective function to upper bound the loss incurred by problem (5.7). Specifically, we select the candidate label j with the maximal modeling output by $j = \arg\max_{j \in S_i} q_{ij}$ where S_i is

the candidate label set containing the indices of candidate labels of the instance $\boldsymbol{x}_i$. The proposed surrogate objective function is given as:

$$\begin{aligned} OPS = \min_{\boldsymbol{p}_i} \left\| \boldsymbol{p}_i - \boldsymbol{q}_i \right\|_2^2 - \lambda p_{ij} \\ \text{s.t.} \quad q_{ik} \leq q_{ij},\ \forall k \in S_i,\ \exists j \in S_i \\ p_{ik} \leq p_{ij},\ \forall k \in [k], \\ \mathbf{1}^\top \boldsymbol{p}_i = 1, \\ \mathbf{0} \leq \boldsymbol{p}_i \leq \boldsymbol{y}_i. \end{aligned} \tag{5.8}$$

Note that the difference between problem (5.8) and problem (5.7) lies in that problem (5.8) adds a constraint to select the label j with the maximal modeling output. Unlike problem (5.7) that considers the possibility of each label being the ground-truth label, problem (5.8) directly assumes the candidate label j with the maximal modeling output $j = \arg\max_{j \in S_i} q_{ij}$ to be the label that is most likely the ground-truth label. This assumption coincides with self-training, which also considers the label with the maximal modeling output as the ground-truth label. Different from self-training that manually performs deterministic pseudo-labeling, our approach aims to automatically enlarge the confidence of the label with the maximal modeling output as much as possible by balancing the two terms in problem (5.8). In this way, we not only avoid the opinionated mistakes by self-training, but also take into account the mutually exclusive relationships among candidate labels.

Theorem 5.2 *$OP \leq OPS$.*

Proof. From the formulation of problem (5.8) and problem (5.7), it is easy to see that $OPS \in \{OPI(j) | j \in S_i\}$. Since $S_i \subset [k]$, $OPS \in \{OPI(j) | j \in [k]\}$. Which means, $\min_{j \in [k]} OPI(j) \leq OPS$. Using Theorem (5.1), $OP = \min_{j \in [k]} OPI(j) \leq OPS$. Theorem (5.2) shows that OPS of problem (5.8) is an upper bound of the loss OP incurred by problem (5.6). Hence we can choose to optimize problem (5.8) for efficiency, as only one QP problem is involved. Such problem could be efficiently tackled by any existing QP tools.

Algorithm 2 The SURE Algorithm

Inputs:

$\mathcal{D}$: the partially labeled training set
λ, β: the hyper-parameters of regularization
$\widetilde{\boldsymbol{x}}$: the unseen test instance

Process:

construct the kernel matrix $\boldsymbol{K} = [\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{n \times n}$;
initialize $\boldsymbol{P} = \boldsymbol{Y}$;
repeat
    update $\boldsymbol{A} = [a_{ij}]_{n \times k}$ and $\boldsymbol{b} = [b_j]_k$ according to Eq. (5.3) and Eq. (5.4);
    update $\boldsymbol{Q} = \boldsymbol{K}\boldsymbol{A} + \boldsymbol{1}\boldsymbol{b}^\top$;
    calculate $\boldsymbol{P}$ by solving (5.8) with a general QP procedure for each training example;
until convergence.
return the predicted label $\widetilde{y}$ according to (5.9).

Output:

$\widetilde{y}$: the predicted label for the test instance $\widetilde{\boldsymbol{x}}$

When the optimization process finishes, we can obtain the predicted label $\widetilde{y}$ of the text instance $\widetilde{\boldsymbol{x}}$ in the following way:

$$\widetilde{y} = \arg\max_{j \in [k]} \sum_{i=1}^{n} a_{ij} \mathcal{K}(\widetilde{\boldsymbol{x}}, \boldsymbol{x}_i) + b_j. \tag{5.9}$$

The pseudo code of SURE is presented in Algorithm 2.

5.4 Experiments

5.4.1 Compared Methods

To demonstrate the effectiveness of SURE, we conduct extensive experiments to compare SURE with the following state-of-the-art partial-label learning methods:

- PLKNN [66]: a K-NN approach that makes predictions by averaging the labeling information of neighboring examples [suggested configuration: $K \in \{5, 6, \cdots, 10\}$];
- CLPL [38]: a convex formulation that deals with partially labeled examples by transforming partial-label learning problem to binary learning problem via feature mapping [suggested configuration: SVM with squared hinge loss];

- IPAL [72]: an instance-based approach that disambiguates candidate labels by an adapted label propagation scheme. [suggested configuration: $\alpha \in \{0, 0.1, \cdots, 1\}$, $K \in \{5, 6, \cdots, 10\}$];
- PLSVM [77]: a maximum margin approach that learns from partially labeled examples by optimizing margin-based objective function [suggested configuration: regularization parameter pool $\lambda \in \{10^{-3}, 10^{-2}, \cdots, 10^{3}\}$];
- PALOC [111]: an approach that adapts one-vs-one decomposition strategy to enable binary decomposition for learning from partially labeled examples [suggested configuration: $\mu = 10$];
- LSBCMM [41]: a maximum likelihood approach that learns from partially labeled examples via mixture models [suggested configuration: k mixture components].

The two parameters λ and β for SURE are chosen from $\{0.001, 0.01, 0.05, 0.1, 0.3, 0.5, 1\}$. Parameters for each algorithm are selected by five-fold cross-validation on the training set. For each dataset, ten-fold cross-validation is performed where mean prediction accuracies and the standard deviations are recorded. In addition, we use t-test at 0.05 significance level for two independent samples to investigate whether SURE is significantly superior/inferior (win/loss) to the compared algorithms for all the experiments.

5.4.2 Controlled Datasets

We collect four controlled datasets including glass, ecoli, deter, and usps. The first three datasets are collected from the UCI repository [96], and the last dataset is collected from the website of the late Sam Roweis[2]. The characteristics of the four datasets are reported in Table 5.1. Following the widely-used controlling protocol [38, 41, 72, 74, 111, 112], each controlled dataset can be used to generate artificial partially labeled datasets. There are three controlling parameters p, r and ϵ where p controls the proportion of partially labeled examples, r decides the number of incorrect labels in the candidate label set, and ϵ denotes the probability of a specific incorrect candidate label appearing with the ground-truth label across all the training examples. As shown in Table 5.1, there are 4

[2] `https://cs.nyu.edu/~roweis/data.html`

Table 5.1: Characteristics of the controlled datasets.

Dataset	deter	ecoli	glass	usps
Examples	358	336	214	9298
Features	23	7	9	256
Labels	6	8	6	10

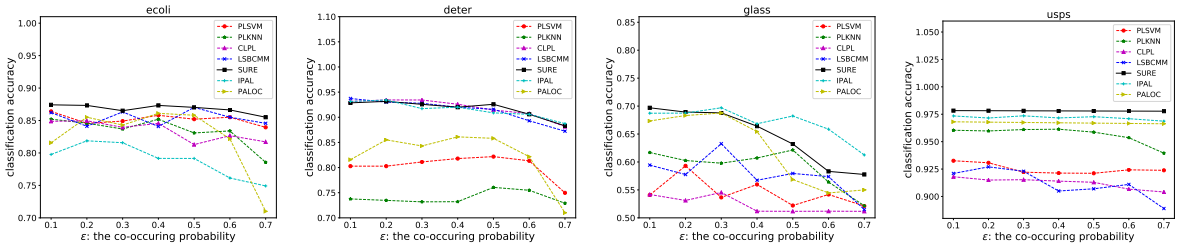
Configurations:

(I) $p = 0.1, r = 1, \epsilon \in \{0.1, 0.2, \cdots, 0.7\}$

(II) $r = 1, p \in \{0.1, 0.2, \cdots, 0.7\}$

(III) $r = 2, p \in \{0.1, 0.2, \cdots, 0.7\}$

(IV) $r = 3, p \in \{0.1, 0.2, \cdots, 0.7\}$



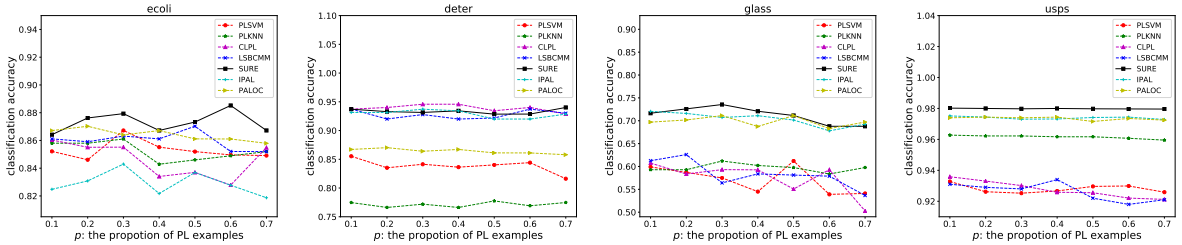
(a) ecoli (b) deter (c) glass (d) usps

Fig. 5.1: Classification accuracy on controlled datasets with ϵ ranging from 0.1 to 0.7 ($p = 1, r = 1$).

configurations, each corresponding to 7 results. Hence we can totally generate $4 \times 4 \times 7 = 112$ different artificial partially labeled datasets.

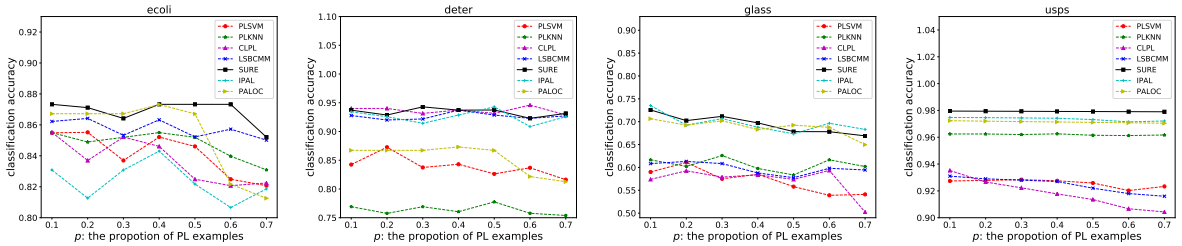
Fig. 5.1 shows the classification accuracy of each algorithm as ϵ ranges from 0.1 to 0.7 when $p = 0.1$ and $r = 1$ (Configuration (I)). In Configuration (I), we need to specifies an incorrect label, which has a probability of ϵ to appear with the correct label. If that label is not selected as an incorrect candidate label, then other incorrect labels would be randomly sampled to be an incorrect candidate label. Figs 5.2, Fig. 5.3, and Fig. 5.4 illustrate the classification accuracy of each algorithm as p ranges from 0.1 to 0.7 when $r = 1, 2$, and 3 (Configuration (II), (III), and (IV)), respectively. In these three settings, r extra labels are randomly chosen to be the false positive labels. That is, the number of candidate labels for each instance is $r + 1$.

As shown in Fig. 5.1, Fig. 5.2, Fig. 5.3, and Fig. 5.4, SURE outperforms other compared algorithms in general. To further statistically compare SURE with other algorithms, the detailed win/tie/loss counts between SURE and the compared algorithms



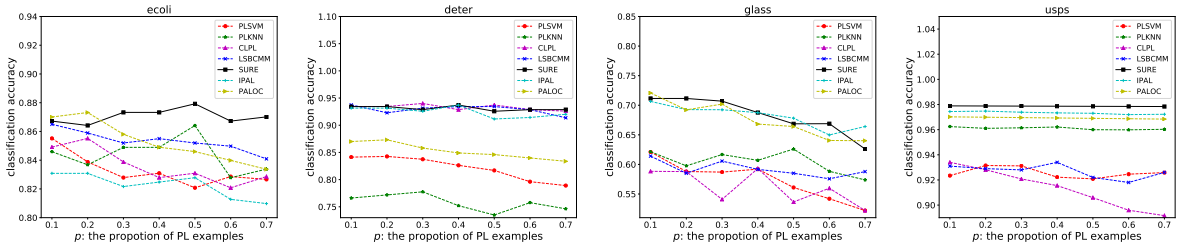
(a) ecoli (b) deter (c) glass (d) usps

Fig. 5.2: Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 1$).



(a) ecoli (b) deter (c) glass (d) usps

Fig. 5.3: Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 2$).



(a) ecoli (b) deter (c) glass (d) usps

Fig. 5.4: Classification accuracy on controlled datasets with p ranging from 0.1 to 0.7 ($r = 3$).

are recorded in Table 5.2. Out of the 112 results, it is easy to observe that:

- SURE achieves superior or at least comparable performance against PLKNN and PLSVM in all cases.
- SURE achieves superior performance against CLPL and LSCMM in 72.3% and 58.9% cases while outperformed by them in only 4.5% and 1.8% cases, respectively.

Table 5.2: Win/tie/loss (t-test at 0.05 significance level for two independent samples) counts on the controlled datasets between SURE and the compared algorithms.

	PLKNN	CLPL	IPAL	PLSVM	PALOC	LSBCMM
(I)	24/4/0	21/6/1	14/11/3	21/7/0	20/8/0	14/14/0
(II)	24/4/0	19/7/2	15/13/0	24/4/0	17/10/1	17/10/1
(III)	24/4/0	21/6/1	14/13/1	25/3/0	17/10/1	19/9/0
(IV)	25/3/0	20/7/1	14/12/2	27/1/0	17/10/1	16/11/1
Total	**97/15/0**	**81/26/5**	**57/49/6**	**97/15/0**	**71/38/3**	**66/44/2**

Table 5.3: Characteristics of real-world partially labeled datasets.

Dataset	Examples	Features	Labels	Avg. CLs	Task Domain
Lost	1122	108	16	2.23	*automatic face naming* [108]
MSRCv2	1758	48	23	3.16	*object classification* [41]
Soccer Player	17472	279	171	2.09	*automatic face naming* [39]
Yahoo! News	22991	163	219	1.91	*automatic face naming* [107]
FG-NET	1002	262	78	7.48	*facial age estimation* [108]

Table 5.4: Classification accuracy of each algorithm on the real-world datasets. Furthermore, •/∘ indicates whether SURE is statistically superior/inferior to the compared algorithm (t-test at 0.05 significance level for two independent samples).

	SURE	PLKNN	CLPL	IPAL	PLSVM	PALOC	LSBCMM
Lost	0.781±0.039	0.432±0.051•	0.742±0.038•	0.678±0.053•	0.729±0.042•	0.629±0.056•	0.693±0.035•
MSRCv2	0.515±0.027	0.417±0.034•	0.413±0.041•	0.529±0.039	0.461±0.046•	0.479±0.042•	0.473±0.037•
Soccer Player	0.533±0.017	0.495±0.018•	0.368±0.010•	0.541±0.016	0.464±0.011•	0.537±0.015	0.498±0.017•
Yahoo! News	0.644±0.015	0.483±0.011•	0.462±0.009•	0.609±0.011•	0.629±0.010•	0.625±0.005•	0.645±0.005
FG-NET	0.078±0.021	0.039±0.018•	0.063±0.027	0.054±0.030•	0.063±0.029	0.065±0.019	0.059±0.025•
FG-NET(MAE3)	0.458±0.024	0.269±0.045•	0.458±0.022	0.362±0.034•	0.356±0.022•	0.435±0.018•	0.382±0.029•
FG-NET(MAE5)	0.615±0.019	0.438±0.053•	0.596±0.017•	0.540±0.033•	0.479±0.016•	0.609±0.043	0.532±0.038•

- SURE outperforms IPAL and PALOC in 50.9% and 63.4% cases while outperformed by them in only 5.4% and 2.7% cases, respectively.

In summary, the effectiveness of SURE on controlled datasets is demonstrated.

5.4.3 Real-World Datasets

Table 5.3 reports the characteristics of real-world partially labeled datasets including Lost [38], MSRCv2 [41], Soccer Player [39], Yahoo! News [107], and FG-NET [108]. These real-world partially labeled datasets are from several task domains. For *automatic face naming* (Lost, Soccer Player, and Yahoo! News), each face (instance) is cropped from an image or a video frame, and the names appearing on the corresponding captions

or subtitles are taken as candidate labels. For *facial age estimation* (FG-NET), human faces are regarded as instances while ages annotated by crowdsourcing labelers serve as candidate labels. For *object classification* (MSRCv2), each image segment is considered as an instance, and objects appearing in the same image are taken as candidate labels. The average number of candidate labels (Avg. CLs) per instance is also reported in Table 5.3.

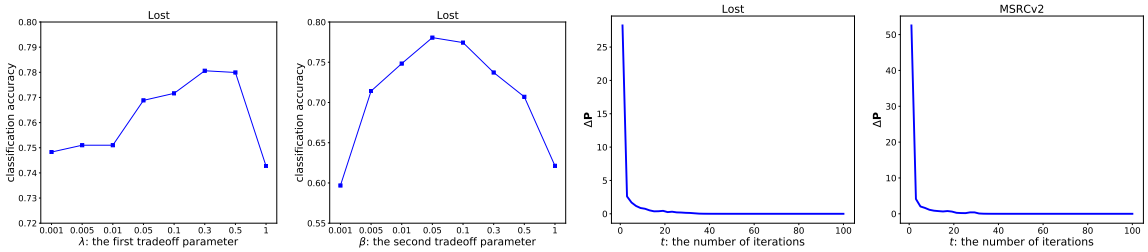
Table 5.4 reports the mean classification accuracy as well as the standard deviation of each algorithm on each real-world dataset. It is worth noting that Avg. CLs of FG-NET is somewhat big, hence the predictive performance of each method is quite bad. For properly evaluating the performance of the facial age estimation task, we further utilize the popular *mean absolute error* (MAE) [102] to provide other two ways of evaluation. Specifically, for FG-NET (MAE3/MAE5), a test example is considered correctly classified if the MAE between the predicted age and the ground-truth age is no more than 3/5 years. As shown in Table 5.4, we can observe that:

- SURE significantly outperforms PLKNN on all the real-world datasets.
- Out of the 42 cases (6 compared algorithms and 7 datasets), SURE significantly outperforms all the compared algorithms in 78.6% cases, and achieves competitive performance in 21.4% cases.
- It is noteworthy that SURE never statistically loses to other compared algorithms.

These experimental results on real-world datasets also demonstrate the effectiveness of SURE.

5.4.4 Further Analysis

Parameter Sensitivity Analysis. There are two tradeoff hyper-parameters λ and β for SURE, which should be manually searched in advance. Hence this section studies how λ and β influence the prediction accuracy produced by SURE. One parameter is varied, while the others are fixed at the best setting. Fig. 5.5a and Fig. 5.5b show the performance of SURE on the Lost dataset given different values of λ and β. It is worth noting that λ decides the importance of the maximum infinity norm regularization. When



(a) Varying λ on Lost (b) Varying β on Lost (c) Lost Curve (d) MSRCv2 Curve

Fig. 5.5: Parameter sensitivity and convergence analysis of SURE.

