Towards Realistic Transfer Learning Methods: Theory and Algorithms

A thesis submitted in fulfilment of the requirements
for the degree of

Doctor of Philosophy
in
Computer Science

by

Feng Liu

to

School of Computer Science
Faculty of Engineering and Information Technology
Australian Artificial Intelligence Institute
University of Technology Sydney
NSW - 2007, Australia
2020

Certificate of Original Authorship

I, *Feng Liu*, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy in the School of Computer Science at the Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

SIGNATURE: Production Note: Signature removed prior to publication.

DATE: 08, Sep., 2020

Abstract

Transfer learning aims to leverage knowledge from domains with abundant labels (i.e., source domains) to help train a classifier or predictor for the domain with insufficient labels (i.e., target domain). The trained classifier or predictor is expected to have better performance (e.g., higher accuracy) than classifiers only trained with data in the target domain.

Although recent research of transfer learning has shown a decent ability to transfer knowledge from a source domain to a target domain, most research require certain assumptions to ensure their efficacy. These assumptions are probably not realistic, which means that existing transfer learning methods still face several unsolved and challenging problems in real world.

This thesis aims to address four orthogonal problems faced by existing transfer learning methods: 1) How to test if feature spaces of two domains are from different distributions; 2) How to transfer knowledge when labels in the source domain cannot be perfectly annotated (i.e., the source domain contains noisy labels); 3) How to transfer knowledge when source and target domains have different dimensions (i.e., heterogeneous scenario); and 4) How to transfer knowledge across multiple source domains and a different-dimension target domain.

To address Problem 1), this thesis presents two new two-sample tests to test

if the feature spaces of source domains and target domain are from different distributions. One is suitable for low-dimension data (Chapter 3) and another for high-dimension data (Chapter 4). If feature spaces of domains are statistically different, we need to use transfer learning methods on these domains. Moreover, the test statistics used in the proposed tests can be used to measure the distributional discrepancy between two domains.

To address Problem 2), this thesis presents a theoretical bound to show that existing transfer learning methods cannot work well when a source domain contain noisy labels. Then, a novel transfer learning approach is proposed to transfer knowledge across a source domain (with noisy labels) and a target domain. Finally, a generalization bound is proved to explain why the proposed method can reliably transfer knowledge across domains in noisy scenario (Chapter 5).

To address Problem 3), the most challenging problem in the field of domain adaptation, Chapter 6 presents a theorem to show when we can reliably transfer knowledge across two different-dimension (i.e., heterogeneous) domains and propose a solution to this problem. Since methods in Chapter 6 assume that the number of samples in two domains must be the same (i.e., two balanced domains), Chapter 7 presents a novel fuzzy-relation based method to transfer knowledge across two imbalanced domains.

To address Problem 4), Chapter 8 presents a novel fuzzy-relation neural network to transfer knowledge from multiple source domains to a target domain, where any of two domains are heterogeneous (i.e., feature spaces of any of two domains have different dimensions).

To conclude, this thesis not only propose a set of effective methods for realistic transfer learning, but also contribute to theory of transfer learning.

Dedication

To my loving wife, parents and families...

ACKNOWLEDGMENTS

It is a memorial and exciting journey at University of Technology Sydney (UTS) for pursuing my Ph.D. degree in the past three and half years. I am sincerely grateful to the people who inspired and helped me in many ways.

I would like to express my foremost and deepest gratitude to my principal supervisor, A./Professor Guangquan Zhang. Without his patience and encouragement, I would not have been able to complete this Ph.D. program. He taught me step by step how to become a qualified researcher from its beginning. He always led me to the right research direction with his expert knowledge of theory and abundant research experience. He placed considerable trust in my research ability and unconditionally support me in pursuing my own research interests. His wisdom and immense knowledge always enlightened me to go further and deeper in my research. Without his critical comments, I would waste my time on trivial research ideas. Discussion with him greatly improves the scientific aspect and quality of my research. He helped me to build my confidence in my research outcomes and to be hopeful when faced with any difficulty, from academic to living.

Meanwhile, I am greatly indebted to my co-advisor, Distinguished Professor

Jie Lu. Her decisiveness and sharp insights continuously motivated me when I got lost or afraid about the future. Her confidence and enthusiasm inspired me to do the right thing even when the road got tough. I felt extremely honored to be guided by such a rigorous researcher as well as an enthusiastic mentor. What she taught me and what I learned from her in the past four years has benefited my Ph.D. study and will be a great treasure throughout my life.

During my Ph.D. period, I am very fortunate to join the Imperfect Information Learning Team as a research intern at RIKEN Center for Advanced Intelligence Project (RIKEN-AIP), working with Prof. Masashi Sugiyama, Dr. Gang Niu and Dr. Bo Han; to join the Gatsby Computational Neuroscience Unit at UCL as a visiting scholar, working with Prof. Arthur Gretton, Dr. Dougal J. Sutherland and Wenkai Xu. I would like to express my thankfulness to Prof. Masashi Sugiyama and Prof. Arthur Gretton that they led me to the machine learning field. Discussion and cooperation with them helped me to deeply understand what the top-tier machine-learning work should be and what characteristics a machine-learning researcher should have. I am impressed and inspired by their persistence, rigour and passion on research.

I would like to express my thankfulness to every member of the Decision Systems & e-Service Intelligence Lab (DeSI) in the Centre for Artificial Intelligence (CAI). It was a wonderful experience to spend four years with these dedicated researchers. I especially thank Dr. Hua Zuo, Dr. Junyun Xuan, Dr. Zheng Yan, Dr. Yi Zhang, Dr. Anjin Liu, Dr. Fujin Zhu, Dr. Feng Gu, Dr. Ning Lu and Zhen Fang who provided insightful comments related to my research problem during my Ph.D. candidature; Dr. Ximeng Wang, Dr. Guanjin Wang, Qian Liu, Dr. Chenlian Hu, Bin Zhang and Bin Wang who have shared their opinions and comments

with me, Dr. Anjin Liu, Dr. Junyun Xuan, Hang Yu and Ruiping Yin who shared my joys and sadness.

I must thank Dr. Gang Niu and Dr. Bo Han for the valuable suggestions for my Ph.D. study. Their persistence, rigour and passion on research impressed and inspired me. I have learned much about how to do an excellent machine-learning research from them. I also express my thankfulness to Dr. Dougal J. Sutherland who patiently guides me how to do an excellent machine-learning research. I am also impressed and inspired by his persistence, rigour and passion on research.

Meanwhile, I genuinely thank Jemima Moore, Sue Felix and Michele Mooney for polishing the language of my publications. They are always patient to all my emails of questioning revised sentences. I thank all my wonderful friends, classmates and colleagues for every enjoyable moment.

Last, I would like to express my heartfelt appreciation and gratitude to my wife, parents, and families for their love and support.

List of Publications

1. **Feng Liu***, Wenkai Xu*, Jie Lu, Guangquan Zhang, Arthur Gretton, Dougal Sutherland. Learning Deep Kernels for Nonparametric Two Sample Test. *Proceedings of the 37th International Conference on Machine Learning* (ICML 2020), online, 13-18 July, 2020. (*Equal Contribution). [ERA: A, CORE: **A***]

2. **Feng Liu**, Guangquan Zhang, Jie Lu, Heterogeneous Domain Adaptation: An Unsupervised Approach, *IEEE Transactions on Neural Networks and Learning Systems* (IEEE-TNNLS), 2020. DOI: 10.1109/TNNLS.2020.2973293. [ERA&CORE: **A***, JCR Q1]

3. **Feng Liu**, Guangquan Zhang, Jie Lu, Multi-source heterogeneous unsupervised domain adaptation via shared-fuzzy-equivalence-relation neural networks, *IEEE Transactions on Fuzzy Systems* (IEEE-TFS), 2020. DOI: 10.1109/TFUZZ.2020.3018191. [ERA&CORE: **A***, JCR Q1]

4. **Feng Liu**, Guangquan Zhang, Jie Lu, Unsupervised Heterogeneous Domain Adaptation via Shared Fuzzy Equivalence Relations, *IEEE Transactions on Fuzzy Systems* (IEEE-TFS), Vol. 26, no. 6, pp. 3555-3568, 2018. [ERA&CORE: **A***, JCR Q1]

5. **Feng Liu**, Jie Lu, Bo Han, Gang Niu, Guangquan Zhang, Masashi Sugiyama. Butterfly: One-step Approach towards Wildly Unsupervised Domain Adaptation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (IEEE-TPAMI). [ERA&CORE: **A***, JCR Q1] (Revise&Resubmit)

6. **Feng Liu**, Jie Lu, Bo Han, Gang Niu, Guangquan Zhang, Masashi Sugiyama. Butterfly: A Panacea for All Difficulties in Wildly Unsupervised Domain Adaptation, *NeurIPS 2019 Workshop on Learning Transferable Skills*, pp. 1-8, Vancouver, Canada, 8-14 December, 2019. [ERA: A, CORE: **A***]

7. **Feng Liu**, Guangquan Zhang, Jie Lu. A Novel Fuzzy Neural Network for Unsupervised Domain Adaptation in Heterogeneous Scenarios, *Proceedings of the 2019 IEEE International Conference on Fuzzy Systems* (FUZZ-IEEE 2019), pp. 1-6, New Orleans, USA, 2019. [ERA&CORE: A]**[Best Student Paper Award]**

8. Yiyang Zhang*, **Feng Liu***, Zhen Fang, Bo Yuan, Guangquan Zhang, Jie Lu, Clarinet: A One-step Approach Towards Budget-friendly Unsupervised Domain Adaptation, *Proceedings of the 29th International Joint Conference on Artificial Intelligence* (IJCAI 2020), 2020. (*Equal Contribution). [ERA: A, CORE: **A***]

9. **Feng Liu**, Guangquan Zhang, Jie Lu, A Novel Non-parametric Two-Sample Test on Imprecise Observations, *Proceedings of the 2020 IEEE International Conference on Fuzzy Systems* (FUZZ-IEEE 2020), online, 19-24 July, 2020. [ERA&CORE: A]

10. **Feng Liu**, Guangquan Zhang, Jie Lu. Unconstrained fuzzy feature fusion for heterogeneous unsupervised domain adaptation, *Proceedings of the 2018 IEEE International Conference on Fuzzy Systems* (FUZZ-IEEE 2018), pp. 1-8, Rio de Janeiro, Brazil, 8-13 July, 2018. [ERA&CORE: A]

11. **Feng Liu**, Guangquan Zhang, Jie Lu. Heterogeneous unsupervised domain adaptation based on fuzzy feature fusion, *Proceedings of the 2017 IEEE International Conference on Fuzzy Systems* (FUZZ-IEEE 2017), pp. 1-6, Naples, Italy, 9-12 July, 2017. [ERA&CORE: A]

12. **Feng Liu**, Guangquan Zhang, Anjin Liu, Jie Lu, Discrepancy of Diverse Subsets for Distribution Comparison, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (IEEE-TPAMI). [ERA&CORE: **A***, JCR Q1] (under review)

13. **Feng Liu**, Zhen Fang, Guangquan Zhang, Jie Lu, Take A Closer Look at Kernel Nonparametric Two-sample Tests via the Lebesgue-Besicovitch Differentiation Theorem, *IEEE Transactions on Neural Networks and Learning Systems* (IEEE-TNNLS). [ERA&CORE: **A***, JCR Q1] (submitted)

14. **Feng Liu**, Guangquan Zhang, Jie Lu, A Knowledge-Ensemble Model for Heterogeneous Unsupervised Domain Adaptation, *IEEE Transactions on Knowledge and Data Engineering* (IEEE-TKDE). [ERA: A, CORE: **A***, JCR Q1] (submitted)

15. Yiyang Zhang*, **Feng Liu***, Zhen Fang, Bo Yuan, Guangquan Zhang, Jie Lu, Learning from a Complementary-label Source Domain: Theory and Algorithms, *IEEE Transactions on Neural Networks and Learning Systems*

(IEEE-TNNLS). (*Equal Contribution). [ERA&CORE: **A***, JCR Q1] (under review)

16. Hua Zuo, Jie Lu, Guangquan Zhang, **Feng Liu**, Fuzzy Transfer Learning Using an Infinite Gaussian Mixture Model and Active Learning, *IEEE Transactions on Fuzzy Systems* (IEEE-TFS), Vol. 27, no. 2, pp. 291 - 303, 2019. [ERA&CORE: **A***, JCR Q1].

17. Zhen Fang, Jie Lu, **Feng Liu**, Junyu Xuan, Guangquan Zhang, Open Set Domain Adaptation: Theoretical Bound and Algorithm, *IEEE Transactions on Neural Networks and Learning Systems* (IEEE-TNNLS), 2020. DOI: 10.1109/TNNLS.2020.3017213. [ERA&CORE: **A***, JCR Q1]

18. Zhen Fang, Jie Lu, **Feng Liu**, Guangquan Zhang. Unsupervised Domain Adaptation with Sphere Retracting Transformation, *Proceedings of the 2019 IEEE International Joint Conference on Neural Networks* (IJCNN 2019), pp. 1-8, Budapest, Hungary, 14-19 July, 2019. [ERA&CORE: A]

19. Anjin Liu, Jie Lu, **Feng Liu**, Guangquan Zhang, Accumulating regional density dissimilarity for concept drift detection in data streams, *Pattern Recognition* (PR), Vol. 76, pp. 256-272, 2018. [ERA&CORE: **A***, JCR Q1]

20. Yi Zhang, Jie Lu, **Feng Liu**, et. al., Does deep learning help topic extraction? A kernel k-means clustering method with word embedding, *Journal of Informetrics*, Vol. 12, no. 4, pp. 1099-1117, 2018. [ERA&CORE: A, JCR Q1]

21. Li Zhong*, Zhen Fang*, **Feng Liu**, Yuan Bo, Guangquan Zhang, Jie Lu, Open Set Domain Adaptation: Theoretical Bound and Algorithm, *IEEE*

Transactions on Neural Networks and Learning Systems (IEEE-TNNLS). (*Equal Contribution). [ERA&CORE: **A***, JCR Q1] (under review)

22. Qian Zhang, Dianshuang Wu, Jie Lu, **Feng Liu**, Guangquan Zhang, A cross-domain recommender system with consistent information transfer, *Decision Support Systems* (DSS), Vol. 104, pp. 49-63, 2017. [ERA: **A***, JCR Q1]

23. Fan Dong, Jie Lu, Yiliao Song, **Feng Liu**, Guangquan Zhang, A Concept Drift Region-based Data Sample Editing Methodology, *IEEE Transactions on Cybernetics* (IEEE-TCYB). [ERA&CORE: A, JCR Q1] (under review)

Table of Contents

List of Publications viii

List of Figures xxi

List of Tables xxviii

1 Introduction 1

1.1 Background 1

1.2 Motivation 3

1.3 Research Questions and Objectives 4

1.4 Research Innovation and Contributions 8

1.4.1 Research Innovation 9

1.4.2 Research Contributions 10

1.5 Research Significance 12

1.6 Thesis Structure 14

2 Literature Review 18

2.1 Transfer Learning 18

2.2 Two-sample Test 21

2.2.1 F-divergence based Non-parametric Two-sample Tests . . 22

2.2.2 Function based Non-parametric Two-sample Tests 23
2.2.3 Subset based Non-parametric Two-sample Tests 25
2.3 Domain Adaptation 27
2.3.1 Homogeneous Unsupervised Domain Adaptation 28
2.3.2 Heterogeneous Domain Adaptation 31

3 Discrepancy of Diverse Subsets: A Non-parametric Two-Sample Test for Low-Dimension Data 37

3.1 Introduction 37
3.2 Problem Setting 41
3.3 Discrepancy of Diverse Subsets 41
3.3.1 Population Form of DDS 41
3.3.2 Empirical DDS 45
3.3.3 Automatic Selection of Beta 47
3.3.4 Consistency of the empirical DDS 48
3.3.5 DDS as a Semimetric in P(X) 50
3.4 DDS for Two-sample Test 51
3.4.1 Asymptotic Distribution of DDS 51
3.4.2 Two Hypothesis Tests based on DDS 54
3.5 DDS for Unsupervised Domain Adaptation 55
3.5.1 Generalization Bound for DDS based Unsupervised Domain Adaption 56
3.5.2 DDS based Unsupervised Domain Adaption 60
3.6 Experiments 62
3.6.1 Evaluation of DDS for Two-sample Test 62

3.6.2 Evaluation of DDS for Unsupervised Domain Adaptation . 69
3.7 Summary 74

4 Maximum Mean Discrepancy with Learned Deep Kernels: A Non-parametric Two-Sample Test for High-Dimension Data 76
4.1 Introduction 76
4.2 Concepts and Notations 79
4.3 Limits of Simple Kernels 84
4.4 Relationship to Classifier-Based Tests 85
4.5 Learning Deep Kernels 88
4.6 Theoretical Analysis 89
4.7 Experimental Results 92
4.7.1 Comparison on Benchmark Datasets 92
4.7.2 Ablation Study 101
4.7.3 Interpretability on *CIFAR-10* vs *CIFAR-10.1* 102
4.8 Summary 104