λ is very small, the mutually exclusive relationships among labels are hardly considered, thus the classification accuracy would be at a low level. As λ increases, we start to consider such exclusive relationships, thus the classification accuracy would increase. Nevertheless, if λ is sufficiently large, we can observe that the accuracy would sharply drop. This is because when we overly concentrate on the mutually exclusive relationships among labels, we will directly regard the candidate label that has the maximal modeling output as the ground-truth label. Since to maximize the infinity norm $\|\boldsymbol{p}\|_\infty$ is overly important, the approximation loss will be totally ignored. From the above, we can draw a conclusion that it would be better to balance the approximation loss and the mutually exclusive relationships among labels. Such conclusion clearly confirms the effectiveness of the SURE approach. Another tradeoff parameter β controls the model complexity. We can observe that the classification accuracy curve of varying β clearly reveals the issues of overfitting and undersfitting.

Illustration of Convergence. We illustrate the convergence of SURE by using the difference of the Frobenius norm of the matrix variable $\boldsymbol{P}$ between two nearby iterations (i.e., $\Delta\boldsymbol{P} = \left\|\boldsymbol{P}^{(t+1)} - \boldsymbol{P}^{(t)}\right\|_F$). Fig. 5.5c and Fig. 5.5d show the convergence curves of SURE on Lost and MSRCv2 respectively. It is apparent that $\Delta\boldsymbol{P}$ gradually decreases to 0 as the number of iterations t increases. Hence the convergence of SURE is demonstrated.

5.5 Chapter Summary

In this chapter, we utilized the idea of self-training to exaggerate the mutually exclusive relationships among candidate labels for further enhancing the practical performance of

partial-label learning. Instead of manually performing pseudo-labeling after model training, we proposed a unified formulation (named SURE) with the maximum infinity norm regularization to train an effective model in conjunction with pseudo-labeling. Extensive experimental results demonstrated the effectiveness of SURE.

Since self-training is a typical semi-supervised learning method, it would be interesting to extend SURE to the setting of semi-supervised learning. Besides, as mutually exclusive relationships exist in general multi-class problems, it would be valuable to explore other possible ways to incorporate such relationships into partial-label learning.

Chapter 6

Partial-Label Learning with Provably Consistency

6.1 Introduction

Partial-label learning aims to deal with the problem where each instance is provided with a set of candidate labels, only one of which is the correct label. In recent years, partial-label learning [38, 63, 67, 71, 72, 75, 113] has attracted increasing attention from machine learning and data mining communities.

In this chapter[1], we for the first time present a novel statistical model to depict the generation process of partially labeled data. Having an explicit data distribution not only helps us to understand how partially labeled examples are generated, but also enables us to perform effective empirical risk minimization. Our proposed data generation model is instance-independent, which does not introduce any extra hidden variable. We verify that the proposed generation model satisfies the key assumption of partial-label learning that the correct label is always included in the set of candidate labels.

Based on the data generation model, , we have the following contributions:

- We derive a novel *risk-consistent* method and a novel *classifier-consistent* method. Most of the existing PLL methods need to specially design complex optimization objectives, which make the optimization process inefficient. In contrast, our proposed PLL methods are model-independent and optimizer-independent, and thus

[1]This chapter has been published in [114].

can be naturally applied to complex models such as deep neural networks with any advanced optimizer.

- We derive an estimation error bound for each of the two methods, which demonstrates that the obtained empirical risk minimizer would approximately converge to the true risk minimizer as the number of training data tends to infinity. We show that the risk-consistent method holds a tighter estimation error bound than the classifier-consistent method and empirically validate that the risk-consistent method achieves better performance when deep neural networks are used.

- To show the effect of our generation model, we also use *entropy* to measure how well the given candidate label sets match our generation model. We find that the candidate label sets with higher entropy better match our generation model, and on such datasets, our proposed PLL methods achieve better performance.

Extensive experiments on benchmark as well as real-world partially labeled datasets clearly validate the effectiveness of our proposed methods.

6.2 Related Formulations

Before presenting our proposed methods, we define some mathematical notations and briefly review relevant formulations of learning with ordinary labels, learning with partial labels, and learning with complementary labels.

6.2.1 Learning with Ordinary Labels

For ordinary multi-class learning, let the feature space be $\mathcal{X} \in \mathbb{R}^d$ and the label space be $\mathcal{Y} = [k]$ (with k classes) where $[k] := \{1, 2, \ldots, k\}$. Let us clearly define that $\boldsymbol{x}$ denotes an instance and $(\boldsymbol{x}, y)$ denotes an example including an instance $\boldsymbol{x}$ and a label y. When ordinary labels are provided, we usually assume each example $(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}$ is independently sampled from an unknown data distribution with probability density $p(\boldsymbol{x}, y)$. Then, the goal of multi-class learning is to obtain a multi-class classifier $f : \mathcal{X} \to \mathbb{R}^k$ that minimizes the following classification risk:

$$R(f) = \mathbb{E}_{p(\boldsymbol{x},y)}[\mathcal{L}(f(\boldsymbol{x}), y)],$$

where $\mathbb{E}_{p(\boldsymbol{x},y)}[\cdot]$ denotes the expectation over the joint probability density $p(\boldsymbol{x}, y)$ and $\mathcal{L} : \mathbb{R}^k \times \mathcal{Y} \to \mathbb{R}_+$ is a multi-class loss function that measures how well a classifier estimates a given label. We say that a method is *classifier-consistent* if the learned classifier by the method is infinite-sample consistent to $\arg\min_{f \in \mathcal{F}} R(f)$, and a method is *risk-consistent* if the method possesses a classification risk estimator that is equivalent to $R(f)$ given the same classifier f. It is worth noting that a risk-consistent method is also classifier-consistent [27]. However, a classifier-consistent method may not be risk-consistent.

6.2.2 Learning with Partial Labels

For learning with partial labels (i.e., partial-label learning), each instance is provided with a set of candidate (partial) labels, only one of which is correct. Suppose the partially labeled dataset is denoted by $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, Y_i)\}_{i=1}^n$ where Y_i is the candidate label set of $\boldsymbol{x}_i$. Since each candidate label set should not be the empty set nor the whole label set, we have $Y_i \in \mathcal{C}$ where $\mathcal{C} = \{2^{\mathcal{Y}} \setminus \emptyset \setminus \mathcal{Y}\}$, $2^{\mathcal{Y}}$ denotes the power set, and $|\mathcal{C}| = 2^k - 2$. The key assumption of partial-label learning lies in that the correct label y_i of $\boldsymbol{x}_i$ must be in the candidate label set, i.e.,

$$p(y_i \in Y_i \mid \boldsymbol{x}_i, Y_i) = 1, \ \forall (\boldsymbol{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}, \ \forall Y_i \in \mathcal{C}. \tag{6.1}$$

Given such data, the goal of partial-label learning is to induce a multi-class classifier $f : \mathcal{X} \to \mathbb{R}^k$ that can make correct predictions on test inputs. To this end, many methods [41,65,70,72,75,113] have been proposed to improve the performance of partial-label learning. However, to the best of our knowledge, there is only one method [38] that possesses statistical consistency by providing a classifier-consistent risk estimator. However, it not only requires the assumption that the data distribution should ensure a limited ambiguity degree, but also relies on some strict conditions (e.g., convexity of loss function and dominance relation [38]). It is still unclear what the explicit formulation of the data distribution for successful partial-label learning would be. Besides, it is also unknown whether there exists a risk-consistent method that possesses a statistical unbiased estimator of the classification risk $R(f)$.

6.2.3 Learning with Multiple Complementary Labels

There is a special case of partial labels, called complementary labels [35, 36, 62]. Each complementary label specifies one of the classes that the example does *not* belong to. A recent study [37] focused on learning with multiple complementary labels. It is worth noting that even though learning with multiple complementary labels could be taken as partial-label learning by regarding complementary labels as non-candidate labels, the formulation of learning with multiple complementary labels is exactly the generalization of previous complementary-label learning studies [35, 36]. Which means, it focuses on generalizing ordinary complementary-label learning to learning with multiple complementary labels instead of learning from candidate labels (i.e., traditional partial-label learning). Suppose each training example is represented by $(\boldsymbol{x}, \overline{Y})$ where $\overline{Y}$ denotes a set of multiple complementary labels, and $(\boldsymbol{x}, \overline{Y})$ is assumed to be independently sampled from the probability distribution with density $\overline{p}(\boldsymbol{x}, \overline{Y})$, which is defined as

$$\overline{p}(\boldsymbol{x}, \overline{Y}) = \sum_{j=1}^{k-1} p(s = j)\overline{p}(\boldsymbol{x}, \overline{Y} \mid s = j), \tag{6.2}$$

where

$$\overline{p}(\boldsymbol{x}, \overline{Y} \mid s = j) = \begin{cases} \frac{1}{\binom{k-1}{j}} \sum_{y \notin \overline{Y}} p(\boldsymbol{x}, y) & \text{if } |\overline{Y}| = j, \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$

Here, the variable s denotes the size of the complementary label set. Supplied with this data distribution, a risk-consistent method [37] was proposed. It is worth noting that following the distribution of complementarily labeled data, although we can obtain partial labels by regarding all the complementary labels as non-candidate labels, the resulting distribution of partially labeled data is not explicitly formulated. It would be natural to ask whether there also exists an explicit formulation of the partially labeled data distribution that enables us to derive a novel classifier-consistent method or a novel risk-consistent method that possesses statistical consistency. In this chapter, we will give an affirmative answer to this question. Specifically, we will show that based on our proposed data generation model, a novel risk-consistent method (the first one for partial-label learning) and a novel classifier-consistent method can be derived accordingly.

6.3 Data Generation Model

In this section, we present an explicit formulation of the generation process of partially labeled data, and show that it satisfies the key assumption (i.e., Eq. (6.1)) of partial-label learning.

6.3.1 Partially Labeled Data Distribution

We assume each partially labeled example $(\boldsymbol{x}, Y)$ is independently drawn from a probability distribution with the following density:

$$\widetilde{p}(\boldsymbol{x}, Y) = \sum_{i=1}^{k} p(Y \mid y = i) p(\boldsymbol{x}, y = i), \tag{6.4}$$

where $p(Y \mid y = i)$ is defined as:

$$p(Y \mid y = i) = \begin{cases} \frac{1}{2^{k-1}-1} & \text{if } i \in Y, \\ 0 & \text{if } i \notin Y. \end{cases} \tag{6.5}$$

In Eq. (6.4), we assume $p(Y \mid \boldsymbol{x}, y) = p(Y \mid y)$, which means, given the correct label y, the candidate label set Y is independent of the instance $\boldsymbol{x}$. This assumption is similar to the conventional modeling of label noise [25] where the observed noisy label is independent of the instance, given the correct label. In addition, there are in total $2^{k-1} - 1$ possible candidate label sets that contain a specific label y. Hence, Eq. (6.5) describes the probability of each candidate label set being uniformly sampled, given a specific label. Here, we show that our assumed data distribution is a valid probability distribution by the following theorem.

Theorem 6.1 *The equality* $\int_{\mathcal{C}} \int_{\mathcal{X}} \widetilde{p}(\boldsymbol{x}, Y) \mathrm{d}\boldsymbol{x} \ \mathrm{d}Y = 1$ *holds.*

Proof. From our formulation of the partially labeled data distribution $\widetilde{p}(\boldsymbol{x}, Y)$, we can

obtain the simplified expression $\widetilde{p}(\boldsymbol{x}, Y) = \frac{1}{2^{k-1}-1} \sum_{y \in Y} p(\boldsymbol{x}, y)$. Then, we have

$$\begin{aligned}
\int_C \int_{\mathcal{X}} \widetilde{p}(\boldsymbol{x}, Y) \mathrm{d}\boldsymbol{x} \ \mathrm{d}Y &= \int_{\mathcal{X}} \sum_{Y \in C} \widetilde{p}(\boldsymbol{x}, Y) \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{2^{k-1}-1} \int_{\mathcal{X}} \sum_{Y \in C} \sum_{y \in Y} p(\boldsymbol{x}, y) \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{2^{k-1}-1} \int_{\mathcal{X}} \sum_{y=1}^{k} \sum_{Y \in \{Y \mid Y \in C, y \in Y\}} p(\boldsymbol{x}, y) \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{2^{k-1}-1} \int_{\mathcal{X}} \sum_{y=1}^{k} (2^{k-1}-1) p(\boldsymbol{x}, y) \mathrm{d}\boldsymbol{x} \\
&= 1,
\end{aligned}$$

which concludes the proof of Theorem 6.1. ∎

Given the assumed data distribution in Eq. (6.4), it would be natural to ask whether our assumed data distribution meets the key assumption of partial-label learning described in Eq. (6.1), i.e., whether the correct label y is always in the candidate label set Y for every partially labeled example $(\boldsymbol{x}, Y)$ sampled from $\widetilde{p}(\boldsymbol{x}, Y)$. The following theorem provides an affirmative answer to this question.

Theorem 6.2 *For any partially labeled example $(\boldsymbol{x}, Y)$ independently sampled from the assumed data distribution in Eq. (6.4), the correct label y is always in the candidate label set Y, i.e., $p(y \in Y \mid \boldsymbol{x}, Y) = 1, \ \forall (\boldsymbol{x}, Y) \sim \widetilde{p}(\boldsymbol{x}, Y)$.*

Proof. It is intuitive to express $p(y \in Y \mid \boldsymbol{x}, Y)$ as

$$\begin{aligned}
p(y \in Y \mid \boldsymbol{x}, Y) &= 1 - p(y \notin Y \mid \boldsymbol{x}, Y) \\
&= 1 - \sum_{i \notin Y} p(y = i \mid \boldsymbol{x}, Y) \\
&= 1 - \sum_{i \notin Y} \frac{p(Y \mid y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x})}{p(Y \mid \boldsymbol{x})} \\
&= 1 - \sum_{i \notin Y} \frac{p(Y \mid y = i) p(y = i \mid \boldsymbol{x})}{\sum_{j=1}^{k} p(Y \mid y = j) p(y = j \mid \boldsymbol{x})} \\
&= 1 - (2^{k-1}-1) \sum_{i \notin Y} \frac{p(Y \mid y = i) p(y = i \mid \boldsymbol{x})}{\sum_{j \in Y} p(y = j \mid \boldsymbol{x})} \\
&= 1,
\end{aligned}$$

where the last equality holds because $p(Y \mid y = i) = 0$ if $i \notin Y$, in terms of Eq. (5). ■

Theorem 6.2 clearly demonstrates that our assumed data distribution in Eq. (6.4) satisfies the key assumption of partial-label learning.

6.3.2 Motivation for Data Generation

Here, we provide a motivation to justify why we mathematically formulated the above data generation model.

Generally, a large number of high-quality samples are notably helpful to machine learning or data mining. However, it is usually difficult for our labelers to directly identify the correct label for each instance [5]. Nonetheless, it would be easier to collect a set of candidate labels that contains the correct label. Suppose there is a labeling system that can uniformly sample a label set Y from $\mathcal{C}$. For each instance $\boldsymbol{x}$, the labeling system uniformly samples a label set Y and asks a labeler whether the correct label y is in the sampled label set Y. In this case, the collected examples whose correct label y is included in the proposed label set Y follow the same distribution as Eq. (6.4). In order to justify that, we first introduce the following lemma.

Lemma 6.1 *Given any instance $\boldsymbol{x}$ with its correct label y, for any unknown label set Y that is uniformly sampled from $\mathcal{C}$, the equality $p(y \in Y \mid \boldsymbol{x}) = 1/2$ holds.*

It is quite intuitive to verify that Lemma 6.1 indeed holds. Specifically, if we do not have any information of Y, we may randomly guess with even probabilities whether the correct y is included in an unknown label set Y or not. Here, we provide a rigorous mathematical proof.

Proof. Let us first consider the case where the correct label y is a specific label i

($i \in [k]$), then we have

$$
\begin{aligned}
p(y \in Y, y = i \mid \boldsymbol{x}) =& p(y \in Y \mid y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) \\
=& \sum_{C \in \mathcal{C}} p(y \in Y, Y = C \mid y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) \\
=& \sum_{C \in \mathcal{C}} p(y \in Y \mid Y = C, y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) p(Y = C \mid \boldsymbol{x}) \\
=& \sum_{C \in \mathcal{C}} p(y \in Y \mid Y = C, y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) p(Y = C) \\
=& \frac{1}{2^k - 2} \sum_{C \in \mathcal{C}} p(y \in Y \mid Y = C, y = i, \boldsymbol{x}) p(y = i \mid \boldsymbol{x}) \\
=& \frac{1}{2^k - 2} |\mathcal{C}^i| \cdot p(y = i \mid \boldsymbol{x}) \\
=& \frac{2^{k-1} - 1}{2^k - 2} p(y = i \mid \boldsymbol{x}) \\
=& \frac{1}{2} p(y = i \mid \boldsymbol{x}),
\end{aligned}
$$

where we have used $p(Y = C \mid \boldsymbol{x}) = p(Y = C) = \frac{1}{2^k - 2}$ because Y is sampled from the whole set of label sets uniformly at random. In addition, $\mathcal{C}^i = \{Y \in \mathcal{C} \mid i \in Y\}$ denotes the set of all the label sets that contain i, hence we can obtain $|\mathcal{C}^i| = 2^{k-1} - 1$. By further summing up the both side over all possible i, we can obtain

$$
\sum_i p(y \in Y, y = i \mid \boldsymbol{x}) = \sum_i \frac{1}{2} p(y = i \mid \boldsymbol{x}) \Rightarrow p(y \in Y \mid \boldsymbol{x}) = \frac{1}{2},
$$

which concludes the proof of Lemma 6.1. ∎

Based on Lemma 6.1, we have the following theorem.