5 Butterfly: A One-step Approach towards Wildly Unsupervised Domain Adaptation 107
5.1 Introduction 107
5.2 Wildly Unsupervised Domain Adaptation 111
5.2.1 Nature of WUDA 112
5.2.2 WUDA ruins HoUDA Methods 113
5.3 Butterfly: Towards Robust One-step Approach 114
5.3.1 Loss Function in Butterfly 115
5.3.2 Training Procedure of Butterfly 115

5.4 Butterfly vs. Two-step Approach 118
5.4.1 Two-step Approach (A Compromise Solution) 119
5.4.2 One-step Approach (Butterfly) 119
5.5 Theoretical Analysis 120
5.5.1 WUDA ruins HoUDA Methods 122
5.5.2 Two-step Approach is a Compromise Solution 123
5.5.3 Why does Butterfly Eliminate Noise Effect? 124
5.5.4 Principle-guided Butterfly 129
5.5.5 A Generalization Bound for WUDA 130
5.6 Comparison to Related Works 133
5.7 Experiments 134
5.7.1 Simulated WUDA Tasks 134
5.7.2 Real-world WUDA Tasks 136
5.7.3 Baselines 137
5.7.4 Network Structure and Optimizer 138
5.7.5 Experimental Setup 139
5.7.6 Results on Simulated WUDA Tasks 140
5.7.7 Results on Real-world WUDA Tasks 144
5.7.8 Ablation Study 144
5.8 Summary 148

6 Heterogeneous Domain Adaptation: An Unsupervised Approach and Its Theoretical Guarantees 149

6.1 Introduction 149
6.2 Problem Setting and Notations 154

6.3 Heterogeneous Unsupervised domain adaptation 155
6.3.1 Unsupervised Knowledge Transfer Theorem for HeUDA . 156
6.3.2 Principal Angle-based Measurement between Heterogeneous Feature Spaces 160
6.3.3 GLG: The Proposed HeUDA Method 163
6.3.4 Discussion of Definitions and Theorems 169
6.3.5 Limitation of GLG 171
6.4 Optimization of GLG 171
6.4.1 Microscopic Analysis of an Eigen Dynamic System 172
6.4.2 Gradients of Cost Function I 173
6.4.3 A Hybrid Optimization Method for GLG 175
6.5 Experiments 176
6.5.1 Datasets for HeUDA 176
6.5.2 Experimental Setup 180
6.5.3 Experiment I: RMG 184
6.5.4 Experiment II: Overall comparisons 187
6.6 Summary 191

7 Shared Fuzzy Equivalence Relations: A Heterogeneous Unsupervised Domain Adaptation Approach for Imbalanced Domains 192
7.1 Introduction 192
7.2 Concepts and Notations 195
7.2.1 Fuzzy Geometry 195
7.2.2 Fuzzy Equivalence Relation 197
7.3 Similarity between Fuzzy Vectors 200

7.3.1 A Metric on Fuzzy Geometry 200
7.3.2 Similarity between Fuzzy Vectors 204
7.4 F-HeUDA via Shared Fuzzy Equivalence Relations 205
7.4.1 Theoretical Guarantees 206
7.4.2 Shared Fuzzy Equivalence Relations (SFER) 211
7.4.3 Learning Process of SFER 212
7.4.4 F-HeUDA via SFER (F-HeUDA) 214
7.5 Experiments 216
7.5.1 Dataset Description and Parameters Setting 216
7.5.2 Prediction Performance 218
7.5.3 Convergence of Learning Algorithm 221
7.5.4 User-oriented Decision Making Pattern 222
7.6 Summary 224

8 Fuzzy-relation Neural Networks: A Multi-source Heterogeneous Unsupervised Domain Adaptation Approach 226

8.1 Introduction 226
8.2 Concepts and Problem Setting 230
8.2.1 Similarity between Fuzzy Vectors 230
8.2.2 Fuzzy Equivalence Relations and Partitioning of Fuzzy Sets 232
8.2.3 Multi-source Unsupervised Domain Adaptation 233
8.3 Shared Fuzzy Equivalence Relations for Multi-source Domains . 234
8.3.1 Theoretical Guarantees for Multi-source SFER 236
8.3.2 Multi-source SFER 239
8.3.3 Learning Process of Multi-source SFER 239

8.4 Shared Fuzzy Equivalence Relations Neural Network for Multi-source HeUDA 243
8.4.1 Structure of the Proposed Neural Network 243
8.4.2 Loss Function of the Proposed Neural Network 247
8.4.3 Learning Process of the Proposed Neural Network 248
8.5 Experiments 250
8.5.1 Datasets and Tasks 250
8.5.2 Experimental Setup 251
8.5.3 Experiment I: Classification Accuracy 253
8.5.4 Experiment II: Stability Analysis 255
8.5.5 Experiment III: Parameters Sensitivity 258
8.6 Summary 260

9 Conclusion and Future Study **263**
9.1 Conclusions 263
9.2 Future Study 267

A Appendix **270**
A.1 Appendix of Chapter 3 270
A.1.1 A Corollary of Radon-Nikodym Theorem 270
A.1.2 Proofs of Theorems 271
A.2 Appendix of Chapter 4 277
A.2.1 Preliminaries 277
A.2.2 Main Results 279
A.2.3 Uniform Convergence Results 281
A.2.4 Constructing Appropriate Kernels 282

A.2.5 Miscellaneous Results 287
A.3 Appendix of Chapter 5 288
A.3.1 Review of Generation of Noisy Labels 288
A.3.2 Proofs 290
A.4 Appendix of Chapter 6 301
A.4.1 Proof of Theorem 6.1 301
A.4.2 Proof of Lemma 6.1 302
A.4.3 Proof of Theorem 6.2 302
A.4.4 Proof of Theorem 6.3 302
A.4.5 Proof of Theorem 6.4 303
A.4.6 Proof of Lemma 6.2 304

Bibliography **306**

LIST OF FIGURES

FIGURE **Page**

1.1 Thesis structure 17

3.1 The proposed DDS based unsupervised domain adaptation (DSA). Red circles with the solid line represent the features of both domains, and blue circles with solid line represent in-sample (source domain) outputs of DSA, and blue circles with dash line represent out-sample (target domain) outputs of DSA. Other circles represent hidden neurons of DSA. In the third layer of DSA, representations of instances of both domains are obtained: $T^(X_s)$ and $T^(X_t)$. 62

4.1 Blob dataset (a), with contours of Gaussian kernel (b) and deep kernel (c) evaluated at 9 locations (contour values are $0.7, 0.8$ and 0.9). Each distribution has 9 modes; the central modes have the same shape, but $\mathbb{Q}$ has a different shape at each other mode. A Gaussian kernel (b) compares points isotropically throughout the space; contours show $k(x, \mu)$ for each mode μ. A deep kernel (c) learned by our methods compares points differently in different locations, allowing better identification of differences between $\mathbb{P}$ and $\mathbb{Q}$. 78

4.2 Results on *Blob-S* and *Blob-D* given $\alpha = 0.05$; see Section 4.7 for details. n_b is the number of samples at each mode, so $n_b = 100$ means drawing 900 samples from each of $\mathbb{P}$ and $\mathbb{Q}$. We report, when increasing n_b, (a) average test power, (b) standard deviation of test power, (c) the value of $\hat{J}_\lambda$, and (d) average type-I error. (a), (b) and (c) are on *Blob-D*, and (d) is on *Blob-S*. Shaded regions show standard errors for the mean, and the black line shows α. 85

4.3 Results on *HDGM-S* and *HDGM-D* for $\alpha = 0.05$ (black line). Left: average test power (a) and Type I error (b) when increasing the number of samples N, keeping $d = 10$. Right: average test power (c) and Type I error (d) when increasing the dimension d, keeping $N = 4\,000$. Shaded regions show standard errors for the mean. 92

4.4 The structure of ϕ_ω in MMD-D on *MNIST*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout. Best viewed zoomed in. 95

4.5 The structure of classifier F in C2ST-S and C2ST-L on *MNIST*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout. In the first layer, we will convert the *CIFAR* images from $32 \times 32 \times 3$ to $64 \times 64 \times 3$. Best viewed zoomed in. 95

4.6 The structure of ϕ_ω in MMD-D on *CIFAR*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout in all layers. In the first layer, we will convert the *CIFAR* images from $32 \times 32 \times 3$ to $64 \times 64 \times 3$. Best viewed zoomed in. 95

4.7 The structure of classifier F in C2ST-S and C2ST-L on *CIFAR*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout. Best viewed zoomed in. 95

4.8 The best test locations (learned by an ME test with $L = 1$) from 10 experiments on *CIFAR-10* vs *CIFAR-10.1*. Average rejection rate is 0.415. 104

4.9 The best test locations (learned by an ME test, $L = 1$, with a deep kernel optimized for an MMD test) from 10 experiments on *CIFAR-10* vs *CIFAR-10.1*. Average rejection rate is 0.637. 105

4.10 The best test locations (selected among existing images with our learned deep kernel, $L = 1$) from 10 experiments on *CIFAR-10* vs *CIFAR-10.1*. Average rejection rate is 0.653. 105

5.1 Wildly unsupervised domain adaptation (WUDA). The blue line denotes that HoUDA transfers knowledge from clean source data (P_s) to unlabeled target data (P_{x_t}). However, perfectly clean data is hard to acquire. This brings *wildly unsupervised domain adaptation* (WUDA), namely transferring knowledge from noisy source data ($\widetilde{P}_s$) to unlabeled target data (P_{x_t}). Note that label corruption process (black dash line) is unknown in practice. To handle WUDA, a compromise solution is a two-step approach (green line), which sequentially combines label-noise algorithms ($\widetilde{P}_s \to \hat{P}_s$, label correction) and existing HoUDA ($\hat{P}_s \to P_{x_t}$). This chapter proposes a robust one-step approach called Butterfly (red line, $\widetilde{P}_s \to P_{x_t}$ directly), which eliminates noise effects from $\tilde{P}_s$. 108

5.2 WUDA ruins representative HoUDA methods. Representative HoUDA methods includes *deep adaptation network* (DAN, a IPM based method [Long et al., 2015]), *domain-adversarial neural network* (DANN, a adversarial training based method [Ganin et al., 2016a]), *asymmetric tri-training domain adaptation* (ATDA, a pseudo-label based method [Saito et al., 2017]) and *transferable curriculum learning* (TCL, a robust HoUDA method [Shu et al., 2019]). B-Net is our proposed WUDA method. We report target-domain accuracy of all methods when the noise rate of source domain changes (a) from 5% to 70% (symmetry-flip noise) and (b) from 5% to 45% (pair-flip noise). Clearly, when the noise rate of source domain increases, target-domain accuracy of representative HoUDA methods drops quickly while that of B-Net keeps stable consistently. 109

5.3 Butterfly Framework. Two networks (F_1 and F_2) in Branch-I are jointly trained on noisy source data and pseudo-labeled target data (mixture domain). Two networks in Branch-II (F_{t1} and F_{t2}) are trained on pseudo-labeled target data. By using dual-checking principle, Butterfly checks high-correctness data out from both mixture and pseudo-labeled target data. After cross-propagating checked data, Butterfly can obtain high-quality *domain-invariant representations* (DIR) and *target-specific representations* (TSR) simultaneously in an iterative manner. Note that the interaction between DIR and TSR happens via the shared CNN. Besides, in the first training epoch, since we do not have any pseudo-labeled target data, we need to use noisy source data as the pseudo-labeled target data, which follows [Saito et al., 2017]. . 112

5.4 Visualization of *MNIST* and *SYND*. 136

5.5 Visualization of *Bing*, *Caltech256*, *ImageNet* and *SUN* (taking "horse" as the common class). 137

5.6 The architecture of B-Net for digit WUDA tasks *SYND* ↔ *MNIST*. We added BN layer in the last convolution layer in CNN and FC layers in F_1 and F_2. We also used dropout in the last convolution layer in CNN and FC layers in F_1, F_2, F_{t1} and F_{t2} (dropout probability is set to 0.5). 138

5.7 The architecture of B-Net for (a) human-sentiment WUDA tasks and (b) real-world WUDA tasks. We added BN layer in the first FC layers in F_1 and F_2. We also used dropout in the first FC layers in F_1, F_2, F_{t1} and F_{t2} (dropout probability is set to 0.5). 139

5.8 Target-domain accuracy vs. number of epochs on four *SYND*→*MNIST* WUDA tasks. 141

5.9 Target-domain accuracy vs. number of epochs on four *MNIST*→*SYND* WUDA tasks. 142

6.1 The progress of the GLG method. The original source and target domains come from the same underlying domain (e.g., classifying Latin sentences or analyzing human sentiment). However, the underlying domain is hard to observe and we can only observe its projection/representation on two (or more) domains, e.g., two heterogeneous domains in this figure. 153

7.1 Relationships among $\sup\{D_\lambda(u,v) : D_\lambda(u,v) \in \Omega(\lambda)\}$, $d(u,\overline{A}_j(\lambda))$, $d(v,\overline{A}_i(\lambda))$ and $d(A_i,A_j)$ 202

7.2 Traditional fuzzy equivalence relations v.s. SFER. In subfigure (a), two domains clearly cannot use the same α to obtain the same number of clusters. But, in SFER, two domains have a much bigger probability of using the same α to obtain the same number of clusters. 212

7.3 The performance of each method (mean accuracy and standard deviation) on 4 tasks. In each subfigure, the minimum mean accuracy is set as 0.4. 220

7.4 The convergence of Algorithm 7.1 on four tasks. Subfigures (a)-(d) illustrate the value of the cost function J_1 and subfigures (e)-(f) illustrate the r_i of R_{TD}^M of the source domain and the target domain. . . . 222

8.1 Three scenarios for the *unsupervised domain adaptation* (UDA) problem. Rectangles represent the labeled source data and triangle represents the unlabeled target data. Labeled source data may come from a) a single source domain (i.e., single-source scenario) or b) multiple source domains whose feature spaces have the same dimension (i.e., multi-homogeneous-source scenario) or c) multiple source domains whose feature spaces have different dimensions (i.e., multi-heterogeneous-source scenario). Given a target domain, since we probably have many different-dimension source domains (i.e., multi-heterogeneous-source scenario), the third scenario is more general than the other scenarios. 227

8.2 Target domain has 7 features while *source domain 1* has 6 and *source domain 2* has 5. Given the same α, traditional fuzzy equivalence relations can only simultaneously cluster 1) features in *source domain 1* as 3 categories, and 2) features in *source domain 2* as 2 categories, and 3) features in target domain as 5 categories. However, multi-source shared fuzzy equivalence relations can simultaneously cluster features in *source domain 1*, *source domain 2* and target domain as 2 categories. 235

8.3 Network struture of SFERNN. SFERNN is a five-layer neural network containing c source branches and one target branch ($c = 2$ in this figure). Network structure (i.e., N_l in this figure and how to connect two adjacent layers) of SFERNN is confirmed by MsSFER. Loss function of SFERNN is composed of two parts. The first part represents cross-entropy loss on labeled data from c source domains. The second part represents distributional discrepancy (MMD) between source domains and target domain. 244

8.4 Analysis of parameters' sensitivity on 3 tasks. Learning rate represents η in Algorithm 8.2 and Lambda represents λ_2 in loss function of SFERNN. 262

List of Tables

Table **Page**

3.1 Synthetic data set results (%). The null hypothesis H_0 is $p = q$. The $H_1^{\Delta_i}$ indicates the percentage of the test reject the null hypothesis, while the magnitude between p and q is Δ_i. Since Δ_1 is equal to 0, $H_1^{\Delta_1}$ is the Type I error (the lower the better), while $H_1^{\Delta_2}$ and $H_1^{\Delta_3}$ are the Type II error (the lower the better). Bold values represent the lowest error. 67

3.2 Average Type I error (%) and corresponding rankings. This table shows the average Type I error for each test and the average values of Type I errors obtained under m_1, m_2, m_3 (see Table 3.1). Bold values represent the lowest error. 68

3.3 Average Type II error (%) and corresponding rankings. This table shows the average Type II error for each test and the average values of Type II errors obtained under m_1, m_2, m_3 (see Table 3.1). Bold values represent the lowest error. 68

3.4 Higgs data set results (%) and corresponding rankings. Real-I has both two-sample sets drawn from P distribution. Real-II has bath two sample sets drawn from Q distribution. Real-III has one sample set drawn from P and the other drawn from Q. Bold values represent the lowest error and lowest running time. 69