Theorem 6.3 *In the above setting, the distribution of the collected data whose correct label $y \in \mathcal{Y}$ is included in the label set $Y \in \mathcal{C}$ is the same as Eq. (6.4), i.e., $p(\boldsymbol{x}, Y \mid y \in Y) = \widetilde{p}(\boldsymbol{x}, Y)$ where $\widetilde{p}(\boldsymbol{x}, Y)$ is defined in Eq. (6.4).*

Proof. Let us express $p(Y \mid y \in Y, \boldsymbol{x})$ as

$$
\begin{aligned}
p(Y \mid y \in Y, \boldsymbol{x}) =& \frac{p(y \in Y, Y \mid \boldsymbol{x})}{p(y \in Y \mid \boldsymbol{x})} \\
=& \frac{p(y \in Y \mid Y, \boldsymbol{x}) p(Y \mid \boldsymbol{x})}{p(y \in Y \mid \boldsymbol{x})} \\
=& \frac{p(y \in Y \mid Y, \boldsymbol{x}) p(Y)}{p(y \in Y \mid \boldsymbol{x})} \\
=& \frac{2}{2^k - 2} p(y \in Y \mid Y, \boldsymbol{x}) \qquad (\because p(y \in Y \mid \boldsymbol{x}) = \frac{1}{2} \text{ and } p(Y) = \frac{1}{2^k - 2}) \\
=& \frac{1}{2^{k-1} - 1} \sum_{y \in Y} p(y \mid \boldsymbol{x}).
\end{aligned}
$$

By further multiplying $p(\boldsymbol{x})$ on both side, we can obtain $p(\boldsymbol{x}, Y \mid y \in Y) = \widetilde{p}(\boldsymbol{x}, Y)$ where $\widetilde{p}(\boldsymbol{x}, Y)$ is our presented data distribution for partial-label learning. ∎

6.4 Consistent Methods

In this section, based on our assumed partially labeled data distribution in Eq. (6.4), we present a novel risk-consistent method and a novel classifier-consistent method, and theoretically derive an estimator error bound for each of them. Both of the consistent methods are agnostic in specific classification models and can be easily trained with stochastic optimization, which ensures their scalability to large-scale datasets.

6.4.1 Risk-Consistent Method

For the risk-consistent method, we employ the *importance reweighting* strategy [48] to rewrite the classification risk $R(f)$ as

$$\begin{aligned}
R(f) &= \mathbb{E}_{p(\boldsymbol{x},y)}[\mathcal{L}\big(f(\boldsymbol{x}),y\big)] \\
&= \int_{\boldsymbol{x}} \sum_{i=1}^{k} p(y=i \mid \boldsymbol{x}) \mathcal{L}\big(f(\boldsymbol{x}),i\big) p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\
&= \int_{\boldsymbol{x}} \sum_{i=1}^{k} \frac{1}{|C|} \sum_{Y \in C} p(Y \mid \boldsymbol{x}) \frac{p(y=i \mid \boldsymbol{x})}{p(Y \mid \boldsymbol{x})} \mathcal{L}\big(f(\boldsymbol{x}),i\big) p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{|C|} \int_{\boldsymbol{x}} \sum_{Y \in C} p(Y|\boldsymbol{x}) \Big[\sum_{i=1}^{k} \frac{p(y=i \mid \boldsymbol{x})}{p(Y \mid \boldsymbol{x})} \mathcal{L}(f(\boldsymbol{x}),i) \Big] p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{2^k-2} \mathbb{E}_{\widetilde{p}(\boldsymbol{x},Y)} \Big[\sum_{i=1}^{k} \frac{p(y=i \mid \boldsymbol{x})}{p(Y \mid \boldsymbol{x})} \mathcal{L}\big(f(\boldsymbol{x}),i\big) \Big] \\
&= R_{\mathrm{rc}}(f). \qquad (6.6)
\end{aligned}$$

Here, $p(Y \mid \boldsymbol{x})$ can be calculated by

$$p(Y \mid \boldsymbol{x}) = \sum_{j=1}^{k} p(Y \mid y=j) p(y=j \mid \boldsymbol{x}) = \frac{1}{2^{k-1}-1} \sum_{j \in Y} p(y=j \mid \boldsymbol{x}), \qquad (6.7)$$

where the last equality holds due to Eq. (6.4). By substituting Eq. (6.7) into Eq. (6.6), we obtain

$$R_{\mathrm{rc}}(f) = \frac{1}{2} \mathbb{E}_{\widetilde{p}(\boldsymbol{x},Y)} \Big[\sum_{i=1}^{k} \frac{p(y=i \mid \boldsymbol{x})}{\sum_{j \in Y} p(y=j \mid \boldsymbol{x})} \mathcal{L}\big(f(\boldsymbol{x}),i\big) \Big]. \qquad (6.8)$$

In this way, its empirical risk estimator can be expressed as

$$\widehat{R}_{\mathrm{rc}}(f) = \frac{1}{2n} \sum_{o=1}^{n} \Big(\sum_{i=1}^{k} \frac{p(y_o=i \mid \boldsymbol{x}_o)}{\sum_{j \in Y_o} p(y_o=j \mid \boldsymbol{x}_o)} \mathcal{L}\big(f(\boldsymbol{x}_o),i\big) \Big), \qquad (6.9)$$

where $\{\boldsymbol{x}_o, Y_o\}_{o=1}^{n}$ are partially labeled examples drawn from $\widetilde{p}(\boldsymbol{x}, Y)$. Note that $p(y = i \mid \boldsymbol{x})$ is not accessible from the given data. Therefore, we apply the softmax function on the model output $f(\boldsymbol{x})$ to approximate $p(y = i \mid \boldsymbol{x})$, i.e., $p(y = i \mid \boldsymbol{x}) = g_i(\boldsymbol{x})$ where $g_i(\boldsymbol{x})$ is the probability of label i being the true label of $\boldsymbol{x}$, which is calculated by

$g_i(\boldsymbol{x}) = \exp(f_i(\boldsymbol{x}))/\sum_{j=1}^{k}\exp(f_j(\boldsymbol{x}))$, and $f_i(\boldsymbol{x})$ is the i-th coordinate of $f(\boldsymbol{x})$. It is worth noting that the non-candidate labels can never be the correct label. Therefore, we further correct $p(y=i \mid \boldsymbol{x})$ by setting the confidence of each non-candidate label to 0, i.e.,

$$p(y=i \mid \boldsymbol{x}) = \begin{cases} g_i(\boldsymbol{x}), & \text{if } i \in Y, \\ 0, & \text{if } i \notin Y. \end{cases} \tag{6.10}$$

As shown in Eq. (6.9), our risk-consistent method does not rely on specific loss functions, hence we simply adopt the widely-used categorical cross entropy loss for practical implementation. The pseudo-code of the Risk-Consistent (RC) method is presented in Algorithm 3. It is worth noting that the algorithmic process of RC surprisingly coincides with that of PRODEN [115]. However, they are derived in totally different manners. Besides, PRODEN does not hold any theoretical guarantee while we show that our proposed RC method is consistent.

Here, we theoretically analyze the estimation error bound of the risk-consistent method. Specifically, we denote by $\widehat{f}_{\mathrm{rc}} = \min_{f\in\mathcal{F}} \widehat{R}_{\mathrm{rc}}(f)$ the empirical risk minimizer and $f^{\star} = \min_{f\in\mathcal{F}} R(f)$ the true risk minimizer. Besides, we define the function space $\mathcal{H}_y$ for the label $y \in \mathcal{Y}$ as $\{h : \boldsymbol{x} \mapsto f_y(\boldsymbol{x}) \mid f \in \mathcal{F}\}$. Let $\mathfrak{R}_n(\mathcal{H}_y)$ be the expected Rademacher complexity [85] of $\mathcal{H}_y$ with sample size n, then we have the following theorem.

Theorem 6.4 *Assume the loss function $\mathcal{L}(f(\boldsymbol{x}), y)$ is ρ-Lipschitz with respect to $f(\boldsymbol{x})$ ($0 < \rho < \infty$) for all $y \in \mathcal{Y}$ and upper-bounded by M, i.e., $M = \sup_{\boldsymbol{x}\in\mathcal{X}, f\in\mathcal{F}, y\in\mathcal{Y}} \mathcal{L}(f(\boldsymbol{x}), y)$. Then, for any $\delta > 0$, with probability at least $1-\delta$,*

$$R(\widehat{f}_{\mathrm{rc}}) - R(f^{\star}) \leq 4\sqrt{2}\rho \sum_{y=1}^{k} \mathfrak{R}_n(\mathcal{H}_y) + M\sqrt{\frac{\log\frac{2}{\delta}}{2n}},$$

The proof of Theorem 6.4 is provided in Appendix B. Generally, $\mathfrak{R}_n(\mathcal{H}_y)$ can be bounded by $C_{\mathcal{H}}/\sqrt{n}$ for a positive constant $C_{\mathcal{H}}$ [27, 34, 86]. Hence Theorem 6.4 demonstrates that the empirical risk minimizer f_{rc} would converge to the true risk minimizer $f^{\star}$ as $n \to \infty$.

6.4.2 Classifier-Consistent Method

For the classifier-consistent method, we start by introducing a transition matrix $\boldsymbol{Q}$ that describes the probability of the candidate label set given an ordinary label. Specifically,

Algorithm 3 The RC Algorithm

Inputs:

f: the randomly initialized model
$T_{\max}$: the number of training epochs
$I_{\max}$: the number of iterations in each epoch
$\widetilde{\mathcal{D}}$: the partially labeled training set $\{(\boldsymbol{x}_i, Y_i)\}_{i=1}^{n}$

Process:

Initialize $p(y_i = j \mid \boldsymbol{x}_i) = 1, \forall j \in Y_i$, otherwise $p(y_i = j \mid \boldsymbol{x}_i) = 0$;
for $t = 1, 2, \ldots, T_{\max}$ **do**
    Shuffle $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, Y_i)\}_{i=1}^{n}$;
    for $j = 1, \ldots, I_{\max}$ **do**
        Fetch mini-batch $\widetilde{\mathcal{D}}_j$ from $\widetilde{\mathcal{D}}$;
        Update model f by $\widehat{R}_{\mathrm{rc}}$ in Eq. (6.9);
        Calculate the softmax output $g_i(\boldsymbol{x}) = \frac{\exp(f_i(\boldsymbol{x}))}{\sum_{j=1}^{k} \exp(f_j(\boldsymbol{x}))}$;
        Update $p(y_i \mid \boldsymbol{x}_i)$ by Eq. (6.10);
    end for
end for

Output:

f: the trained model

the transition matrix $\boldsymbol{Q}$ is defined as

$$\begin{bmatrix} p(Y = C_1|y = 1) & \cdots & p(Y = C_{2^k-2}|y = 1) \\ p(Y = C_1|y = 2) & \cdots & p(Y = C_{2^k-2}|y = 2) \\ \vdots & \ddots & \vdots \\ p(Y = C_1|y = k) & \cdots & p(Y = C_{2^k-2}|y = k) \end{bmatrix}, \tag{6.11}$$

which means $Q_{ij} = p(C_j \mid y = i)$ where $C_j \in C$ ($j \in [2^k - 2]$) is a specific label set. By further taking into account the assumed data distribution in Eq. (6.4), we can instantiate the transition matrix $\boldsymbol{Q}$ as $Q_{ij} = \frac{1}{2^{k-1}-1}$ if $i \in C_j$, otherwise $Q_{ij} = 0$. Let us introduce $q_j(\boldsymbol{x}) = p(Y = C_j \mid \boldsymbol{x})$ and $g_i(\boldsymbol{x}) = p(y = i \mid \boldsymbol{x})$, then we can obtain $q(\boldsymbol{x}) = \boldsymbol{Q}^\top g(\boldsymbol{x})$.

Given each partially labeled example $(\boldsymbol{x}, Y)$ sampled from $\widetilde{p}(\boldsymbol{x}, Y)$, the proposed classifier-consistent risk estimator is presented as

$$R_{\mathrm{cc}}(f) = \mathbb{E}_{\widetilde{p}(\boldsymbol{x},Y)}[\mathcal{L}(q(\boldsymbol{x}), \widetilde{y})], \text{ where } Y = C_{\widetilde{y}}. \tag{6.12}$$

In this formulation, we regard the candidate label set Y as a virtual label $\widetilde{y}$ if Y is a specific label set $C_{\widetilde{y}}$. Since there are $2^k - 2$ possible label sets, we denote by $\widetilde{\mathcal{Y}}$ the

virtual label space where $\widetilde{\mathcal{Y}} = [2^k - 2]$ and $\widetilde{y} \in \widetilde{\mathcal{Y}}$. It is worth noting that the transition matrix $\boldsymbol{Q}$ has full rank, because all rows of $\boldsymbol{Q}$ are linearly independent by the definition of $\boldsymbol{Q}$. Then, in order to prove that this method is classifier-consistent, we introduce the following lemma.

Lemma 6.2 *If certain loss functions are used (e.g., the softmax cross entropy loss or mean squared error), the optimal mapping $g^\star$ satisfies $g_i^\star(\boldsymbol{x}) = p(y = i \mid \boldsymbol{x})$.*

Proof. **Cross Entropy Loss.** If the cross entropy loss is used, we have the following optimization problem:

$$\phi(g) = -\sum_{i=1}^{k} p(y = i \mid \boldsymbol{x}) \log(g_i(\boldsymbol{x}))$$
$$\text{s.t.} \quad \sum_{i=1}^{k} g_i(\boldsymbol{x}) = 1.$$

By using the Lagrange multiplier method, we can obtain the following non-constrained optimization problem:

$$\Phi(g) = -\sum_{i=1}^{k} p(y = i \mid \boldsymbol{x}) \log(g_i(\boldsymbol{x})) + \lambda(\sum_{i=1}^{k} g_i(\boldsymbol{x}) - 1)).$$

By setting the derivative to 0, we obtain

$$g_i^\star(\boldsymbol{x}) = \frac{1}{\lambda} p(y = i \mid \boldsymbol{x}).$$

Because $\sum_{i=1}^{k} g_i^\star(\boldsymbol{x}) = 1$ and $\sum_{i=1}^{k} p(y = i \mid \boldsymbol{x}) = 1$, we have

$$\sum_{i=1}^{k} g_i^\star(\boldsymbol{x}) = \frac{1}{\lambda} \sum_{i=1}^{k} p(y = i \mid \boldsymbol{x}) = 1.$$

Therefore, we can easily obtain $\lambda = 1$. In this way, $g_i^\star = \frac{1}{\lambda} p(y = i \mid \boldsymbol{x}) = p(y = i \mid \boldsymbol{x})$, which concludes the proof of cross entropy loss. It is worth noting that this proof can be found in [62, 115].

Mean Squared Error. If the mean squared error is used, we have the following optimization problem:

$$\phi(g) = \sum_{i=1}^{k} (p(y = i \mid \boldsymbol{x}) - g_i(\boldsymbol{x}))^2$$
$$\text{s.t.} \quad \sum_{i=1}^{k} g_i(\boldsymbol{x}) = 1.$$

Algorithm 4 The CC Algorithm

Inputs:

f: the randomly initialized model
$T_{\max}$: the number of training epochs
$I_{\max}$: the number of iterations in each epoch
$\widetilde{\mathcal{D}}$: the partially labeled training set $\{(\boldsymbol{x}_i, Y_i)\}_{i=1}^{n}$

1: **for** $t = 1, 2, \ldots, T_{\max}$ **do**
2: Shuffle the partially labeled training set $\widetilde{\mathcal{D}} = \{(\boldsymbol{x}_i, Y_i)\}_{i=1}^{n}$;
3: **for** $j = 1, \ldots, I_{\max}$ **do**
4: Fetch mini-batch $\widetilde{\mathcal{D}}_j$ from $\widetilde{\mathcal{D}}$;
5: Update model f by minimizing the empirical risk estimator $\widehat{R}_{\mathrm{cc}}$ in Eq. (6.13);
6: **end for**
7: **end for**

Output:

f: the trained model

By using the Lagrange multiplier method, we can obtain the following non-constrained optimization problem:

$$\Phi(g) = \sum_{i=1}^{k} (p(y = i \mid \boldsymbol{x}) - g_i(\boldsymbol{x}))^2 + \lambda'(\sum_{i=1}^{k} g_i(\boldsymbol{x}) - 1)).$$

By setting the derivative to 0, we obtain

$$g_i^{\star}(\boldsymbol{x}) = p(y = i \mid \boldsymbol{x}) - \frac{\lambda'}{2}.$$

Because $\sum_{i=1}^{k} g_i^{\star}(\boldsymbol{x}) = 1$ and $\sum_{i=1}^{k} p(y = i \mid \boldsymbol{x}) = 1$, we have

$$\sum_{i=1}^{k} g_i^{\star}(\boldsymbol{x}) = \sum_{i=1}^{k} p(y = i \mid \boldsymbol{x}) - \frac{\lambda' k}{2}$$
$$0 = -\frac{\lambda' k}{2}.$$

Since $k \neq 0$, we can obtain $\lambda' = 0$. In this way, $g_i^{\star} = p(y = i \mid \boldsymbol{x}) - \frac{\lambda'}{2} = p(y = i \mid \boldsymbol{x})$, which concludes the proof of mean squared error.

Theorem 6.5 *When the transition matrix $\boldsymbol{Q}$ has full rank and the condition in 6.2 is satisfied, the minimizer $f_{\mathrm{cc}} = \arg\min_{f \in \mathcal{F}} R_{\mathrm{cc}}(f)$ is also the true minimizer $f^{\star} = \arg\min_{f \in \mathcal{F}} R(f)$, i.e., $f_{\mathrm{cc}} = f^{\star}$ (classifier-consistency).*

Proof. According to Lemma 2, by minimizing $R_{\mathrm{cc}}(f)$ with the cross entropy loss, we can obtain

$$q_j^\star(\boldsymbol{x}) = p(Y = C_j \mid \boldsymbol{x}), \forall j \in [2^k - 2].$$

Let us introduce $\widetilde{\boldsymbol{v}} = [p(Y = C_1 \mid \boldsymbol{x}), p(Y = C_2 \mid \boldsymbol{x}), \ldots, p(Y = C_{2^k-2} \mid \boldsymbol{x})]$ and $\boldsymbol{v} = [p(y = 1 \mid \boldsymbol{x}), p(y = 2 \mid \boldsymbol{x}), \ldots, p(y = k \mid \boldsymbol{x})]$. We have

$$\widetilde{\boldsymbol{v}} = \boldsymbol{Q}^\top \boldsymbol{v}.$$

Since $q^\star(\boldsymbol{x}) = \widetilde{\boldsymbol{v}}$ and $g^\star(\boldsymbol{x}) = \boldsymbol{v}$, we have $q^\star(\boldsymbol{x}) = \boldsymbol{Q}^\top g^\star(\boldsymbol{x})$ where $g^\star(\boldsymbol{x}) = \mathrm{softmax}(f^\star(\boldsymbol{x}))$. On the other hand, by minimizing $R_{\mathrm{cc}(f)}$, we have:

$$q^\star(\boldsymbol{x}) = \boldsymbol{Q}^\top g_{\mathrm{cc}}(\boldsymbol{x}),$$

where $g_{\mathrm{cc}}(\boldsymbol{x}) = \mathrm{softmax}(f_{\mathrm{cc}}(\boldsymbol{x})))$. In this way, we can obtain $\boldsymbol{Q}^\top g^\star(\boldsymbol{x}) = \boldsymbol{Q}^\top g_{\mathrm{cc}}(\boldsymbol{x})$. Therefore, when $\boldsymbol{Q}$ has full rank, we obtain $g_{\mathrm{cc}} = g^\star$, which implies $f_{\mathrm{cc}} = f^\star$. ∎

As suggested by Lemma 6.2, we adopt the cross entropy loss in our classifier-consistent risk estimator (i.e., Eq. (6.12)) for practical implementation. In this way, we have the following empirical risk estimator:

$$\begin{aligned}
\widehat{R}_{\mathrm{cc}}(f) &= -\frac{1}{n}\sum_{i=1}^{n}\Big(\sum_{j=1}^{2^k-2}\mathbb{I}(Y_i = C_j)\log(q_j(\boldsymbol{x}_i))\Big) \\
&= -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{2^k-2}\mathbb{I}(Y_i = C_j)\log\big(\boldsymbol{Q}[:, j]^\top g(\boldsymbol{x})\big) \\
&= -\frac{1}{n}\sum_{i=1}^{n}\log\Big(\frac{1}{2^{k-1}-1}\sum_{y\in Y_i} g_y(\boldsymbol{x})\Big) \\
&= -\frac{1}{n}\sum_{i=1}^{n}\log\Big(\frac{1}{2^{k-1}-1}\sum_{y\in Y_i}\frac{\exp(f_y(\boldsymbol{x}))}{\sum_j \exp(f_j(\boldsymbol{x}))}\Big),
\end{aligned} \tag{6.13}$$

where $\mathbb{I}[\cdot]$ is the indicator function. For the expected risk estimator $R_{\mathrm{cc}}(f)$, it seems that the transition matrix $\boldsymbol{Q} \in \mathbb{R}^{k\times(2^k-2)}$ is indispensable. Unfortunately, it would be computationally prohibitive, since $2^k - 2$ is an extremely large number if the number of classes k is large. However, for practical implementation, Eq. (6.13) shows that we do not need to explicitly calculate and store the transition matrix $\boldsymbol{Q}$, which brings no pain

to optimization. The pseudo-code of the Classifier-Consistent (CC) method is presented in Algorithm 4.