3.5 Real-world datasets for transfer learning 70

3.6 Classification accuracy for DDS-based domain adaptation and MMD-based domain adaptation on 28 transfer recognition tasks (object recognition (the first 8 rows) and face recognition (the other rows)). The results show that DDS outperformed MMD in 26 transfer tasks. Bold values represent the highest accuracy. Please note that 1-NN means the 1 nearest neighbor and NN means the neural network. . . 75

4.1 Specifications of $\mathbb{P}$ and $\mathbb{Q}$ of synthetic datasets. $\mu_1^b = [0,0], \mu_2^b = [0,1], \mu_3^b = [0,2], \ldots, \mu_8^b = [2,1], \mu_9^b = [2,2]$ (same with Figure 4.1a). $\mu_1^h = \mathbf{0}_d$, $\mu_2^h = 0.5 \times \mathbf{1}_d$, I_d is an identity matrix with size d. $\Delta_i^b = -0.02 - 0.002 \times (i-1)$ if $i < 5$ and $\Delta_i^b = 0.02 + 0.002 \times (i-6)$ if $i > 5$. if $i = 5$, $\Delta_i^b = 0$ (same with Figure 4.1a). Δ_1^h and Δ_2^h are set to 0.5 and -0.5, respectively. 97

4.2 *Higgs* ($\alpha = 0.05$): average test power±standard error for N samples. Bold represents the highest mean per row. 97

4.3 Results on *Higgs* ($\alpha = 0.05$). We report average Type I error on *Higgs* dataset when increasing number of samples (N). Note that, in *Higgs*, we have two types of Type I errors: 1) Type I error when two samples drawn from $\mathbb{P}$ (no Higgs bosons) and 2) Type I error when two samples drawn from $\mathbb{Q}$ (having Higgs bosons). Type I reported here is the average value of 1) and 2). Since Type I error reported here is the average value of two average Type I errors, we do not report standard errors of the average Type I error in this table. 98

4.4 *MNIST* ($\alpha = 0.05$): average test power±standard error for comparing N real images to N DCGAN samples. 98

4.5 Results on *MNIST* given $\alpha = 0.05$. We report average Type I error±standard errors on real-*MNIST* vs. real-*MNIST* when increasing number of samples (N). 98

4.6 *CIFAR-10.1* ($\alpha = 0.05$): mean rejection rates. 101

4.7 Mean test power on *Blob* ($n_b = 40$), *HDGM* ($N = 4000, d = 10$), *Higgs* ($N = 3000$) and *MNIST* ($N = 400$) for $\alpha = 0.05$. See Section 4.7.2 for the naming scheme; S+C corresponds to C2ST-S, L+C to C2ST-L, and D+J to MMD-D. L+M is the method proposed by Kirchler et al. [2019]. 103

4.8 Paired t-test results ($\alpha = 0.05$) for the results of Section 4.7.1. For *HDGM*, we fix $d = 10$ (corresponding to Figure 4.3a). ✓ indicates MMD-D achieved statistically significantly higher mean test power than the other method, × that it did not. 103

5.1 Target-domain accuracy on 8 digit WUDA tasks (*SYND*↔*MNIST*). Bold value represents the highest accuracy in each row. 140

5.2 Target-domain accuracy on 12 human-sentiment WUDA tasks with the 20% noise rate. Bold values mean the highest values in each row. 143
5.3 Target-domain accuracy on 12 human-sentiment WUDA tasks with the 45% noise rate. Bold values mean the highest values in each row. 143
5.4 Target-domain accuracy on 3 real-world WUDA tasks. The source domain is the *Bing* dataset that contains noisy information from the Internet. Bold value represents the highest accuracy in each row. . . 144
5.5 Results of ablation study. Average target-domain accuracy on 8 simulated digit WUDA tasks (*Digit*), 24 simulated human-sentiment WUDA tasks (*Sentiment*) and 3 real-world WUDA tasks (*Real-world*). Bold value represents the highest accuracy in each row. 147

6.1 Description of the original datasets. 178
6.2 Transfer tasks (10 tasks in total). 179
6.3 Same-domain accuracy of each target domain using 5-fold SVM. . . . 183
6.4 The classification results for RMG and CM. 185
6.5 The results of the MMD test for the mapped and adapted domains in two extreme situations (lowest and highest accuracy) of task CD2CO among 50-time experiments. 186
6.6 The classification results (AVG±STD) for GLG and benchmark models. Bold values represent the lowest average accuracy in each task. . . . 190

7.1 The overall performance of each method on four tasks. 219
7.2 The overall STD value of each method on four tasks. 221
7.3 The prediction performance of F-HeUDA When chaning α. 223

8.1 Description of the three datasets. 251

8.2 Classification accuracy of the baselines and SFERNN on the Australian credit task (Task 1). SFERNN can extract useful information from both the *source domains* (SDs) and obtain better average accuracy on the *target domain* (TD) than all the baselines, no matter which source domain these baselines select. 254

8.3 Classification accuracy of the baselines and SFERNN on the German credit task (Task 2). SFERNN can extract useful information from both the *source domains* (SDs) and obtain a better average accuracy on the *target domain* (TD) than all the baselines, no matter which source domain these baselines select. 256

8.4 Classification accuracy of the baselines and SFERNN on the Japanese credit task (Task 3). SFERNN can extract useful information from both the *source domains* (SDs) and obtain a better average accuracy on the *target domain* (TD) than all the baselines, no matter which source domain these baselines select. 257

8.5 STD of classification accuracy of baselines and SFERNN on the Australian credit task (Task 1). SFERNN can use knowledge in the *source domains* (SDs) to obtain more stable accuracy across 50 experiments on the *target domain* (TD) than most baselines. 259

8.6 The STD of the classification accuracy of the baselines and SFERNN on the German credit task (Task 2). SFERNN can use knowledge in *source domains* (SDs) to obtain more stable accuracy across 50 experiments on the target domain (TD) than most baselines. 260

8.7 The STD of classification accuracy of the baselines and SFERNN on the Japanese credit task (Task 3). SFERNN can use knowledge in *source domains* (SDs) to obtain more stable accuracy across 50 experiments on the *target domain* (TD) than most baselines. 261

CHAPTER 1

INTRODUCTION

1.1 Background

One important goal for artificial intelligence is to use computers to recognize new data patterns based on existing knowledge. Early on in the field's development, researchers used existing data to train a classifier to predict the labels of future samples. This approach is now commonly regarded as conventional machine learning. However, conventional machine learning methods share a common assumption: training data and testing data are from the same distribution. If training data and testing data are from different distributions, the negative influence that has on the testing set was revealed [Ben-David and Blitzer, 2006]. Hence, transfer learning methods were proposed to minimize this divergence and learn a classifier with stronger generalization-ability for both sets. These transfer learning methods attracted significant attention [Ganin et al., 2016a;

Long et al., 2018; Pan and Yang, 2010], and their training sets and testing sets were extended to more general concepts: source domains and target domains [Liu et al., 2018b; Lu et al., 2015].

Subsequently, researchers considered how to leverage the knowledge in a source domain to help predict the labels in a target domain. For example, the knowledge gained from recognizing cars could be used to help recognize trucks, value predictions for US real estate could help predict real estate values in Australia, or knowledge learned by classifying English documents could be used to help classify Spanish documents. As such, transfer learning methods are widely used in many applications [Chalmers et al., 2018; Liu et al., 2019; Zhao et al., 2017, 2014; Zhuo and Yang, 2014].

Of the proposed transfer learning methods, domain adaptation methods have demonstrated good success in various practical applications in recent years [Courty et al., 2017; Ghifary et al., 2017; Gong et al., 2019]. Most domain adaptation methods focus on *homogeneous unsupervised domain adaptation* (HoUDA); that is, where the source and target domains have similar, same-dimensionality feature spaces and there are no labeled samples in the target domain [Ben-David et al., 2010a]. Nevertheless, given the time and cost associated with human labeling, target domains are heterogeneous[1]. Thus, heterogeneous domain adaptation methods are proposed to transfer knowledge from a source domain to a heterogeneous target domain.

[1]In the field of domain adaptation, “heterogeneity" often represents that 1) dimensionality of source and target domains are different and 2) features of two domains are disjoint.

1.2 Motivation

Since the data volume tends to be very large in the current era, it is impossible for us to annotate every sample we have seen. As a result, most datasets only contain very few labels or no labels. To classify samples in these datasets, we are encouraged to use transfer learning methods to leverage knowledge from labeled datasets to help train a classifier on few-labeled or unlabeled datasets. However, most transfer learning methods require certain assumptions to ensure their efficacy. These assumptions are probably not realistic. For example, 1) the homogeneous unsupervised domain adaptation methods assume that all labels in the source domain must be *true labels* and 2) the heterogeneous domain adaptation methods assume that the target domain must have *few labels* or have *paired samples* with the source domain. These assumptions greatly limit us to apply existing transfer learning methods in real-world scenarios.