Here, we also establish an estimation error bound for the classifier-consistent method. Let $\widehat{f}_{\mathrm{cc}} = \arg\min_{f\in\mathcal{F}} \widehat{R}_{\mathrm{cc}}(f)$ be the empirical minimizer and $f^\star = \arg\min_{f\in\mathcal{F}} R(f)$ be the true minimizer. Besides, we define the function space $\mathcal{H}_y$ for the label $y \in \mathcal{Y}$ as $\{h : \boldsymbol{x} \mapsto f_y(\boldsymbol{x}) \mid f \in \mathcal{F}\}$. Then, we have the following theorem.

Theorem 6.6 *Assume the loss function $\mathcal{L}(q(\boldsymbol{x}), \widetilde{y})$ is ρ'-Lipschitz with respect to $f(\boldsymbol{x})$ $(0 < \rho < \infty)$ for all $\widetilde{y} \in \widetilde{\mathcal{Y}}$ and upper-bounded by M, i.e., $M = \sup_{\boldsymbol{x}\in\mathcal{X}, f\in\mathcal{F}, \widetilde{y}\in\widetilde{\mathcal{Y}}} \mathcal{L}(q(\boldsymbol{x}), \widetilde{y})$. Then, for any $\delta > 0$, with probability at least $1-\delta$,*

$$R_{\mathrm{cc}}(\widehat{f}_{cc}) - R_{\mathrm{cc}}(f^\star) \leq 4\sqrt{2}\rho' \sum_{y=1}^{k} \mathfrak{R}_n(\mathcal{H}_y) + 2M\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

The proof is provided in Appendix D. Generally, $\mathfrak{R}_n(\mathcal{H}_y)$ can be bounded by $C_{\mathcal{H}}/\sqrt{n}$ for a positive constant $C_{\mathcal{H}}$ [27, 34, 86]. Hence Theorem 6.6 demonstrates that the empirical risk minimizer $\widehat{f}_{\mathrm{cc}}$ would converge to the true risk minimizer $f^\star$ as $n \to \infty$.

Theoretical Comparison Between RC and CC. There exists a clear difference between the estimation error bounds in Theorem 6.4 and Theorem 6.6, especially in the last term. If we assume that ρ for RC and ρ' for CC hold the same value, we can find that the estimation error bound in Theorem 6.6 would be looser than that in Theorem 6.4. Therefore, we could expect that RC may have better performance than CC. In addition, RC needs to estimate the prediction confidence of each example. Intuitively, complex models like deep neural networks normally provide more accurate estimation than linear models. Therefore, we speculate that when more complex models are used, the superiority of RC would be more remarkable. We will demonstrate via experiments that RC is generally superior to CC when deep neural networks are used.

6.5 Experiments

We implement extensive experiments on various datasets to validate the effectiveness of the proposed methods.

Table 6.1: Characteristics of the controlled datasets.

Dataset	#Train	#Test	#Features	#Classes	Model
Yeast	1,335	149	8	10	Linear Model
Texture	4,950	550	40	11	Linear Model
Dermatology	329	37	34	6	Linear Model
Har	9,269	1,030	561	6	Linear Model
20Newsgroups	16,961	1,885	300	20	Linear Model
MNIST	60,000	10,000	784	10	three-layer (d-500-10) MLP, LeNet
Fashion-MNIST	60,000	10,000	784	10	three-layer (d-500-10) MLP, LeNet
Kuzushiji-MNIST	60,000	10,000	784	10	three-layer (d-500-10) MLP, LeNet
CIFAR-10	50,000	10,000	3,072	10	34-layer ResNet, 22-layer DenseNet

Table 6.2: Characteristics of the real-world partially labeled datasets.

Dataset	#Examples	#Features	#Classes	Avg. #CLs	Application Domain	Model
Lost	1,122	108	16	2.23	*automatic face naming* [108]	Linear Model
MSRCv2	1,758	48	23	3.16	*object classification* [41]	Linear Model
BirdSong	4,998	38	13	2.18	*bird song classification* [116]	Linear Model
Soccer Player	17,472	279	171	2.09	*automatic face naming* [39]	Linear Model
Yahoo! News	22,991	163	219	1.91	*automatic face naming* [107]	Linear Model

6.5.1 Experimental Setup

Datasets. We collect four widely used benchmark datasets including MNIST [91], Kuzushiji-MNIST [92], Fashion-MNIST [93], and CIFAR-10 [95], and five datasets from the UCI Machine Learning Repository [95] including Yeast, Texture, Dermatology, Har, and 20Newsgroups. For 20Newsgroups, we obtained the tf-idf features, and applied TruncatedSVD [97] to reduce the dimension to 300. In order to generate candidate label sets on these datasets, following the real-world example in Section 6.3.2, we uniformly sample the candidate label set that includes the correct label from $\mathcal{C}$ for each instance. In addition, we also use five widely used real-world partially labeled datasets, including Lost [38], BirdSong [116], MSRCv2 [41], Soccer Player [39], Yahoo! News [107]. Since our proposed methods do not rely on specific classification models, we use various base models to validate the effectiveness of our methods, including linear model, three-layer (d-500-k) MLP, 5-layer LeNet, 34-layer ResNet [98], and 22-layer DenseNet [99]. The detailed descriptions of these datasets with the corresponding base models are provided in Table 6.1 and Table 6.2.

Table 6.3: Test accuracy (mean±std) of each method using neural networks on benchmark datasets. ResNet is trained on CIFAR-10, and MLP is trained on the other three datasets.

	MNIST	Kuzushiji-MNIST	Fashion-MNIST	CIFAR-10
RC	**98.00±0.11%**	**89.38±0.28%**	**88.38±0.16%**	**77.93±0.59%**
CC	97.87±0.10%•	88.83±0.40%•	87.88±0.25%•	75.78±0.27%•
GA	96.37±0.13%•	84.23±0.19%•	85.57±0.16%•	72.22±0.19%•
NN	96.75±0.08%•	82.36±0.41%•	86.25±0.14%•	68.09±0.31%•
Free	88.48±0.37%•	70.31±0.68%•	81.34±0.47%•	17.74±1.20%•
PC	92.47±0.13%•	73.45±0.20%•	83.37±0.31%•	46.53±2.01%•
Forward	97.64±0.11%•	87.64±0.13%•	86.73±0.15%•	71.18±0.92%•
EXP	97.81±0.04%•	88.48±0.29%•	87.96±0.06%•	73.22±0.66%•
LOG	97.86±0.11%•	88.24±0.08%•	88.31±0.26%	75.38±0.34%•
MAE	97.82±0.11%•	88.43±0.32%•	87.83±0.22%•	66.91±3.08%•
MSE	96.95±0.14%•	85.16±0.44%•	85.72±0.26%•	66.15±2.13%•
GCE	96.71±0.08%•	85.19±0.39%•	86.88±0.16%•	68.39±0.71%•
Phuber-CE	95.10±0.34%•	80.66±0.41%•	85.33±0.23%•	58.60±0.95%•

Table 6.4: Test accuracy (mean±std) of each method using neural networks on benchmark datasets. DenseNet is trained on CIFAR-10, and LeNet is trained on the other three datasets.

	MNIST	Kuzushiji-MNIST	Fashion-MNIST	CIFAR-10
RC	**99.04±0.03%**	**94.00±0.30%**	**89.48±0.15%**	**78.53±0.46%**
CC	98.99±0.08%	93.86±0.18%	88.98±0.20%•	75.71±0.18%•
GA	98.68±0.05%•	90.39±0.26%•	87.95±0.12%•	71.85±0.19%•
NN	98.51±0.08%•	89.60±0.34%•	88.47±0.15%•	71.98±0.35%•
Free	80.48±2.06%•	71.18±1.38%•	74.02±3.88%•	45.94±0.83%•
PC	95.03±0.16%•	79.62±0.11%•	83.98±0.20%•	54.18±2.10%•
Forward	98.80±0.04%•	93.87±0.14%	88.72±0.17%•	73.56±1.47%•
EXP	98.82±0.03%•	92.69±0.31%•	88.99±0.25%•	75.02±1.02%•
LOG	98.88±0.08%•	93.97±0.25%	88.75±0.28%•	75.54±0.59%•
MAE	98.88±0.05%•	93.04±0.52%•	87.30±3.16%•	67.74±0.89%•
MSE	98.38±0.05%•	88.37±0.55%•	88.18±0.08%•	70.66±0.59%•
GCE	98.63±0.06%•	91.27±0.30%•	88.66±0.16%•	72.09±0.51%•
Phuber-CE	96.92±0.18%•	82.24±2.45%•	87.02±0.09%•	66.47±0.35%•

Compared Methods. We compare with 6 state-of-the-art partial-label learning methods including SURE [75], CLPL [38], IPAL [72], PLSVM [13], PLECOC [70], PLKNN [66]. The detailed information of these compared partial-label learning methods is listed as follows:

- SURE [75]: It iteratively enlarges the confidence of the candidate label with the highest probability to be the correct label.
- CLPL [38]: It uses a convex formulation by using the one-versus-all strategy in the

Table 6.5: Transductive accuracy of each method using neural networks on benchmark datasets. ResNet is trained on CIFAR-10, and MLP is trained on the other three datasets.

	MNIST	Kuzushiji-MNIST	Fashion-MNIST	CIFAR-10
RC	**98.81±0.02%**	**97.45±0.06%**	**94.30±0.09%**	**87.48±0.44%**
CC	98.77±0.06%	97.31±0.05%•	93.55±0.14%•	86.15±0.26%•
GA	96.72±0.11%•	94.85±0.08%•	87.34±0.10%•	76.70±0.21%•
NN	97.25±0.08%•	93.91±0.06%•	88.83±0.18%•	74.31±0.35%•
Free	88.38±0.51%•	83.73±0.31%•	82.77±0.61%•	17.74±1.11%•
PC	93.42±0.12%•	88.26±0.10%•	85.54±0.18%•	46.93±2.35%•
Forward	98.68±0.04%•	96.89±0.07%•	91.48±0.26%•	78.72±1.32%•
EXP	98.70±0.03%	97.03±0.12%•	92.60±0.05%•	79.52±0.56%•
LOG	98.75±0.06%	97.18±0.06%•	93.52±0.06%•	85.96±0.45%
MAE	98.63±0.05%•	97.01±0.04%•	92.02±0.08%•	74.31±3.24%•
MSE	97.35±0.24%•	95.61±0.06%•	90.53±0.12%•	69.81±2.43%•
GCE	97.15±0.03%•	95.41±0.04%•	90.80±0.16%•	77.77±0.60%•
Phuber-CE	95.59±0.30%•	91.66±0.23%•	88.65±0.12%•	65.42±0.96%•

multi-class loss function.

- IPAL [72]: It is a non-parametric method that applies the label propagation strategy [83] to iteratively update the confidence of each candidate label.

- PLSVM [13]: It is a maximum margin-based method that differentiates candidate labels from non-candidate labels by maximizing the margin between them.

- PLECOC [70]: It adapts the Error-Correcting Output Codes method to deal with partially labeled examples in a disambiguation-free manner.

- PLKNN [66]: It adapts the widely-used k-nearest neighbors method to make predictions for partially labeled examples.

For all the above methods, their parameters are specified or searched according to the suggested parameter settings by respective papers. It is worth noting that since all the compared partial-label learning methods use full batch size, we also use full batch size (with 2000 training epochs) for our proposed methods RC and CC, to keep fair comparisons.

Besides, we also compare with various *complementary-label learning* (CLL) methods for two reasons: 1) By regarding each non-candidate label as a complementary label, we can transform the partially labeled dataset into complementarily labeled dataset, thus

Table 6.6: Transductive accuracy of each method using neural networks on benchmark datasets. DenseNet is trained on CIFAR-10, and LeNet is trained on the other three datasets.

	MNIST	Kuzushiji-MNIST	Fashion-MNIST	CIFAR-10 ResNet
RC	**99.46±0.02%**	98.69±0.03%	**94.32±0.07%**	**86.77±0.47%**
CC	99.43±0.03%	**98.78±0.01%**	94.31±0.17%	85.38±0.16%•
GA	95.58±0.02%•	97.13±0.02%•	89.33±0.03%•	75.38±0.23%•
NN	98.72±0.04%•	96.99±0.06%•	90.35±0.19%•	75.12±0.25%•
Free	79.98±2.03%•	84.01±1.36%•	75.03±3.95%•	46.65±0.35%•
PC	95.32±0.13%•	90.80±0.12%•	85.39±0.18%•	55.68±2.30%•
Forward	99.25±0.04%•	98.72±0.06%	92.77±0.23%•	78.74±1.41%•
EXP	99.27±0.01%•	98.38±0.11%•	93.23±0.04%•	79.84±1.22%•
LOG	99.38±0.09%	98.75±0.06%	93.52±0.07%	84.10±0.54%•
MAE	99.29±0.03%•	98.47±0.17%•	90.10±3.41%•	74.05±0.87%•
MSE	98.71±0.03%•	95.53±0.17%•	90.81±0.18%•	79.12±0.40%•
GCE	98.84±0.02%•	97.48±0.16%•	91.72±0.08%•	79.47±0.38%•
Phuber-CE	97.31±0.07%•	92.44±1.19%•	88.94±0.11%•	70.73±0.39%•

we can directly use CLL methods. 2) Existing CLL methods can be applied to large-scale datasets. The compared CLL methods include GA, NN, and Free [36], PC [35], Forward [62], the unbiased risk estimator [37] with bounded losses MAE, MSE, GCE, Phuber-CE, and the surrogate losses EXP and LOG. The detailed information of these compared CLL methods is listed as follows:

- PC [35]: It utilizes the pairwise comparison strategy (with sigmoid loss) in the multi-class loss function to learn from complementarily labeled data.
- Forward [62]: It conducts forward correction by estimating the latent class transition probability matrix to learn from complementarily labeled data.
- Free, NN, GA [36]: These are three methods adapted from the same unbiased risk estimator for learning from complementarily labeled data. For the Free method, it minimizes the original empirical risk estimator. For the NN method, it corrects the negative term in the risk estimator using max operator. For the GA method, it uses a gradient ascent strategy to prevent from overfitting.
- MAE, MSE, GCE, Phuber-CE [37]: These are four methods that insert conventional bounded multi-class loss functions into the unbised risk estimator for learning with multiple complementary labels.

Table 6.7: Test accuracy (mean±std) of each method using linear model on UCI datasets.

	Texture	Yeast	Dermatology	Har	20Newsgroups
RC	99.24±0.14%	59.89±1.27%	99.41±1.00%	98.03±0.09%	**75.99±0.53%**
CC	98.02±2.91%•	**59.97±1.57%**	**99.73±0.85%**	**98.10±0.18%**	75.97±0.54%
SURE	95.38±0.28%•	54.39±1.32%•	97.48±0.32%•	97.43±0.24%•	69.82±0.26%•
CLPL	91.93±0.97%•	54.58±2.11%•	99.62±0.85%	97.48±0.18%•	71.44±0.55%•
PLECOC	69.69±4.82%•	37.37±9.73%•	87.84±5.30%•	96.97±0.29%•	15.32±7.86%•
PLSVM	49.38±9.99%•	45.70±8.01%•	80.00±7.53%•	91.64±1.43%•	32.59±8.91%•
PLKNN	96.78±0.31%•	47.79±2.41%•	80.54±5.06%•	94.17±0.59%•	27.18±0.65%•
IPAL	**99.45±0.23%**	48.99±3.84%•	98.65±2.27%•	96.55±0.40%•	48.36±0.85%•

Table 6.8: Test accuracy (mean±std) of each method using linear model on real-world datasets.

	Lost	MSRCv2	BirdSong	Soccer Player	Yahoo! News
RC	**79.43±3.26%**	46.56±2.71%	71.94±1.72%	**57.00±0.97%**	**68.23±0.83%**
CC	79.29±3.19%	47.22±3.02%	**72.22±1.71%**	56.32±0.64%	68.14±0.81%
SURE	71.33±3.57%•	46.88±4.67%	58.92±1.28%•	49.41±086%•	45.49±1.15%•
CLPL	74.87±4.30%•	36.53±4.59%•	63.56±1.40%•	36.82±1.04%•	46.21±0.90%•
PLECOC	49.03±8.36%•	41.53±3.25%•	71.58±1.81%	53.70±2.02%•	66.22±1.01%•
PLSVM	75.31±3.81%•	35.85±4.41%•	49.90±2.07%•	46.29±0.96%•	56.85±0.91%•
PLKNN	36.73±2.99%•	41.36±2.89%•	64.94±1.42%•	49.62±0.67%•	41.07±1.02%•
IPAL	72.12±4.48%•	**50.80±4.46%**◦	72.06±1.55%	55.03±0.77%•	66.79±1.22%•

- EXP, LOG [37]: They are two methods for learning with multiple complementary labels. For these two methods, upper-bound surrogate loss functions are used in the derived empirical risk estimator [37].

Hyper-parameters for all the CLL methods are selected so as to maximize the accuracy on a validation set, which is constructed by randomly sampling 10% of the training set.