To remove the assumptions in existing transfer learning methods and move forward to realistic transfer learning, this thesis concludes four unsolved challenges faced by existing methods and proposes corresponding theory and methods to address the four challenges. The four challenges are 1) how to reliably test if the feature spaces of two domains are from different distributions; 2) how to reliably transfer knowledge from a source domain to a target domain when source domain contains noisy labels, 3) how to reliably transfer knowledge from a source domain to a heterogeneous target domain where there are no labeled samples available and 4) how to reliably transfer knowledge from multiple source domains to a heterogeneous and unlabeled target domain. This thesis gives comprehensive analysis and solutions to all the above-mentioned challenges.

1.3 Research Questions and Objectives

This thesis aims to develop a set of theory and methods towards realistic transfer learning and will answer the following research questions:

RESEARCH QUESTION 1 (RQ1): *how to reliably test if the feature spaces of two domains are from different distributions?*

Determining if two sets of samples are from the same distribution is the prerequisites of using transfer learning methods. If both are from different distributions, we need to use transfer learning methods to eliminate the distributional discrepancy between them. If not, there is no need to consider transfer learning methods. To do so, researchers directly adopt existing two-sample tests to show when we need transfer learning methods [Ben-David et al., 2010b; Pan and Yang, 2010]. However, most existing two-sample tests focus on simple scenarios, such as two sets of samples from Gaussian distributions, or have strong assumptions, which are not applicable to various complex domains [Chen and Friedman, 2017; Gretton et al., 2012]. Such complex domains include RGB images, videos and high-sparse data. Thus, it is necessary to propose new two-sample tests for modern machine-learning datasets. Moreover, the test statistics in two-sample tests can also be used to measure the discrepancy between two domains.

RESEARCH QUESTION 2 (RQ2): *how to reliably transfer knowledge from a source domain to a target domain when labels in the source domain cannot be perfectly annotated (i.e., the source domain contains noisy labels)?*

In the wild, the data volume of source domain tends to be large [Tan et al., 2014]. To avoid the expensive labeling cost, labeled data in source domain normally come from amateur annotators or the Internet [Lee et al., 2018; Schroff

et al., 2011; Tommasi and Tuytelaars, 2014]. Unfortunately, existing HoUDA methods share an implicit assumption that *there are no noisy source data* [Shu et al., 2019; Yu et al., 2017]. Therefore, existing methods cannot well handle HoUDA in the noisy scenario. To validate this fact, we empirically reveal the deficiency of existing HoUDA methods (Figure 5.2, e.g., DAN [Long et al., 2015] and DANN [Ganin et al., 2016a]).

RESEARCH QUESTION 3 (RQ3): *how to reliably transfer knowledge from a source domain to a heterogeneous and unlabeled target domain?*

Since unlabeled data in target domain can be easily obtained, considering *heterogeneous unsupervised domain adaptation* (HeUDA) setting has the greatest potential in real world [Saito et al., 2018]. Although existing HeUDA methods can transfer knowledge from a source domain to a heterogeneous target domain, they still need parallel sets to bridge two heterogeneous domains, which is not realistic. For example, credit assessment data are confidential and private, and the information of each sample cannot be accessed. Thus, we cannot find similar instances between two credit-assessment domains. Namely, parallel sets (needed by existing HeUDA methods) do not often exist. To the best of our knowledge, little theoretical discussion has taken place in regard to the absence of a parallel set in the HeUDA. This limits HeUDA methods to be used in more scenarios.

RESEARCH QUESTION 4 (RQ4): *how to reliably transfer knowledge from multiple source domains to a heterogeneous and unlabeled target domain?*

For existing HoUDA methods, they assume that source data come from the single source domain (i.e., single-source scenario [Long et al., 2019]) or from multiple source domains whose feature spaces have the same dimension (i.e., multi-homogeneous-source scenario [Hoffman et al., 2018a]). Namely, we can only

use labeled data from single source domain or from multiple homogeneous source domains to train a classifier for unlabeled data in target domain. However, in real world, given a target domain, we probably have multiple different-dimension (*heterogeneous*) source domains, which does not satisfy the assumption of existing HoUDA methods.

For example, assume that our task is to assess Japanese (J) credit using 15 features (a target domain only containing unlabeled data), where the meanings of these features are masked to protect private information. Furthermore, we have German (G) and Australian (A) credit assessment datasets (two source domains containing labeled data), where the number of features in G and A are 24 and 14, respectively. Then, we want to train a classifier with data from G, A and J to assess Japanese credit (a UDA problem). However, existing UDA methods can only use knowledge from G or A rather than G and A. This results in a serious problem: *we can only use part of labeled data to help assess Japanese credit and we must abandon the other part.* Thus, it is necessary to investigate how to reliably transfer knowledge from multiple source domains to a heterogeneous and unlabeled target domain.

This thesis aims to achieve the following objectives, which are expected to answer the above research questions:

RESEARCH OBJECTIVE 1 (RO1): *To propose new nonparametric two-sample tests for modern machine-learning datasets.* (aims to answer RQ1)

There are two main types of two-sample tests: parametric and nonparametric. Parametric tests are based on the assumption that two distributions have a known form, e.g., a normal distribution. However, it is difficult to confirm whether one sample has a specific form of distribution, parametric tests have limited

applications in real world [Chen and Friedman, 2017].

In contrast, nonparametric tests allow for a direct comparison between two distributions without making any assumptions (see [Gibbons and Chakraborti, 2011] for a survey). For this reason, nonparametric tests have attracted the attention of many machine-learning researchers over the last decade. Nonparametric tests can be included in three main categories: f-divergence-based tests [Nguyen et al., 2010], function-based tests [Jitkrittum et al., 2016] and subset-based tests [Lu et al., 2016].

To propose new two-sample tests for modern machine-learning datasets, we will address key problems faced by function-based tests and subset-based tests. As a result, we will propose one subset-based two sample test for low-dimension data and one function-based two sample test for high-dimension data. Corresponding theories will also be proved to ensure their efficacy. Note that, since f-divergence-based tests are mostly used on one-dimension data, this study does not focus on f-divergence-based tests.

RESEARCH OBJECTIVE 2 (RO2): *To build the theoretical foundation and propose a new method for HoUDA under noisy scenario.* (aims to answer RQ2)

Since the foundation of HoUDA methods is all based on [Ben-David et al., 2010b] that assumes that source domain only contains clean labels, we will first, in theory, to propose a new theoretical bound for HoUDA problem under noisy scenario. This new bound will reveal why existing HoUDA methods do not work well under noisy scenario. Guided by the new bound, we will propose a solution to HoUDA problem under noisy scenario and validate it using real-world datasets.

RESEARCH OBJECTIVE 3 (RO3): *To build the theoretical foundation and propose new methods for heterogeneous unsupervised domain adaptation.* (aims

to answer RQ3)

Due to the lack of theoretical foundations for *heterogeneous unsupervised domain adaptation* (HeUDA) problem, HeUDA methods assume the existence of parallel sets that can bridge two heterogeneous domains, which is not realistic. To remove this assumption, we will first build the theoretical foundation to show when and how we can transfer transfer knowledge from a source domain to a heterogeneous and unlabeled target domain. The theory will help us prevent the negative transfer and reliably transfer knowledge across two domains. Based on the proposed theory, we will propose valid HeUDA methods and evaluate these methods using real-world datasets.

RESEARCH OBJECTIVE 4 (RO4): *To develop a new method to transfer knowledge from multiple source domains to a heterogeneous target domain where there are no labeled samples available.* (aims to answer RQ4)

In real world, given a target domain, we probably have multiple different-dimension (heterogeneous) source domains. Thus, to step towards realistic transfer learning, we will consider to extend the HeUDA problem to *multi-source HeUDA* (MsHeUDA) problem. After formalizing MsHeUDA problem, we will propose a valid method to address this problem and use real-world datasets to evaluate its efficacy.

1.4 Research Innovation and Contributions

This thesis aims to step forwards realistic transfer learning by addressing key problems faced by existing transfer learning methods. The main contributions of this study are summarised as follows:

1.4.1 Research Innovation

Innovation 1. This study is the first to propose a continuous and differentiable subset-based nonparametric two-sample test statistic. Compared to existing subset-based test statistics, DDS can be used to derive gradients, which is useful for many machine learning problems, e.g., domain adaptation.