For our proposed methods RC (Algorithm 3) and CC (Algorithm 4), we only need to search learning rate and weight decay from $\{10^{-6}, \ldots, 10^{-1}\}$, since there are no other hyper-parameters in our methods. Hyper-parameters are selected so as to maximize the accuracy on a validation set (10% of the training set) of partially labeled data. We implement them using PyTorch [100] and use the Adam [101] optimizer with the mini-batch size set to 256 and the number of epochs set to 250. For all the parametric methods, we adopt the same base model for fair comparisons.

6.5.2 Experimental Results

We run 5 trials on the four benchmark datasets and run 10 trials (with 90%/10% train/test split) on UCI datasets and real-world partially labeled datasets, and record

the mean accuracy with standard deviation (mean±std). We also use paired t-test at 5% significance level, and $\bullet$/$\circ$ represents whether the *best* of RC and CC is significantly better/worse than other compared methods. Besides, the best results are highlighted in bold. Table 6.3 and Table 6.4 report the test performance of each method using neural networks on benchmark datasets. From the two tables, we can observe that RC always achieves the best performance and significantly outperforms other compared methods in most cases.

Besides, we also record the transductive performance (accuracy of the training set evaluated by the true labels) of each method. Table 6.5 and 6.6 report the transductive accuracy of each method using different neural networks on benchmark datasets. As shown in the two tables, our proposed methods RC and CC still significantly outperform other compared methods in most cases. In addition, it is worth noting that the gap of transductive accuracy between RC and CC is not so significant. However, as shown in Table 6.3 and Table 6.4, the gap of test accuracy between RC and CC is quite significant. These observations further support our conjecture that the estimation error bound of RC is probably tighter than that of CC.

In addition, we record the test accuracy and transductive accuracy at each training epoch to provide more detailed visualized results in Fig. 6.1 and Fig. 6.2. Here, we record the test accuracy at each training epoch to provide more detailed visualized results. To avoid the overcrowding of many curves in a single figure, we only use seven methods including RC, CC, GA, NN, Free, PC, and Forward. The linear model and the MLP model are trained on the benchmark datasets. Fig. 6.1 and Fig. 6.2 reports the experimental results of the seven methods for different datasets and models. Dark colors show the mean accuracy of 5 trials and light colors show the standard deviation. As shown in Fig. 6.1 and Fig. 6.2, our proposed partial-label learning methods RC and CC still consistently outperform other compared methods, even when the simple linear model is used.

Table 6.7 and Table 6.8 report the test performance of each method using linear model on UCI datasets and real-world partially labeled datasets, respectively. We can find that RC and CC generally achieve superior performance against other compared methods on both UCI datasets and real-world partially labeled datasets.

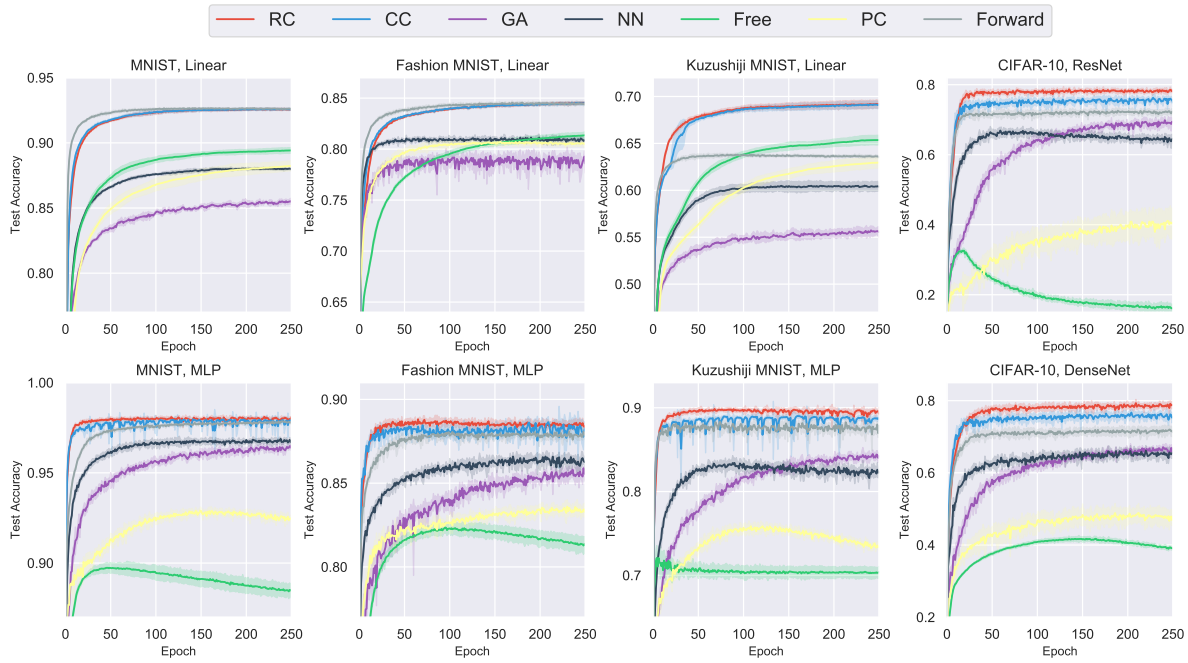


Fig. 6.1: Classification accuracy (mean test accuracy with standard deviation of 5 trials) of different methods on different datasets.

Performance Comparison Between RC and CC. It can be seen that when linear model is used, RC and CC achieve similar performance. However, RC significantly outperforms CC when deep neural networks are used. These observations accord with our derived estimation error bounds for RC and CC, i.e., RC achieves notably smaller estimation error than CC when complex models are used.

6.5.3 Effectiveness of Generation Model

We use *entropy* to measure how well given candidate label sets match the proposed generation model. By this measure, we could know ahead of model training whether to apply our proposed methods or not on a specific dataset. We expect that the higher the entropy, the better the match, thus the better the performance of our proposed methods. To verify our conjecture, we generate various candidate labels sets by different generation models, and the experimental results agree with our conjecture. We further show via experiments that even when given candidate label sets do not match our proposed generation model well, our methods still significantly outperform other compared methods. These experimental results are provided in Appendix F.

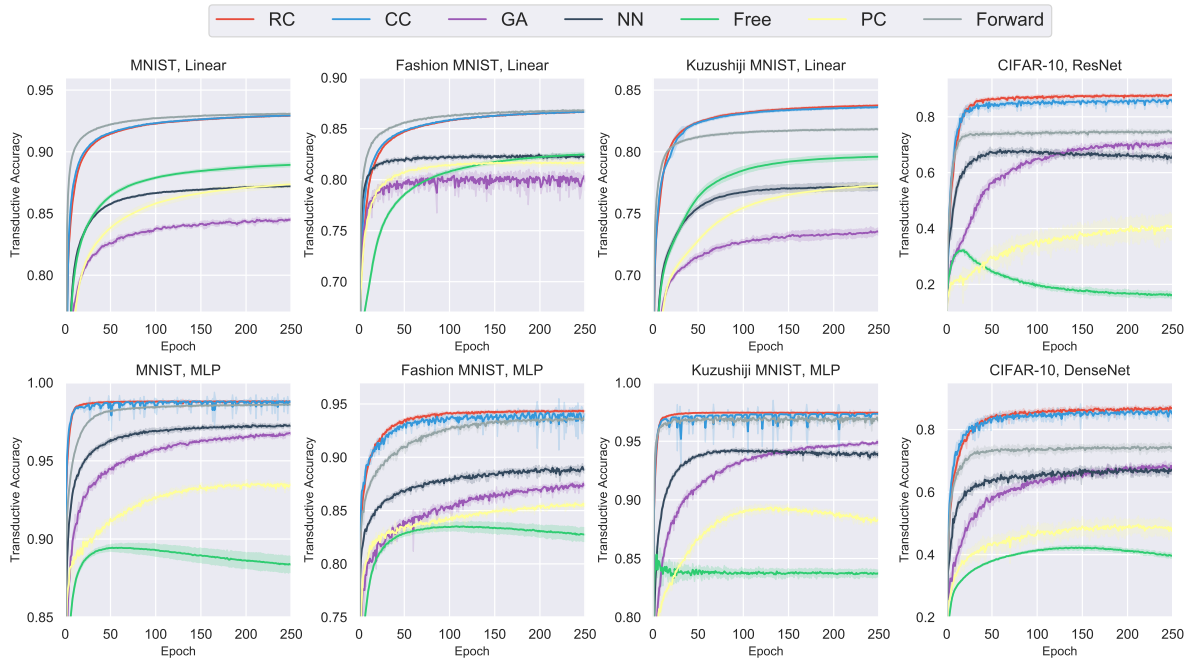


Fig. 6.2: Classification accuracy (mean transductive accuracy with standard deviation of 5 trials) of different methods on different datasets.

Table 6.9: Test accuracy (mean±std%) of the RC method using neural networks on benchmark datasets with different generation models. The best performance is highlighted in bold.

	Case 1	Case 2	Case 3	Case 4	Case 5	Our Case	Supervised
MLP MNIST	95.29• (±0.14)	97.17• (±0.04)	97.68• (±0.10)	97.93 (±0.15)	98.97 (±0.12)	**98.00** (±0.11)	98.48 (±0.00)
MLP KMNIST	79.88• (±0.47)	85.65• (±0.38)	88.04• (±0.37)	89.07• (±0.20)	89.34 (±0.18)	**89.38** (±0.21)	91.53 (±0.00)
MLP FMNIST	79.78• (±0.32)	84.97• (±0.32)	87.05• (±0.17)	88.09• (±0.18)	88.27 (±0.24)	**88.38** (±0.23)	89.37 (±0.00)
LeNet MNIST	98.82• (±0.05)	99.02 (±0.06)	99.02 (±0.06)	**99.04** (±0.08)	**99.04** (±0.05)	**99.04** (±0.08)	99.22 (±0.00)
LeNet KMNIST	92.81• (±0.39)	93.54• (±0.21)	93.71• (±0.20)	93.77• (±0.23)	93.89 (±0.25)	**94.00** (±0.31)	95.34 (±0.00)
LeNet FMNIST	81.59• (±0.18)	86.49• (±0.31)	88.48• (±0.15)	89.24• (±0.11)	89.45 (±0.18)	**89.48** (±0.11)	89.93 (±0.00)

Here, we would like to test the performance of our methods under different data generation processes. As indicated before, our proposed partial-label learning methods are based on the proposed data generation model. Therefore, we would like to investigate

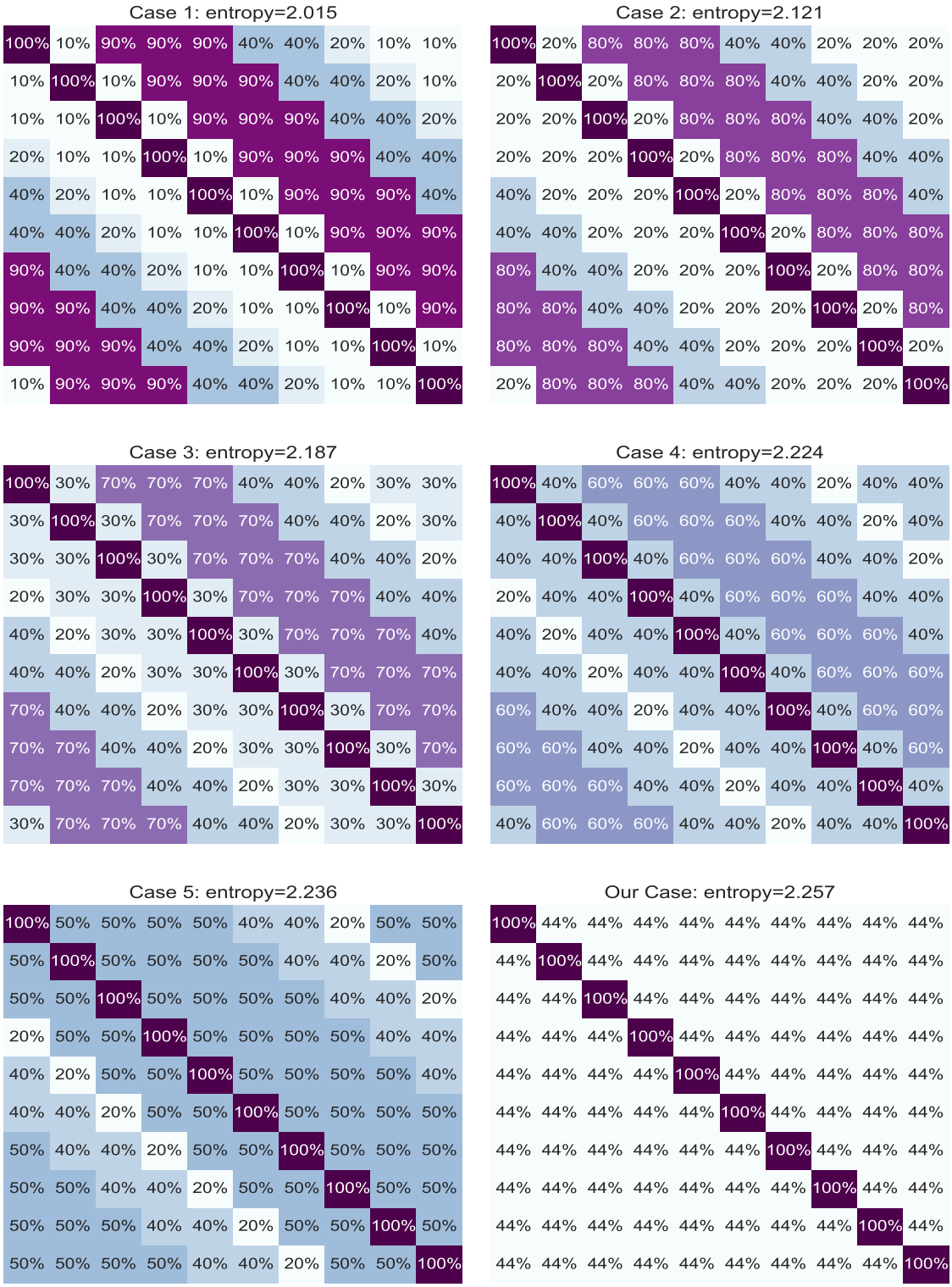


Fig. 6.3: Heatmaps of different generation processes of candidate label sets.

Table 6.10: Test accuracy (mean±std%) of the CC method using neural networks on benchmark datasets with different generation models. The best performance is highlighted in bold.

	Case 1	Case 2	Case 3	Case 4	Case 5	Our Case	Supervised
MLP MNIST	96.36• (±0.17)	97.49• (±0.10)	97.76 (±0.12)	97.85 (±0.08)	**97.87** (±0.17)	**97.87** (±0.10)	98.48 (±0.00)
MLP KMNIST	80.65• (±0.86)	86.43• (±0.80)	88.06• (±0.57)	88.69 (±0.21)	88.73 (±0.44)	**88.83** (±0.40)	91.53 (±0.00)
MLP FMNIST	79.81• (±0.45)	84.49• (±0.33)	86.47• (±0.14)	87.52• (±0.15)	87.64 (±0.18)	**87.80** (±0.25)	89.37 (±0.00)
LeNet MNIST	98.28• (±0.19)	98.83• (±0.08)	98.93 (±0.07)	98.94 (±0.02)	98.95 (±0.09)	**98.99** (±0.08)	99.22 (±0.00)
LeNet KMNIST	86.67• (±1.22)	92.16• (±0.30)	93.13• (±0.26)	93.41• (±0.30)	93.81 (±0.22)	**93.86** (±0.18)	95.34 (±0.00)
LeNet FMNIST	77.75• (±5.32)	86.11• (±0.31)	87.86• (±0.20)	88.53• (±0.31)	88.97 (±0.25)	**88.98** (±0.20)	89.93 (±0.00)

the influence of different generation models on our proposed methods. We use *entropy* to measure how well given candidate label sets match the proposed generation model. By this measure, we could know ahead of model training whether to apply our proposed methods or not on a specific dataset. We expect that the higher the entropy, the better the match, thus the better the performance of our proposed methods. To verify our conjecture, we generate various candidate labels sets by different generation models. It is worth noting that the averaged number of candidate labels per instance (Avg. #CLs) plays an important role in partially labeled datasets. Intuitively, the performance of partial-label learning methods would generally be better if trained on the datasets with smaller Avg. #CLs. The Avg. #CLs of our generation model is 5. Therefore, to keep fair comparisons, the Avg. #CLs of other studied generation models is also kept as 5.

In following experiments, we still focus on the case where the candidate label set is independent of the instance. We additionally introduce the *class transition matrix* (denoted by $\boldsymbol{T}$) for partially labeled data, where T_{ij} describes the probability of the label j being a candidate label given the true label i for each instance. Intuitively, $T_{ii} = 1$ always holds since the true label is always a candidate label. In this way, we provide various formulations of the matrix $\boldsymbol{T}$ to instantiate different generation models.

The studied generation models are illustrated in Figure 6.3. As shown in Figure 6.3, we provide six cases of generation models, and each of them holds a value of entropy.

Table 6.11: Test accuracy (mean±std) of each method using neural networks on benchmark datasets. DenseNet is trained on CIFAR-10, and LeNet is trained on the other three datasets. Candidate label sets are generated by the generation model in Case 1 (entropy=2.015).

	MNIST	Kuzushiji-MNIST	Fashion-MNIST	CIFAR-10
RC	**98.82±0.05%**	**92.81±0.39%**	**81.59±0.18%**	**68.18±0.60%**
CC	98.28±0.19%•	86.67±1.22%•	77.75±5.32%•	56.13±3.33%•
GA	97.29±0.19%•	83.79±0.98%•	70.91±0.99%•	41.57±1.35%•
NN	69.51±2.06%•	51.03±1.88%•	53.13±2.04%•	31.54±1.65%•
Free	15.29±0.58%•	13.60±0.37%•	10.58±0.54%•	12.53±0.34%•
PC	96.56±0.25%•	85.60±0.45%•	80.98±0.44%	65.97±0.39%•
Forward	95.87±4.82%•	90.83±0.82%•	59.66±2.75%•	51.25±0.49%•
EXP	84.37±9.30%•	71.10±5.74%•	59.56±8.43%•	30.35±0.38%•
LOG	98.17±0.10%•	87.85±0.82%•	77.50±5.12%•	54.61±4.04%•
MAE	56.81±8.36%•	49.78±9.03%•	36.41±0.29%•	30.61±0.43%•
MSE	95.80±0.24%•	74.95±0.84%•	58.85±3.52%•	58.18±1.25%•
GCE	95.92±0.09%•	80.49±1.10%•	72.25±0.35%•	57.47±0.59%•
Phuber-CE	79.41±1.61%•	59.88±1.06%•	58.65±1.22%•	57.53±3.36%•

The value of entropy is calculated by the following two steps: 1) The matrix $\boldsymbol{T}$ is normalized by $\boldsymbol{P}_{ij} = \boldsymbol{T}_{ij}/(\sum_j \boldsymbol{T}_{ij}),\ \forall i, j \in [k]$. 2) The entropy of the case is calculated by $-\frac{1}{k}\sum_{i=1}^{k}\sum_{j=1}^{k}\boldsymbol{P}_{ij}\log\boldsymbol{P}_{ij}$. As in our proposed generation model, given the true label, other labels have the same probability to be a candidate label, our case achieves the maximum entropy (i.e., 2.257).

Table 6.9 and Table 6.10 report the test performance (mean±std) of the RC method and the CC method using neural networks on benchmark datasets with different generation models. From the two tables, we can observe that the higher the entropy, the better the match, thus the better the performance of our proposed methods. Thus, our conjecture is clearly validated.

We further conduct experiments with the generation model of Case 1 where given candidate label sets do not match our proposed generation model well. The experimental results are shown in Table 6.11. As can be seen from Table 6.11, our proposed methods still significantly outperform other compared methods in most cases, and RC always achieves the best performance.

6.6 Chapter Summary

In this chapter, we for the first time provided an explicit mathematical formulation of the partially labeled data generation process for partial-label learning. Based on our data generation model, we further derived a novel *risk-consistent* method and a novel *classifier-consistent* method. To the best of our knowledge, we provided the first risk-consistent partial-label learning method. Besides, our proposed methods do not reply on specific models and can be easily trained with stochastic optimization, which ensures their scalability to large-scale datasets. In addition, we theoretically derived an *estimation error bound* for each of the proposed methods. Finally, extensive experimental results clearly demonstrated the effectiveness of the proposed generation model and two partial-label learning methods.

Chapter 7

Conclusions and Future Directions

7.1 Conclusions

Weakly supervised learning has attracted increasing attention in recent years. In this thesis, we studied two advanced topics in weakly supervised learning, i.e., *complementary-label learning* and *partial-label learning*.

In Chapter 3, we proposed an extended problem setting to allow multiple complementary labels for each example and two ways for learning with multiple complementary labels. The first way allows us to use any existing complementary-label learning method by decomposing a set of complementary labels into many single complementary labels. However, we find that such decomposition would conceptually dilute the supervision information. Hence the second way alleviates this problem by resorting to an unbiased risk estimator, minimizing which processes each set of complementary labels as a whole and possesses an estimation error bound. We further improved this way into minimizing properly chosen upper bounds for practical implementation. Experiments demonstrated the effectiveness of the proposed methods.

In Chapter 4, we formalized the confidences of candidate labels being the ground-truth label as latent label distributions, and proposed a novel unified framework to estimate the latent label distributions while training the model simultaneously. We presented a biconvex formulation with constrained local consistency and adopted an alternating method to solve this optimization problem. The process of alternating optimization exactly facilitates the mutual adaption of the model training and the constrained label propagation. Experiments demonstrated the effectiveness of the proposed method.

In Chapter 5, we provided the first attempt to improve the popular self-training strategy for partial-label learning. We proposed a novel unified formulation the maximum infinity norm regularization to train an effective model in conjunction with pseudo-labeling. Instead of simply selecting the correct label with high predicting confidence, we impose the maximum infinity norm regularization in the model training process to perform effective pseudo-labeling. To optimize the final objective function, we encountered a convex-concave problem. We demonstrated that the optimization of the convex-concave problem would be consistent with the optimization of multiple *quadratic programming* (QP) problems. In order to speedup the optimization process, we further proposed a surrogate objective for practical implementation. Experiments demonstrated the effectiveness of the proposed method.

In Chapter 6, we proposed the first generation model of candidate label sets, and developed two novel partial-label learning methods that are guaranteed to be provably consistent, i.e., one is *risk-consistent* (RC) and the other is *classifier-consistent* (CC). To the best of our knowledge, we provided the first risk-consistent method for partial-label learning. Our methods are advantageous, since they are agnostic in specific classification models and can be easily trained with stochastic optimization, which ensures their scalability to large-scale datasets. Experiments demonstrated effectiveness of the proposed generation model and two partial-label learning methods.

In this thesis, we have investigated learning with multiple complementary labels, which is exactly a generalization of previous studies on ordinary CLL. Although we could simply regard learning with multiple complementary labels as PLL by taking complementary labels as non-candidate labels, the proposed methods for learning with multiple complementary labels are not directly designed for PLL. We have also presented four PLL methods (including LALO, SURE, CC, and RC). Regarding experimental performance: when simple models are used (e.g., linear model or kernel model), RC and CC achieve similar performance, RC and CC outperform SURE and LALO, and SURE outperforms LALO; when complex models are used (e.g., deep neural networks), RC outperforms CC while SURE and LALO are not directly compatible with other complex models and large-scale datasets. Regarding learning consistency: RC is a risk-consistent method, CC is a classifier-consistent method, while SURE and LALO do not hold learning consistency.

7.2 Future Directions

In the final section of this thesis, we show some potentially interesting future directions of weakly supervised learning.

7.2.1 Online Weakly Supervised Learning

In *online learning* [117–122], the learner aims to learn and update the classifier for processing the training examples that arrive in a sequential order at each step. Most of the existing weakly supervised learning methods work on a batch learning or offline learning mode, which cannot deal with the online learning setting with sequential data. In many real-world scenarios, the training data with weak supervision is collected in sequence, hence traditional batch learning methods that require all training data to be provided at once would not work. Therefore, online weakly supervised learning would be a promising technique to solve this problem. There are very few online weakly supervised learning methods proposed for *positive-unlabeled learning* [123], *complementary-label learning* [84], and *partial-label learning* [124], *noisy-label learning* [125], and *multiple-instance learning* [126]. We believe that online weakly supervised metric learning is worth a further study in the future.

7.2.2 Weakly Supervised Metric Learning

Metric learning [127–131] is a branch of machine learning that aims to learn a good distance metric for capturing the latent relationships among instances. The conventional setting of metric learning normally requires the assumption that all the examples are provided with the correct label. However, in many real-world scenarios, we may only collect training examples with weak supervision. Therefore, how to learn a good distance metric with weak supervision is practically important. Although very few studies have investigated metric learning in the presence of noisy labels [132] and logistic labels [133], we think that more weakly supervised learning frameworks could be further studied.

7.2.3 Weakly Supervised Domain Adaptation

Domain adaptation [134–140] aims to transfer knowledge in the presence of the domain gap. Most of the existing studies mainly focus on the three types of domain adaptation:

supervised domain adaptation where the data in the target domain are fully labeled, *semi-supervised domain adaptation* where a small fraction of labeled data and a large fraction of unlabeled data exist in the target domain, and *unsupervised domain adaptation* where the data in the target domain are fully unlabeled. We would like to advocate an interesting setting called weakly supervised domain adaptation where the data in the source domain has only weak supervision and the data in the target domain are fully unlabeled. To this end, There have been several studies on weakly supervised domain adaption. For example, a transferable curriculum learning method [141] has been proposed to train deep networks under the weakly supervised domain domain adaptation setting where the data in the source data has label noise and the data in the target domain are fully unlabeled. Such setting has also investigated by [142]. Besides, another weakly supervised domain adaptation setting where the data in the source domain are complementarily labeled and the data in the target domain are fully unlabeled has also been studied [143, 144]. Therefore, we suggest that more types of weak supervision in the source domain would be valuable to be studied.

Appendices

Appendix A

Mathematical Proofs in Chapter 3

A.1 Proof of Theorem 3.4

Recall that the expected risk and empirical risk are represented as

$$R(f) = \sum_{j=1}^{k-1} p(s=j)\overline{R}_j(f) = \sum_{j=1}^{k-1} p(s=j)\mathbb{E}_{p(\boldsymbol{x},\overline{Y}|s=j)}[\overline{\mathcal{L}}_j\big(f(\boldsymbol{x}),\overline{Y}\big)],$$
$$\widehat{R}(f) = \sum_{j=1}^{k-1} \frac{p(s=j)}{n_j} \sum_{i=1}^{n_j} \overline{\mathcal{L}}_j\big(f(\boldsymbol{x}_i),\overline{Y}_i\big).$$

Here, with a slight abuse of notation, we simply write $\overline{R}_j(f)$ as $R_j(f)$, and define $\widehat{R}_j(f) = 1/n_j \sum_{i=1}^{n_j} \overline{\mathcal{L}}_j\big(f(\boldsymbol{x}_i),\overline{Y}_i\big)$. Thus we have $R(f) = \sum_{j=1}^{k-1} p(s=j)R_j(f)$ and $\widehat{R}(f) = \sum_{j=1}^{k-1} p(s=j)\widehat{R}_j(f)$. Since $f^\star = \arg\min_{f\in\mathcal{F}} R(f)$ and $\widehat{f} = \arg\min_{f\in\mathcal{F}} \widehat{R}(f)$, we can obtain the following lemma.

Lemma A.1 *The following inequality holds:*

$$R(\widehat{f}) - R(f^\star) \leq 2\sum_{j=1}^{k-1} p(s=j) \sup_{f\in\mathcal{F}} \left|\widehat{R}_j(f) - R_j(f)\right|.$$

Proof. It would be intuitive to obtain

$$\begin{aligned}
R(\widehat{f}) - R(f^\star) =& R(\widehat{f}) - \widehat{R}(\widehat{f}) + \widehat{R}(\widehat{f}) - R(f^\star) \\
\leq& R(\widehat{f}) - \widehat{R}(\widehat{f}) + R(\widehat{f}) - R(f^\star) \\
\leq& 2\sup_{f\in\mathcal{F}} \left|\widehat{R}(f) - R(f)\right| \\
=& 2\sup_{f\in\mathcal{F}} \left|\sum_{j=1}^{k-1} p(s=j)\widehat{R}_j(f) - \sum_{j=1}^{k-1} p(s=j)R_j(f)\right| \\
\leq& 2\sum_{j=1}^{k-1} p(s=j) \sup_{f\in\mathcal{F}} \left|\widehat{R}_j(f) - R_j(f)\right|,
\end{aligned}$$

which concludes the proof of Lemma A.1. ■

Then, we will bound $\sup_{f\in\mathcal{F}}\left|\widehat{R}_j(f) - R_j(f)\right|$ for $j = \{1,\dots,k-1\}$. Before that, we define a function space as

$$\mathcal{H}_j := \{(\boldsymbol{x},\overline{Y}) \in \mathcal{X}\times\overline{\mathcal{Y}}_j \mapsto \overline{\mathcal{L}}_j\big(f(\boldsymbol{x}),\overline{Y}\big) \mid f\in\mathcal{F}\},$$

where

$$\overline{\mathcal{L}}_j\big(f(\boldsymbol{x}),\overline{Y}\big) := \sum\nolimits_{y\notin\overline{Y}} \mathcal{L}\big(f(\boldsymbol{x}),y\big) - \frac{k-1-j}{j}\sum\nolimits_{y'\in\overline{Y}} \mathcal{L}\big(f(\boldsymbol{x}),y'\big).$$

Since our proof is based on *Rademacher complexity* [85], we first give its definition.

Definition 1 (Rademacher complexity [85]) *Let $Z_1,\dots,Z_n$ be n independent and identically distributed random variables sampled from a probability distribution $\mathcal{D}$, $\mathcal{H} = \{h : \mathcal{Z}\to\mathbb{R}\}$ be a class of measurable functions. Then the expected Rademacher complexity of $\mathcal{H}$ is defined as*

$$\mathfrak{R}_n(\mathcal{H}) = \mathbb{E}_{Z_1,\dots,Z_n\sim\mathcal{D}}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum\nolimits_{i=1}^{n}\sigma_i h(Z_i)\right],$$

where $\boldsymbol{\sigma} = (\sigma_1,\dots,\sigma_n)$ are Rademacher variables taking the value from $\{-1,+1\}$ with even probabilities.

Then, we have the following lemma.

Lemma A.2 *Let $C_{\mathcal{L}} = \sup_{\boldsymbol{x}\in\mathcal{X},f\in\mathcal{F},y\in\mathcal{Y}}\mathcal{L}\big(f(\boldsymbol{x}),y\big)$. Then, for all $j = \{1,\dots,k-1\}$, for any $\delta > 0$, with probability at least $1-\delta$,*

$$\sup_{f\in\mathcal{F}}\left|\widehat{R}_j(f) - R_j(f)\right| \le 2\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) + (2k-2j-1)C_{\mathcal{L}}\sqrt{\frac{\log\frac{2}{\delta}}{2n_j}},$$

where

$$\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) = \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in\mathcal{H}_j}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i h(\boldsymbol{x}_i,\overline{Y}_i)\right].$$

Proof. To prove this lemma, we first show that the single direction $\sup_{f\in\mathcal{F}}\big(\widehat{R}_j(f) - R_j(f)\big)$ is bounded with probability at least $1-\frac{\delta}{2}$, and the other direction can be similarly proved. By the definition of $\overline{\mathcal{L}}_j$, we can easily know the possible maximum of $\overline{\mathcal{L}}_j$ is

$(k-j)C_{\mathcal{L}}$, and the possible minimum is $-(k-1-j)C_{\mathcal{L}}$. Suppose an example $(\boldsymbol{x}_i, \overline{Y}_i)$ is replaced by another arbitrary example $(\boldsymbol{x}_i', \overline{Y}_i')$, then the change of $\sup_{f\in\mathcal{F}}\left(\widehat{R}_j(f) - R_j(f)\right)$ is no greater than $((2k-2j-1)C_{\mathcal{L}})/n_j$. Then, by applying *McDiarmid's inequality* [145], for any $\delta > 0$, with probability at least $1-\frac{\delta}{2}$,

$$\sup_{f\in\mathcal{F}}\left(\widehat{R}_j(f) - R_j(f)\right) \leq \mathbb{E}\Big[\sup_{f\in\mathcal{F}}\left(\widehat{R}_j(f) - R_j(f)\right)\Big] + (2k-2j-1)C_{\mathcal{L}}\sqrt{\frac{\log\frac{2}{\delta}}{2n_j}}. \tag{A.1}$$

In addition, it is routine [146] to show

$$\mathbb{E}\Big[\sup_{f\in\mathcal{F}}\left(\widehat{R}_j(f) - R_j(f)\right)\Big] \leq 2\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j), \tag{A.2}$$

Combing Eq. (A.1) and Eq. (A.2), we have for any $\delta > 0$, with probability at least $1-\frac{\delta}{2}$,

$$\sup_{f\in\mathcal{F}}\left(\widehat{R}_j(f) - R_j(f)\right) \leq 2\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) + (2k-2j-1)C_{\mathcal{L}}\sqrt{\frac{\log\frac{2}{\delta}}{2n_j}}.$$

By further taking into account the other side $\sup_{f\in\mathcal{F}}\left(R_j(f) - \widehat{R}_j(f)\right)$, we have for any $\delta > 0$, with probability at least $1-\delta$,

$$\sup_{f\in\mathcal{F}}\left|\widehat{R}_j(f) - R_j(f)\right| \leq 2\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) + (2k-2j-1)C_{\mathcal{L}}\sqrt{\frac{\log\frac{2}{\delta}}{2n_j}}.$$

which concludes the proof of Lemma A.2. ■

Next, we will bound the expected Rademacher complexity of the function space $\mathcal{H}_j$, i.e., $\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j)$.

Lemma A.3 *Assume the loss function $\mathcal{L}\big(f(\boldsymbol{x}), y\big)$ is ρ-Lipschitz with respect to $f(\boldsymbol{x})$ $(0 < \rho < \infty)$ for all $y \in \mathcal{Y}$. Then, for all $j = \{1, \ldots, k-1\}$, the following inequality holds:*

$$\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) \leq \frac{\rho(k-1)}{j}\sum\nolimits_{y=1}^{k}\mathfrak{R}_{n_j}(\mathcal{G}_y),$$

where

$$\begin{aligned}\mathcal{G}_y =& \{g : \boldsymbol{x} \mapsto f_y(\boldsymbol{x}) \mid f \in \mathcal{F}\},\\ \mathfrak{R}_{n_j}(\mathcal{G}_y) =& \mathbb{E}_{\boldsymbol{x}_i\sim p(\boldsymbol{x})}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{g\in\mathcal{G}_y}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}g(\boldsymbol{x}_i)\right].\end{aligned}$$

Proof. The expected Rademacher complexity of $\mathcal{H}_j$ can be expressed as

$$\begin{aligned}
\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) =& \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in\mathcal{H}_j}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i h(\boldsymbol{x}_i,\overline{Y}_i)\right] \\
=& \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\sum\nolimits_{y\notin\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right.\right. \\
& \left.\left. -\frac{k-j-1}{j}\sum\nolimits_{y'\in\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\right)\right] \\
\leq& \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\sum\nolimits_{y\notin\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right] \\
& + \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\frac{k-j-1}{j}\sum\nolimits_{y'\in\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\right)\right].
\end{aligned}$$

Here, we introduce random variables $\alpha_{i,y} = \mathbb{I}[y \in \overline{Y}_i],\ \forall i \in \{1,\cdots,n\}, y \in \mathcal{Y}$, where $\mathbb{I}[\cdot]$ denotes the indicator function. In other words, given a complementary label set $\overline{Y}_i$, if a specific label y satisfies the condition $y \in \overline{Y}_i$, then $\mathbb{I}[y \in \overline{Y}_i] = 1$, otherwise $\mathbb{I}[y \in \overline{Y}_i] = 0$. Then, we can obtain

$$\begin{aligned}
& \overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) \\
\leq& \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\sum\nolimits_{y\notin\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right] \\
& + \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\frac{k-j-1}{j}\sum\nolimits_{y'\in\overline{Y}_i}\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\right)\right] \\
=& \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\sum\nolimits_{y=1}^{k}(1-\alpha_{i,y})\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right] \\
& + \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\frac{k-j-1}{j}\sum\nolimits_{y=1}^{k}\alpha_{i,y}\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right] \\
=& \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\sum\nolimits_{y=1}^{k}\frac{1}{2}(1-2\alpha_{i,y}+1)\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right] \\
& + \mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum\nolimits_{i=1}^{n_j}\sigma_i\left(\frac{k-j-1}{j}\sum\nolimits_{y=1}^{k}\frac{1}{2}(2\alpha_{i,y}-1+1)\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right].
\end{aligned}$$

Thus, we have

$$
\begin{aligned}
&\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) \\
\leq &\mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{2n_j}\sum_{i=1}^{n_j}\left(\sum_{y=1}^{k}(1-2\alpha_{i,y})\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y\big)+\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right] \\
&+\mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{2n_j}\sum_{i=1}^{n_j}\left(\frac{k-j-1}{j}\sum_{y=1}^{k}(2\alpha_{i,y}-1)\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\right.\right. \\
&\qquad\left.\left.+\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\right)\right].
\end{aligned}
$$