Innovation 2. This study is the first to construct a valid deep kernel to help test if two sets of samples are from the same distribution. Since the proposed deep kernel is a non-translation-invariant kernel, it overcomes many drawbacks of translation-invariant kernels (e.g., Gaussian kernel). Due to the non-translation-invariant property, the deep-kernel-based nonparametric two-sample test is more suitable for complex and high-dimension data, e.g., RGB images (common type of data in real world).

Innovation 3. This study is the first to formalize the *wildly unsupervised domain adaptation* (WUDA) problem, where classifiers for the target domain have to be trained with *noisy* labeled data from source domain and unlabeled data from target domain. WUDA is a new, more realistic and more challenging problem setting than unsupervised domain adaptation problem, a new theoretical bound is proved to build a theoretical foundation for WUDA.

Innovation 4. This study is the first to build a theoretical foundation for HeUDA problem and propose a theoretical-guaranteed HeUDA method. An unsupervised knowledge transfer theorem is proved in this study. This theorem can help prevents negative transfer for HeUDA methods. We empirically shows

that existing deep domain adaptation methods will cause extreme negative transfer if taking no care of conditions of the theorem.

Innovation 5. This study is the first to formalize the *multi-source HeUDA* (MsHeUDA) problem. Compared to existing problem settings in the field of domain adaptation, MsHeUDA is more realistic and difficult and cannot be solved using existing methods.

1.4.2 Research Contributions

Contribution 1. A new subset-based two-sample test, called DDS, is proposed for low-dimension machine-learning datasets.

1) This study proves the following properties of DDS: (a) its empirical form is unbiased; (b) the convergence rate of its empirical form benefits from the dimensionality of the samples; and (c) it is semimetric in a space consisting only of Borel probability measures.

2) This study proves asymptotic behavior of the empirical DDS, which provides a solid foundation for DDS-based two-sample test. A DDS-based domain adaptation bound is proved, which highlights that generalization performance in unsupervised domain adaptation problem is controlled by the empirical DDS.

Contribution 2. A new function-based two-sample test, called MMD-D, is proposed for high dimension machine-learning datasets.

1) This study provides the first proof of consistency for the proposed test, which applies both to kernels on deep features and to simpler radial basis kernels or multiple kernel learning.

2) This study evaluates the proposed test on several simulated and real-world datasets, including complex synthetic distributions, high-energy physics data, and RGB images.

Contribution 3. A novel method, called Butterfly-Net, is proposed to address *wildly unsupervised domain adaptation* (WUDA) problem.

1) This study theoretically proves that WUDA ruins all UDA methods if taking no care of label noise in source domain. An upper bound of target-domain risk is also proved for WUDA problem.

2) This study presents a principle-guided *Butterfly* framework, a powerful and efficient solution to WUDA.

Contribution 4. This study presents a theoretical-guaranteed HeUDA method, called GLG.

1) This study presents an effective heterogeneous unsupervised domain adaptation method, called GLG. GLG is able to transfer knowledge from a source domain to an unlabeled target domain in settings where both domains have heterogeneous feature spaces and are free of parallel sets.

2) This study presents a new principal angle based metric to show the extent to which homogeneous representations have preserved the geometric distance between the original domains. The proposed metric also reveals the relationship between two heterogeneous feature spaces.

Contribution 5. This study presents a method, called F-HeUDA, to address HeUDA problem when two domains are imbalanced.

1) This study presents a novel F-HeUDA method, using n-dimensional fuzzy geometry and fuzzy equivalence relations to address HeUDA problem. Both fuzzy technologies successfully overcome the drawbacks of CCA and the Granssmann manifold (two domains must have the same number of instances).

2) This study investigated two important properties of fuzzy equivalence relations, which are the theoretical guarantees of the proposed method and key parts of the proposed method.

Contribution 6. This study presents an effective solution, called SFERNN, to solve the Multi-source HeUDA problem.

1) This study presents a SFERNN method to solve the multi-source HeUDA problem. SFERNN is based on a novel fuzzy equivalence relation, called *multi-source shared fuzzy equivalence relations* (MsSFER). Compared to traditional fuzzy equivalence relations and shared fuzzy equivalence relations, MsSFER can extract shared fuzzy information contained in multiple heterogeneous domains.

2) The proposed method, SFERNN, is the first method that has the ability to extract the shared fuzzy information of multiple heterogeneous domains. Previous methods can only extract the shared information of multiply homogeneous domains or two heterogeneous domains.

1.5 Research Significance

The theoretical and practical significance of this thesis is summarised as follows:

Theoretical significance: This thesis investigates two two-sample tests and proves their theoretical properties and builds theoretical foundations for three

new research directions in the field of unsupervised domain adaptation. These theoretical results have the greatest potential to guide future researchers to develop more realistic transfer learning methods.

Researchers can follow the proposal of DDS to convert statistics of more subset-based two-sample tests to be continuous and differentiable, which will enable these statistics to be applied to address more machine learning problems. Meanwhile, using the proof skills that are used to prove properties of DDS, researchers can also prove properties of statistics of other subset-based two-sample tests.

Since it is the first time that deep kernel is constructed to solve two-sample test problem and we provides the first proof of consistency for the proposed test, the proofs skills used in this thesis can greatly help researchers develop more deep kernels with theoretical guarantees. The proof of the consistency can apply to kernels on deep features and to simpler radial basis kernels or multiple kernel learning.

This thesis opens a new research direction for the field of domain adaptation: wildly unsupervised domain adaptation, which is more realistic than unsupervised domain adaptation. It has been theoretically shown that existing unsupervised domain adaptation methods cannot address this new problem. New theoretical bounds are also provided to help better understanding the new problem. Based on these theoretical bounds, in the future, researchers can develop more methods to address this new problem and prove their methods' generalization bound and estimation error bound using proof skills in this thesis.

This thesis is the first to to build a theoretical foundation for heterogeneous unsupervised domain adaptation problems (including multi-source scenario). The

theoretical results show that existing unsupervised domain adaptation methods will suffer from the issue of negative transfer if they do not take care the conditions of the proposed theorem. Based on the theoretical results presented in this thesis, in the future, researchers can develop more methods to address heterogeneous unsupervised domain adaptation problem under theoretical guarantees.

Practical significance: The findings of this research contribute to the benefit of society given the increasing demand of cross-domain classification in modern life. This study presents two two-sample tests for modern machine-learning datasets and four methods to transfer knowledge across source and target domains. All of these tests and methods are validated by real-world datasets, which means practitioners can directly use the proposed ones to solve real-world problems. The findings help resolve the real-world problems of transfer learning. There is potential of many other applications that could benefit from this study, such as computer vision, natural language processing and recommender systems.

1.6 Thesis Structure

The structure of the thesis is shown in Figure 1.1 and the chapters are organised as follows:

- CHAPTER 2 presents the literature of transfer learning, two-sample test and domain adaptation, thereby revealing limitations of current research.
- CHAPTER 3 presents a novel subset-based statistic, called DDS, that is continuous, differentiable, and able to automatically confirm the size of each subset. The proposed statistic naturally has advantages of subset-based

statistics and is beneficial for solving two important problems: two-sample test problem and unsupervised domain adaptation problem. This chapter addresses RQ1 to achieve RO1 when facing low-dimension data.

- CHAPTER 4 presents a class of kernel-based two-sample tests, which aim to determine whether two sets of samples are drawn from the same distribution. Our tests are constructed from kernels parameterized by deep neural nets, trained to maximize test power. These tests adapt to variations in distribution smoothness and shape over space, and are especially suited to high dimensions and complex data. This chapter addresses RQ1 to achieve RO1 when facing high-dimension and complex data.

- CHAPTER 5 presents a new, more realistic and more challenging problem setting, where classifiers have to be trained with *noisy* labeled data from source domain and unlabeled data from target domain—we name it *wildly unsupervised domain adaptation* (WUDA). We show that WUDA ruins all UDA methods if taking no care of label noise in source domain, and to this end, we propose a *Butterfly* framework, a powerful and efficient solution to WUDA. This chapter addresses RQ2 to achieve RO2.

- CHAPTER 6 presents: 1) an unsupervised knowledge transfer theorem that guarantees the correctness of transferring knowledge; and 2) a principal angle-based metric to measure the distance between two pairs of domains: one pair comprises the original source and target domains and the other pair comprises two homogeneous representations of two domains. The theorem and the metric have been implemented in an innovative transfer method, called a *Grassmann-Linear monotonic maps-geodesic flow kernel*

(GLG), that is specifically designed for *heterogeneous unsupervised domain adaptation* (HeUDA). This chapter addresses RQ3 to achieve RO3 when two domains are balanced.