Here, because $(1-2\alpha_{i,y})\sigma_i$ and $(2\alpha_{i,y}-1)\sigma_i$, and σ_i follow the same distribution, we have

$$
\begin{aligned}
\overline{\mathfrak{R}}_{n_j}(\mathcal{H}_j) \leq &\mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{2n_j}\sum_{i=1}^{n_j}\left(\sum_{y=1}^{k}(1-2\alpha_{i,y})\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right.\right. \\
&\qquad\left.\left.+\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y\big)\right)\right] \\
&+\mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{2n_j}\sum_{i=1}^{n_j}\left(\frac{k-j-1}{j}\sum_{y=1}^{k}(2\alpha_{i,y}-1)\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\right.\right. \\
&\qquad\left.\left.+\sigma_i\mathcal{L}\big(f(\boldsymbol{x}),y'\big)\right)\right] \\
\leq &\mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum_{i=1}^{n_j}\sum_{y=1}^{k}\sigma_i\mathcal{L}\big(f(\boldsymbol{x}_i),y\big)\right] \\
&+\mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum_{i=1}^{n_j}\frac{k-j-1}{j}\sum_{y=1}^{k}\sigma_i\mathcal{L}\big(f(\boldsymbol{x}_i),y\big)\right] \\
= &\frac{k-1}{j}\mathbb{E}_{(\boldsymbol{x}_i,\overline{Y}_i)\sim\overline{p}(\boldsymbol{x},\overline{Y}|s=j)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum_{i=1}^{n_j}\sum_{y=1}^{k}\sigma_i\mathcal{L}\big(f(\boldsymbol{x}_i),y\big)\right] \\
\leq &\frac{k-1}{j}\sum_{y=1}^{k}\mathbb{E}_{\boldsymbol{x}_i\sim p(\boldsymbol{x})}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{f\in\mathcal{F}}\frac{1}{n_j}\sum_{i=1}^{n_j}\sigma_i\mathcal{L}\big(f(\boldsymbol{x}_i),y\big)\right] \\
\leq &\frac{k-1}{j}\sum_{y=1}^{k}\mathfrak{R}_{n_j}(\mathcal{L}\circ\mathcal{F}) \\
\leq &\frac{\sqrt{2}\rho k(k-1)}{j}\sum_{y=1}^{k}\mathfrak{R}_{n_j}(\mathcal{G}_y),
\end{aligned}
$$

where we applied the Rademacher vector contraction inequality [147] in the last inequality. ■

Under the assumptions described in the above three lemmas (Lemma A.1, Lemma A.2, and Lemma A.3), for any $\delta > 0$, with probability at least $1 - \delta$,

$$R(\widehat{f}) - R(f^\star) \leq \sum_{j=1}^{k-1} p(s=j)\Bigg(\frac{4\sqrt{2}\rho k(k-1)}{j}\sum_{y=1}^{k}\mathfrak{R}_{n_j}(\mathcal{G}_y) + (4k-4j-2)C_{\mathcal{L}}\sqrt{\frac{\log\frac{2(k-1)}{\delta}}{2n_j}}\Bigg).$$

It is clear that by combining the above three lemmas, Theorem 4 is proved. ■

A.2 Derivations and Boundness of the Used Loss Functions

A.2.1 Derivations of the Used Loss Functions

Conventionally, the label for each instance $\boldsymbol{x}$ is in one-hot encoding. Concretely, if the label of $\boldsymbol{x}$ is y, then we represent the label vector as $\boldsymbol{e}_y$ where $e_{yj} = 1$ if $j = y$, otherwise 0. In this way, we provide the detailed derivations of CCE, MAE, and MSE as follows.

- Categorical Cross Entropy (CCE):

$$\mathcal{L}_{\mathrm{CCE}}(f(\boldsymbol{x}), y) = -\sum_{j=1}^{k} e_{yj}\log p_{\boldsymbol{\theta}}(j|\boldsymbol{x}) = -\log p_{\boldsymbol{\theta}}(y|\boldsymbol{x}).$$

- Mean Absolute Error (MAE):

$$\mathcal{L}_{\mathrm{MAE}}(f(\boldsymbol{x}), y) = \sum_{j=1}^{k}\left|p_{\boldsymbol{\theta}}(j|\boldsymbol{x}) - e_{yj}\right| = 2 - 2p_{\boldsymbol{\theta}}(y|\boldsymbol{x}).$$

- Mean Square Error (MSE):

$$\mathcal{L}_{\mathrm{MSE}}(f(\boldsymbol{x}), y) = \sum_{j=1}^{k}\left(p_{\boldsymbol{\theta}}(j|\boldsymbol{x}) - e_{yj}\right)^2 = 1 - 2p_{\boldsymbol{\theta}}(y|\boldsymbol{x}) + \sum_{j=1}^{k} p_{\boldsymbol{\theta}}(j|\boldsymbol{x})^2.$$

A.2.2 Boundness of the Used Loss Functions

Firstly, it is clear that each loss function is non-negative. Besides, for each loss function, the loss becomes larger if $p_{\boldsymbol{\theta}}(y|\boldsymbol{x})$ gets smaller given the correct label y. Note that $0 < p_{\boldsymbol{\theta}}(y|\boldsymbol{x}) < 1$, hence the upper bound of each loss function is stated as follows.

- MAE: $\mathcal{L}_{\mathrm{MAE}}(f(\boldsymbol{x}), y) < 2$.
- MSE: $\mathcal{L}_{\mathrm{MSE}}(f(\boldsymbol{x}), y) < 1 - 0 + \sum_{j=1}^{k} p_{\boldsymbol{\theta}}(j|\boldsymbol{x})^2 < 2$.
- GCE: $\mathcal{L}_{\mathrm{GCE}}(f(\boldsymbol{x}), y) < 1/q$ where $q = 0.7$.
- PHuber-CE: $\mathcal{L}_{\text{PHuber-CE}}(f(\boldsymbol{x}), y) < \log \tau + 1$ where $\tau = 10$.

Note that for CCE, $\mathcal{L}_{\mathrm{CCE}}(f(\boldsymbol{x}), y) < -\log 0 = \infty$. Therefore, we can know that MAE, MSE, GCE, and PHuber-CE are upper-bounded, while CCE is not upper-bounded.

Appendix B

Mathematical Proofs in Chapter 6

B.1 Proof of Theorem 6.4

Our proof of this theorem is based on *Rademacher complexity* [85]. Before proving Theorem 4, we introduce the following lemma.

Lemma B.4 *Let $\widehat{f}$ be the empirical risk minimizer (i.e., $\widehat{f} = \arg\min_{f\in\mathcal{F}} \widehat{R}(f)$) and $f^\star$ be the true risk minimizer (i.e., $f^\star = \arg\min_{f\in\mathcal{F}} R(f)$), then the following inequality holds:*

$$R(\widehat{f}) - R(f^\star) \leq 2 \sup_{f\in\mathcal{F}} |\widehat{R}(f) - R(f)|.$$

Proof. It is intuitive to obtain

$$\begin{aligned} R(\widehat{f}) - R(f^\star) &\leq R(\widehat{f}) - \widehat{R}(\widehat{f}) + \widehat{R}(\widehat{f}) - R(f^\star) \\ &\leq R(\widehat{f}) - \widehat{R}(\widehat{f}) + R(\widehat{f}) - R(f^\star) \\ &\leq 2 \sup_{f\in\mathcal{F}} |\widehat{R}(f) - R(f)|, \end{aligned}$$

which completes the proof. The same proof has been provided in [146]. ∎

Then, we define a function space for our RC method as

$$\mathcal{G}_{\text{rc}} = \{(\boldsymbol{x}, Y) \mapsto \frac{1}{2} \sum_{i=1}^{k} \frac{p(y = i \mid \boldsymbol{x})}{\sum_{j\in Y} p(y = j \mid \boldsymbol{x})} \mathcal{L}(f(\boldsymbol{x}), i) \mid f \in \mathcal{F}\},$$

where $(\boldsymbol{x}, Y)$ is randomly sampled from $\widetilde{p}(\boldsymbol{x}, Y)$. Let $\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\text{rc}})$ be the expected Rademacher complexity of $\mathcal{G}_{\text{rc}}$, i.e.,

$$\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\text{rc}}) = \mathbb{E}_{\widetilde{p}(\boldsymbol{x},Y)} \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{g\in\mathcal{G}_{\text{rc}}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i g(\boldsymbol{x}_i, Y_i) \right].$$

Then we have the following lemma.

Lemma B.5 *Suppose the loss function $\mathcal{L}$ is bounded by M, i.e., $M = \sup_{\boldsymbol{x}\in\mathcal{X},f\in\mathcal{F},y\in\mathcal{Y}} \mathcal{L}(f(\boldsymbol{x}), y)$, then for any $\delta > 0$, with probability at least $1-\delta$,*

$$\sup_{f\in\mathcal{F}} \left| R_{\mathrm{rc}}(f) - \widehat{R}_{\mathrm{rc}}(f) \right| \leq 2\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{rc}}) + \frac{M}{2}\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Proof. In order to prove this lemma, we first show that the one direction $\sup_{f\in\mathcal{F}} R_{\mathrm{rc}}(f) - \widehat{R}_{\mathrm{rc}}(f)$ is bounded with probability at least $1-\delta/2$, and the other direction can be similarly shown. Suppose an example $(\boldsymbol{x}_i, Y_i)$ is replaced by another arbitrary example $(\boldsymbol{x}_i', Y_i')$, then the change of $\sup_{f\in\mathcal{F}} R_{\mathrm{rc}}(f) - \widehat{R}_{\mathrm{rc}}(f)$ is no greater than $M/(2n)$, since $\mathcal{L}$ is bounded by M. By applying *McDiarmid's inequality* [145], for any $\delta > 0$, with probability at least $1-\delta/2$,

$$\sup_{f\in\mathcal{F}} R_{\mathrm{rc}}(f) - \widehat{R}_{\mathrm{rc}}(f) \leq \mathbb{E}\left[\sup_{f\in\mathcal{F}} R_{\mathrm{rc}}(f) - \widehat{R}_{\mathrm{rc}}(f) \right] + \frac{M}{2}\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Using the same trick in [146], we can obtain

$$\mathbb{E}\left[\sup_{f\in\mathcal{F}} R_{\mathrm{rc}}(f) - \widehat{R}_{\mathrm{rc}}(f) \right] \leq 2\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{rc}}).$$

By further taking into account the other side $\sup_{f\in\mathcal{F}} \widehat{R}_{\mathrm{rc}}(f) - R_{\mathrm{rc}}(f)$, we have for any $\delta > 0$, with probability at least $1-\delta$,

$$\sup_{f\in\mathcal{F}} \left| R_{\mathrm{rc}}(f) - \widehat{R}_{\mathrm{rc}}(f) \right| \leq 2\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{rc}}) + \frac{M}{2}\sqrt{\frac{\log\frac{2}{\delta}}{2n}},$$

which concludes the proof. ∎

Next, we will bound the expected Rademacher complexity of $\mathcal{G}_{\mathrm{rc}}$ (i.e., $\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{rc}})$) by the following lemma.

Lemma B.6 *Assume the loss function $\mathcal{L}(f(\boldsymbol{x}), y)$ is ρ-Lipschitz with respect to $f(\boldsymbol{x})$ $(0 < \rho < \infty)$ for all $y \in \mathcal{Y}$. Then, the following inequality holds:*

$$\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{rc}}) \leq \sqrt{2}\rho \sum_{y=1}^{k} \mathfrak{R}_n(\mathcal{H}_y),$$

where

$$\mathcal{H}_y = \{h : \boldsymbol{x} \mapsto f_y(\boldsymbol{x}) \mid f \in \mathcal{F}\},$$
$$\mathfrak{R}_n(\mathcal{H}_y) = \mathbb{E}_{p(\boldsymbol{x})}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in\mathcal{H}_y} \frac{1}{n}\sum_{i=1}^{n} h(\boldsymbol{x}_i)\right].$$

Proof. First of all, we introduce $p_i(\boldsymbol{x}) = \frac{p(y=i|\boldsymbol{x})}{\sum_{j\in Y} p(y=j|\boldsymbol{x})}$ for each example $(\boldsymbol{x}, Y)$. Thus we have $0 \leq p_i(\boldsymbol{x}) \leq 1, \forall i \in [k]$ and $\sum_{i=1}^{k} p_i(\boldsymbol{x}) = 1$ since $p_i(\boldsymbol{x}) = 0$ if $i \notin Y$. In this way, we can obtain $\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{rc}}) \leq \mathfrak{R}_n(\mathcal{L} \circ \mathcal{F})$ where $\mathcal{L} \circ \mathcal{F}$ denotes $\{\mathcal{L} \circ f \mid f \in \mathcal{F}\}$. Since $\mathcal{H}_y = \{h : \boldsymbol{x} \mapsto f_y(\boldsymbol{x}) \mid f \in \mathcal{F}\}$ and the loss function $\mathcal{L}(f(\boldsymbol{x}), y)$ is ρ-Lipschitz with respect to $f(\boldsymbol{x})$ $(0 < \rho < \infty)$ for all $y \in \mathcal{Y}$, by the Rademacher vector contraction inequality [147], we have $\mathfrak{R}_n(\mathcal{L} \circ \mathcal{F}) \leq \sqrt{2}\rho \sum_{y=1}^{k} \mathfrak{R}_n(\mathcal{H}_y)$, which concludes the proof of Lemma B.6. ∎

Combining Lemma B.4, Lemma B.5, and Lemma B.6, Theorem 4 is proved. ∎

B.2 Proof of Theorem 6.6

Since this proof is somewhat similar to the proof of Theorem 4, we briefly sketch the key points.

We define a function space for our CC method as

$$\mathcal{G}_{\mathrm{cc}} = \{(\boldsymbol{x}, Y) \mapsto \mathcal{L}(q(\boldsymbol{x}), \widetilde{y}) \mid f \in \mathcal{F}\},$$

where $(\boldsymbol{x}, Y)$ is randomly sampled from $\widetilde{p}(\boldsymbol{x}, Y)$ and $C_{\widetilde{y}} = Y$ (i.e., Y is the $\widetilde{y}$-th label set in C). Let $\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{cc}})$ be the expected Rademacher complexity of $\mathcal{G}_{\mathrm{cc}}$, i.e.,

$$\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{cc}}) = \mathbb{E}_{\widetilde{p}(\boldsymbol{x},Y)}\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{g\in\mathcal{G}_{\mathrm{cc}}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i g(\boldsymbol{x}_i, Y_i)\right].$$

Then we have the following lemma.

Lemma B.7 *Suppose the loss function $\mathcal{L}$ is bounded by M, i.e., $M = \sup_{\boldsymbol{x}\in\mathcal{X}, f\in\mathcal{F}, \widetilde{y}\in\widetilde{\mathcal{Y}}} \mathcal{L}(q(\boldsymbol{x}), \widetilde{y})$, then for any $\delta > 0$, with probability at least $1 - \delta$,*

$$\sup_{f\in\mathcal{F}} \left|R_{\mathrm{cc}}(f) - \widehat{R}_{\mathrm{cc}}(f)\right| \leq 2\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{cc}}) + \frac{M}{2}\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Proof. In order to prove this lemma, we first show that the one direction $\sup_{f\in\mathcal{F}} R_{\mathrm{cc}}(f) - \widehat{R}_{\mathrm{cc}}(f)$ is bounded with probability at least $1-\delta/2$, and the other direction can be similarly shown. Suppose an example $(\boldsymbol{x}_i, Y_i)$ is replaced by another arbitrary example $(\boldsymbol{x}_i', Y_i')$, then the change of $\sup_{f\in\mathcal{F}} R_{\mathrm{cc}}(f) - \widehat{R}_{\mathrm{cc}}(f)$ is no greater than M/n, since $\mathcal{L}$ is bounded by M. By applying *McDiarmid's inequality* [145], for any $\delta > 0$, with probability at least $1-\delta/2$,

$$\sup_{f\in\mathcal{F}} R_{\mathrm{cc}}(f) - \widehat{R}_{\mathrm{cc}}(f) \leq \mathbb{E}\left[\sup_{f\in\mathcal{F}} R_{\mathrm{cc}}(f) - \widehat{R}_{\mathrm{cc}}(f)\right] + M\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Using the same trick in [146], we can obtain $\mathbb{E}[\sup_{f\in\mathcal{F}} R_{\mathrm{cc}}(f) - \widehat{R}_{\mathrm{cc}}(f)] \leq 2\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{cc}})$. By further taking into account the other side $\sup_{f\in\mathcal{F}} \widehat{R}_{\mathrm{cc}}(f) - R_{\mathrm{cc}}(f)$, we have for any $\delta > 0$, with probability at least $1-\delta$,

$$\sup_{f\in\mathcal{F}} \left|R_{\mathrm{cc}}(f) - \widehat{R}_{\mathrm{cc}}(f)\right| \leq 2\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{cc}}) + M\sqrt{\frac{\log\frac{2}{\delta}}{2n}},$$

which concludes the proof. ∎

Suppose the loss function $\mathcal{L}(q(\boldsymbol{x}), \widetilde{y})$ is ρ'-Lipschitz with respect to $f(\boldsymbol{x})$ $(0 \leq \rho \leq \infty)$ for all $\widetilde{y} \in \widetilde{\mathcal{Y}}$, by the Rademacher vector contraction inequality [147], we can obtain $\widetilde{\mathfrak{R}}_n(\mathcal{G}_{\mathrm{cc}}) \leq \sqrt{2}\rho' \sum_{y=1}^{k} \mathfrak{R}_n(\mathcal{H}_y)$. By further taking into account Lemma B.7 and Lemma B.4, for any $\delta > 0$, with probability $1-\delta$,

$$R_{\mathrm{cc}}(\widehat{f}_{\mathrm{cc}}) - R_{\mathrm{cc}}(f^\star) = R_{\mathrm{cc}}(\widehat{f}_{\mathrm{cc}}) - R_{\mathrm{cc}}(f_{\mathrm{cc}}) \leq 4\sqrt{2}\rho' \sum_{y=1}^{k} \mathfrak{R}_n(\mathcal{H}_y) + 2M\sqrt{\frac{\log\frac{2}{\delta}}{2n}},$$

which concludes the proof of Theorem 6. ∎

References

[1] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[5] Z. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2018.

[6] O. Chapelle, B. Scholkopf, and A. Zien, *Semi-Supervised Learning*. MIT Press, 2006.

[7] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 3, no. 1, pp. 1–130, 2009.

[8] G. Niu, W. Jitkrittum, B. Dai, H. Hachiya, and M. Sugiyama, "Squared-loss mutual information regularization: A novel information-theoretic approach to semi-supervised learning," in *Proceedings of the International Conference on Machine Learning*, 2013, pp. 10–18.

[9] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Recognition and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[10] Y.-F. Li and D.-M. Liang, "Safe semi-supervised learning: a brief introduction," *Frontiers of Computer Science*, vol. 13, no. 4, pp. 669–676, 2019.

[11] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in Neural Information Processing Systems*, 2017, pp. 1195–1204.

[12] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 5049–5059.

[13] C. Elkan and K. Noto, "Learning classifiers from only positive and unlabeled data," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2008, pp. 213–220.

[14] M. C. du Plessis, G. Niu, and M. Sugiyama, "Analysis of learning from positive and unlabeled data," in *Advances in Neural Information Processing Systems*, 2014, pp. 703–711.

[15] ——, "Convex formulation for learning from positive and unlabeled data," in *Proceedings of the International Conference on Machine Learning*, 2015, pp. 1386–1394.

[16] R. Kiryo, G. Niu, M. C. du Plessis, and M. Sugiyama, "Positive-unlabeled learning with non-negative risk estimator." in *Advances in Neural Information Processing Systems*, 2017, pp. 1674–1684.

[17] T. Sakai, M. C. du Plessis, G. Niu, and M. Sugiyama, "Semi-supervised classification based on classification from positive and unlabeled data," in *Proceedings of the International Conference on Machine Learning*, 2017, pp. 2998–3006.

[18] T. Sakai, G. Niu, and M. Sugiyama, "Semi-supervised auc optimization based on positive-unlabeled learning," *Machine Learning*, vol. 107, no. 4, pp. 767–794, 2018.

[19] C. Zhang, D. Ren, T. Liu, J. Yang, and C. Gong, "Positive and unlabeled learning with label disambiguation." in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2019, pp. 4250–4256.

[20] C. Gong, H. Shi, T. Liu, C. Zhang, J. Yang, and D. Tao, "Loss decomposition and centroid estimation for positive and unlabeled learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[21] C. Gong, T. Liu, J. Yang, and D. Tao, "Large-margin label-calibrated support vector machines for positive and unlabeled learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3471–3483, 2019.

[22] A. Menon, B. Van Rooyen, C. S. Ong, and B. Williamson, "Learning from corrupted binary labels via class-probability estimation," in *Proceedings of the International Conference on Machine Learning*, 2015, pp. 125–134.

[23] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1944–1952.

[24] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. Erfani, S. Xia, S. Wijewickrema, and J. Bailey, "Dimensionality-driven learning with noisy labels," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 3355–3364.

[25] B. Han, J. Yao, G. Niu, M. Zhou, I. Tsang, Y. Zhang, and M. Sugiyama, "Masking: A new perspective of noisy supervision," in *Advances in Neural Information Processing Systems*, 2018, pp. 5836–5846.

[26] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Advances in Neural Information Processing Systems*, 2018, pp. 8527–8537.

[27] X. Xia, T. Liu, N. Wang, B. Han, C. Gong, G. Niu, and M. Sugiyama, "Are anchor points really indispensable in label-noise learning?" in *Advances in Neural Information Processing Systems*, 2019, pp. 6835–6846.

[28] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," in *Proceedings of the International Conference on Computer Vision*, 2019, pp. 322–330.

[29] H. Wei, L. Feng, X. Chen, and B. An, "Combating noisy labels by agreement: A joint training method with co-regularization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020.

[30] T. Ishida, G. Niu, and M. Sugiyama, "Binary classification for positive-confidence data." in *Advances in Neural Information Processing Systems*, 2018, pp. 5917–5928.

[31] K. Shinoda, H. Kaji, and M. Sugiyama, "Binary classification from positive data with skewed confidence," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2020, pp. 3328–3334.

[32] H. Bao, G. Niu, and M. Sugiyama, "Classification from pairwise similarity and unlabeled data." in *Proceedings of the International Conference on Machine Learning*, 2018, pp. 452–461.

[33] N. Lu, G. Niu, A. K. Menon, and M. Sugiyama, "On the minimal supervision for training any binary classifier from only unlabeled data," in *Proceedings of the International Conference on Learning Representations*, 2019.

[34] N. Lu, T. Zhang, G. Niu, and M. Sugiyama, "Mitigating overfitting in supervised classification from two unlabeled datasets: A consistent risk correction approach," in *AISTATS*, 2020.

[35] T. Ishida, G. Niu, W. Hu, and M. Sugiyama, "Learning from complementary labels," in *Advances in Neural Information Processing Systems*, 2017, pp. 5644–5654.

[36] T. Ishida, G. Niu, A. K. Menon, and M. Sugiyama, "Complementary-label learning for arbitrary losses and models." in *Proceedings of the International Conference on Machine Learning*, 2019, pp. 2971–2980.

[37] L. Feng, T. Kaneko, B. Han, G. Niu, B. An, and M. Sugiyama, "Learning with multiple complementary labels," in *Proceedings of the International Conference on Machine Learning*, 2020, pp. 3072–3081.

[38] T. Cour, B. Sapp, and B. Taskar, "Learning from partial labels," *Journal of Machine Learning Research*, vol. 12, no. 5, pp. 1501–1536, 2011.

[39] Z.-N. Zeng, S.-J. Xiao, K. Jia, T.-H. Chan, S.-H. Gao, D. Xu, and Y. Ma, "Learning by associating ambiguously labeled images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 708–715.

[40] J. Luo and F. Orabona, "Learning from candidate labeling sets," in *Advances in Neural Information Processing Systems*, 2010, pp. 1504–1512.

[41] L.-P. Liu and T. G. Dietterich, "A conditional multinomial mixture model for superset label learning," in *Advances in Neural Information Processing Systems*, 2012, pp. 548–556.

[42] M. Rezaei, H. Yang, and C. Meinel, "Recurrent generative adversarial network for learning imbalanced medical image semantic segmentation," *Multimedia Tools and Applications*, pp. 1–20, 2019.

[43] Y.-F. Li, L.-Z. Guo, and Z.-H. Zhou, "Towards safe weakly supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[44] L.-Z. Guo, Z.-Y. Zhang, Y. Jiang, Y.-F. Li, and Z.-H. Zhou, "Safe deep semi-supervised learning for unseen-class unlabeled data," in *Proceedings of the 37th International Conference on Machine Learning*, 2020, pp. 3897–3906.

[45] L.-Z. Guo, Z. Zhou, and Y.-F. Li, "Record: Resource constrained semi-supervised learning under distribution shift," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020, pp. 1636–1644.

[46] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of Statistical Planning and Inference*, vol. 90, no. 2, pp. 227–244, 2000.

[47] M. Sugiyama, M. Krauledat, and K.-R. MÃžller, "Covariate shift adaptation by importance weighted cross validation," *Journal of Machine Learning Research*, vol. 8, no. May, pp. 985–1005, 2007.

[48] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate shift by kernel mean matching," *Dataset Shift in Machine Learning*, vol. 3, no. 4, p. 5, 2009.

[49] M. Sugiyama and M. Kawanabe, *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.

[50] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Advances in Neural Information Processing Systems*, 2007, pp. 137–144.

[51] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," *arXiv preprint arXiv:0902.3430*, 2009.

[52] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine Learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[53] C. Zhang, L. Zhang, and J. Ye, "Generalization bounds for domain adaptation," in *Advances in Neural Information Processing Systems*, 2012, pp. 3320–3328.

[54] I. Redko, A. Habrard, and M. Sebban, "Theoretical analysis of domain adaptation with optimal transport," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2017, pp. 737–753.

[55] Y. Zhang, T. Liu, M. Long, and M. I. Jordan, "Bridging theory and algorithm for domain adaptation," *arXiv preprint arXiv:1904.05801*, 2019.

[56] S. Kuroki, N. Charoenphakdee, H. Bao, J. Honda, I. Sato, and M. Sugiyama, "Unsupervised domain adaptation based on source-guided discrepancy," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 4122–4129.

[57] C. Huang, Y. Li, C. C. Loy, and X. Tang, "Learning deep representation for imbalanced classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5375–5384.

[58] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range loss for deep face recognition with long-tailed training data," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5409–5418.

[59] Y.-X. Wang, D. Ramanan, and M. Hebert, "Learning to model the tail," in *Advances in Neural Information Processing Systems*, 2017, pp. 7029–7039.

[60] Y. Cui, Y. Song, C. Sun, A. Howard, and S. Belongie, "Large scale fine-grained categorization and domain-specific transfer learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4109–4118.

[61] T. Zhang, "Statistical analysis of some multi-category large margin classification methods," *Journal of Machine Learning Research*, vol. 5, no. 10, pp. 1225–1251, 2004.

[62] X. Yu, T. Liu, M. Gong, and D. Tao, "Learning with biased complementary labels," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 68–83.

[63] L. Liu and T. Dietterich, "Learnability of the superset label learning problem," in *Proceedings of the International Conference on Machine Learning*, 2014, pp. 1629–1637.

[64] E. Hüllermeier and W. Cheng, "Superset learning based on generalized loss minimization," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2015, pp. 260–275.

[65] C. Gong, T. Liu, Y. Tang, J. Yang, J. Yang, and D. Tao, "A regularization approach for instance-based superset label learning," *IEEE Transactions on Cybernetics*, vol. 48, no. 3, pp. 967–978, 2018.

[66] E. Hüllermeier and J. Beringer, "Learning from ambiguously labeled examples," *Intelligent Data Analysis*, vol. 10, no. 5, pp. 419–439, 2006.

[67] Y.-C. Chen, V. M. Patel, R. Chellappa, and P. J. Phillips, "Ambiguously labeled learning using dictionaries," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2076–2088, 2014.

[68] C.-H. Chen, V. M. Patel, and R. Chellappa, "Learning from ambiguously labeled face images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[69] Y. Yao, C. Gong, J. Deng, X. Chen, J. Wu, and J. Yang, "Deep discriminative cnn with temporal ensembling for ambiguously-labeled image classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, pp. 12 669–12 676.

[70] M.-L. Zhang, F. Yu, and C.-Z. Tang, "Disambiguation-free partial label learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 10, pp. 2155–2167, 2017.

[71] R. Jin and Z. Ghahramani, "Learning with multiple labels," in *Advances in Neural Information Processing Systems*, 2003, pp. 921–928.

[72] M.-L. Zhang and F. Yu, "Solving the partial label learning problem: An instance-based approach." in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2015, pp. 4048–4054.

[73] C.-Z. Tang and M.-L. Zhang, "Confidence-rated discriminative partial label learning." in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, pp. 2611–2617.

[74] L. Feng and B. An, "Leveraging latent label distributions for partial label learning." in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018, pp. 2107–2113.

[75] ——, "Partial label learning with self-guided retraining," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, pp. 3542–3549.

[76] ——, "Partial label learning by semantic difference maximization," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2019, pp. 2294–2300.

[77] N. Nguyen and R. Caruana, "Classification with partial labels," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp. 551–559.

[78] F. Yu and M.-L. Zhang, "Maximum margin partial label learning," in *Proceedings of Asian Conference on Machine Learning*, 2016, pp. 96–111.

[79] ——, "Maximum margin partial label learning," *Machine Learning*, vol. 106, no. 4, pp. 573–593, 2017.

[80] J. Chai, I. W. Tsang, and W. Chen, "Large margin partial label machine," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 2594–2608, 2019.

[81] G. Lyu, S. Feng, T. Wang, and C. Lang, "A self-paced regularization framework for partial-label learning," *IEEE Transactions on Cybernetics*, 2020.

[82] T. Cour, B. Sapp, C. Jordan, and B. Taskar, "Learning from ambiguously labeled images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 919–926.

[83] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems*, 2004, pp. 321–328.

[84] T. Kaneko, I. Sato, and M. Sugiyama, "Online multiclass classification based on prediction margin for partial feedback," *arXiv preprint arXiv:1902.01056*, 2019.

[85] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, no. 11, pp. 463–482, 2002.

[86] N. Golowich, A. Rakhlin, and O. Shamir, "Size-independent sample complexity of neural networks," *arXiv preprint arXiv:1712.06541*, 2017.

[87] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems*, 2018, pp. 8778–8788.

[88] A. K. Menon, A. S. Rawat, S. J. Reddi, and S. Kumar, "Can gradient clipping mitigate label noise?" in *Proceedings of the International Conference on Learning Representations*, 2020.

[89] A. Ghosh, H. Kumar, and P. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[90] X. Wang, E. Kodirov, Y. Hua, and N. M. Robertson, "Improving mae against cce under label noise," *arXiv preprint arXiv:1903.12141*, 2019.

[91] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[92] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha, "Deep learning for classical japanese literature," *arXiv preprint arXiv:1812.01718*, 2018.

[93] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[94] K. Lang, "Newsweeder: Learning to filter netnews," in *Proceedings of the International Conference on Machine Learning*, 1995.

[95] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[96] C. L. Blake and C. J. Merz, "Uci repository of machine learning databases," 1998. [Online]. Available: http://archive.ics.uci.edu/ml/index.php

[97] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Review*, vol. 53, no. 2, pp. 217–288, 2011.

[98] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[99] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[100] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8026–8037.

[101] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations*, 2015.

[102] M.-L. Zhang, B.-B. Zhou, and X.-Y. Liu, "Partial label learning via feature-aware disambiguation," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1335–1344.

[103] F. Yu and M.-L. Zhang, "Maximum margin partial label learning," in *Proceedings of the Asian Conference on Machine Learning*, 2016, pp. 96–111.

[104] W. Liu and S.-F. Chang, "Robust multi-class transductive learning with graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 381–388.

[105] F. Nie, D. Xu, I. W.-H. Tsang, and C. Zhang, "Flexible manifold embedding: A framework for semi-supervised and unsupervised dimension reduction," *IEEE Transactions on Image Processing*, vol. 19, no. 7, pp. 1921–1932, 2010.

[106] J. Gorski, F. Pfeuffer, and K. Klamroth, "Biconvex sets and optimization with biconvex functions: A survey and extensions," *Mathematical Methods of Operations Research*, vol. 66, no. 3, pp. 373–407, 2007.

[107] M. Guillaumin, J. Verbeek, and C. Schmid, "Multiple instance metric learning from automatically labeled bags of faces," *Lecture Notes in Computer Science*, vol. 63, no. 11, pp. 634–647, 2010.

[108] G. Panis and A. Lanitis, "An overview of research activities in facial age estimation using the fg-net aging database," in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 737–750.

[109] B. Schölkopf, A. J. Smola *et al.*, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2002.

[110] A. L. Yuille and A. Rangarajan, "The concave-convex procedure," *Neural Computation*, vol. 15, no. 4, pp. 915–936, 2003.

[111] X. Wu and M.-L. Zhang, "Towards enabling binary decomposition for partial label learning," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2018, pp. 2427–2436.

[112] J. Wang and M.-L. Zhang, "Towards mitigating the class-imbalance problem for partial label learning," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2018, pp. 2427–2436.

[113] G. Lyu, S. Feng, T. Wang, C. Lang, and Y. Li, "Gm-pll: Graph matching based partial label learning," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[114] L. Feng, J. Lv, B. Han, M. Xu, G. Niu, X. Geng, B. An, and M. Sugiyama, "Provably consistent partial-label learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[115] J. Lv, M. Xu, L. Feng, G. Niu, X. Geng, and M. Sugiyama, "Progressive identification of true labels for partial-label learning," in *Proceedings of the International Conference on Machine Learning*, 2020.

[116] F. Briggs, X. Z. Fern, and R. Raich, "Rank-loss support instance machines for miml instance annotation," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 534–542.

[117] S. Shalev-Shwartz *et al.*, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2011.

[118] P. Auer, N. Cesa-Bianchi, and C. Gentile, "Adaptive and self-confident on-line learning algorithms," *Journal of Computer and System Sciences*, vol. 64, no. 1, pp. 48–75, 2002.

[119] Y. Li and P. M. Long, "The relaxed online maximum margin algorithm," in *Advances in Neural Information Processing Systems*, 2000, pp. 498–504.

[120] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the 20th International Conference on Machine Learning*, 2003, pp. 928–936.

[121] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online learning: A comprehensive survey," *arXiv preprint arXiv:1802.02871*, 2018.

[122] O. Dekel, N. Haghtalab, P. Jaillet *et al.*, "Online learning with a hint," in *Advances in Neural Information Processing Systems*, 2017, pp. 5299–5308.

[123] C. Zhang, C. Gong, T. Liu, X. Lu, W. Wang, and J. Yang, "Online positive and unlabeled learning," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 2020, pp. 2248–2254.

[124] R. Bhattacharjee and N. Manwani, "Online algorithms for multiclass classification using partial labels," in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2020, pp. 249–260.

[125] L. Jian, F. Gao, P. Ren, Y. Song, and S. Luo, "A noise-resilient online learning algorithm for scene classification," *Remote Sensing*, vol. 10, no. 11, p. 1836, 2018.

[126] M. Li, J. T. Kwok, and B.-L. Lu, "Online multiple instance learning with no regret," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1395–1401.

[127] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proceedings of the 24th International Conference on Machine Learning*, 2007, pp. 209–216.

[128] B. Kulis *et al.*, "Metric learning: A survey," *Foundations and Trends in Machine Learning*, vol. 5, no. 4, pp. 287–364, 2012.

[129] A. Globerson and S. T. Roweis, "Metric learning by collapsing classes," in *Advances in Neural Information Processing Systems*, 2006, pp. 451–458.

[130] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin, "Deep metric learning with angular loss," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2593–2601.

[131] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4004–4012.

[132] D. Wang and X. Tan, "Robust distance metric learning in the presence of label noise," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[133] M. Huai, C. Miao, Y. Li, Q. Suo, L. Su, and A. Zhang, "Metric learning from probabilistic labels," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2018, pp. 1541–1550.

[134] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proceedings of the International Conference on Machine Learning*, 2015, pp. 1180–1189.

[135] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Universal domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2720–2729.

[136] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.

[137] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2010.

[138] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2018, pp. 1640–1650.

[139] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[140] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3723–3732.

[141] Y. Shu, Z. Cao, M. Long, and J. Wang, "Transferable curriculum for weakly-supervised domain adaptation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 4951–4958.

[142] F. Liu, J. Lu, B. Han, G. Niu, G. Zhang, and M. Sugiyama, "Butterfly: A panacea for all difficulties in wildly unsupervised domain adaptation," *arXiv preprint arXiv:1905.07720*, 2019.

[143] Y. Zhang, F. Liu, Z. Fang, B. Yuan, G. Zhang, and J. Lu, "Clarinet: A one-step approach towards budget-friendly unsupervised domain adaptation," *arXiv preprint arXiv:2007.14612*, 2020.

[144] ——, "Learning from a complementary-label source domain: Theory and algorithms," *arXiv preprint arXiv:2008.01454*, 2020.

[145] C. McDiarmid, "On the method of bounded differences," in *Surveys in Combinatorics*, 1989.

[146] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. MIT Press, 2012.

[147] A. Maurer, "A vector-contraction inequality for rademacher complexities," in *Proceedings of the International Conference on Algorithmic Learning Theory*, 2016, pp. 3–17.