- CHAPTER 7 proposes a novel F-HeUDA method, adopting n-dimensional fuzzy geometry and fuzzy equivalence relations to address heterogeneous domain adaptation issues. Both fuzzy technologies successfully overcome the drawbacks of canonical correlation analysis and the Granssmann manifold (two domains must have the same number of instances). As a result, the proposed method can transfer more knowledge from a source domain to a target domain than existing methods when there are very few instances in the target domain. This chapter addresses RQ3 to achieve RO3 when two domains are imbalanced.

- CHAPTER 8 presents a *shared-fuzzy-equivalence-relations neural network* (SFERNN) for addressing multi-source heterogeneous unsupervised domain adaptation problem. SFERNN is a five-layer neural network containing c source branches and one target branch. The network structure of SFERNN is first confirmed by a novel fuzzy relation called *multi-source shared fuzzy equivalence relations*. Then, we optimize parameters of SFERNN via minimizing cross-entropy loss on c source branches and the distributional discrepancy between each source branch and the target branch. This chapter addresses RQ4 to achieve RO4.

- CHAPTER 9 summarises the findings of this thesis and points to directions for future work.

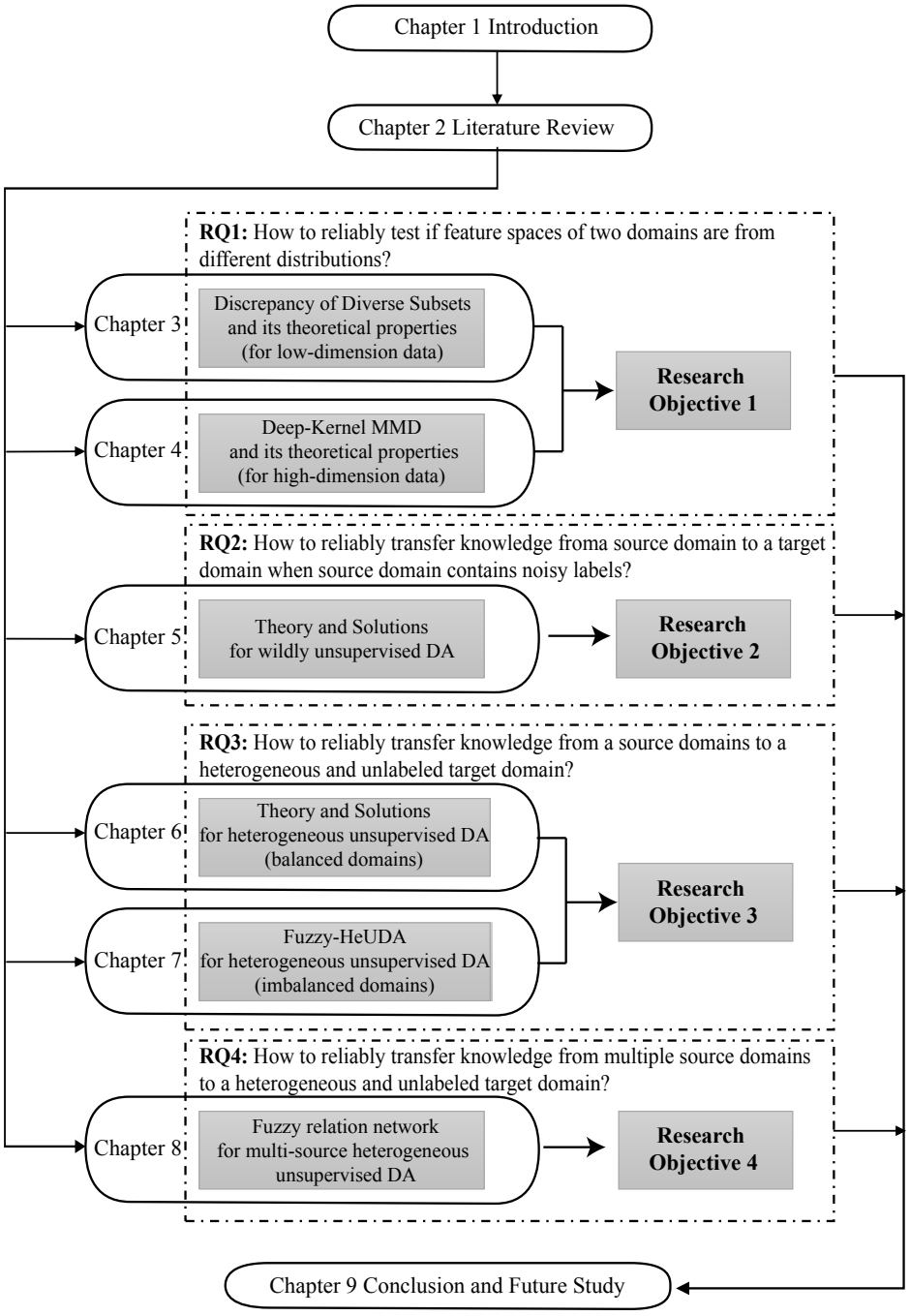


Figure 1.1: Thesis structure

CHAPTER 2

LITERATURE REVIEW

In this chapter, we review recent papers related to this thesis, which includes three main parts: transfer learning, two-sample test and domain adaptation.

2.1 Transfer Learning

Since transfer learning methods can help leverage knowledge in a source domain to help train a classifier or predictor for the target domain. They are widely used to address many problems, such as object recognition [Luo et al., 2018], face recognition [Long et al., 2013], AI planning [Zhuo and Yang, 2014], reinforcement learning [A. C. Bianchi et al., 2015; Nguyen et al., 2017], recommender systems [Zhao et al., 2017], and natural language processing [Zhao et al., 2014].

Since most prior methods have not simultaneously reduced the difference in both the marginal distribution and conditional distribution between domains,

Long et al. [2013] proposed a method to jointly adapt both the marginal distribution and conditional distribution in a principled dimensionality reduction procedure, and construct new feature representation that is effective and robust for substantial distribution difference. Due to the merits of adversarial-training strategy, conditional domain adaptation network is proposed to use outer product to generate better feature representations of domains [Long et al., 2018]. The proposed methods are successfully used to recognize objects and faces across two domains.

In the filed of AI planing, a novel algorithm framework, called TRAMP, is proposed by Zhuo and Yang [2014]. The proposed method can learn action models with limited training data in a target domain, via transferring as much of the available information from source domains. TRAMP is evaluated in different settings to see their advantages and disadvantages in six planning domains, including four international planning competition domains.

To apply transfer learning method to the field of reinforcement learning, a scalable methodology is introduced to learn and transfer knowledge of the transition (and reward) methods for model-based reinforcement learning in a complex world [Nguyen et al., 2017]. Recently, Gamrian and Goldberg [2019] demonstrate that a trained agent fails completely when facing small visual changes, and that fine-tuning—the common transfer learning paradigm—fails to adapt to these changes, to the extent that it is faster to re-train the model from scratch. To propose a better method, Gamrian and Goldberg [2019] separated the visual transfer task from the control policy and then achieved substantially better sample efficiency and transfer behavior. It allows an agent trained on the source task to transfer well to the target tasks.

A novel transfer learning framework for recommender systems is proposed in [Zhao et al., 2017], which can facilitate flexible knowledge transfer across different systems with low cost by using an active learning principle to construct entity correspondences across systems. Since many recommender systems suffer from cold-start problems due to a lack of sufficient preference data. Cross-domain recommender systems have been proposed as one possible solution [Zhang et al., 2017]. The proposed solution outperforms five benchmarks and increases the accuracy of recommendations in the target domain, especially with sparse data [Zhang et al., 2017].

A novel machine learning framework called online transfer learning is proposed by Zhao et al. [2014], which aims to attack an online learning task on a target domain by transferring knowledge from some source domain. They proposed effective algorithms to solve online classification tasks, and validated their algorithms using natural language processing tasks. Huang and Paul [2019] describes two complementary ways to adapt classifiers to shifts across time. First, they present a simple method for constructing the diachronic word embeddings. Second, They propose a time-driven neural classification model inspired by methods for domain adaptation. Experiments on six corpora show how these methods can make classifiers more robust over time.

Although recent research of transfer learning has shown a decent ability to transfer knowledge from a source domain to a target domain, most research require certain assumptions to ensure their efficacy. These assumptions are probably not realistic, which means that existing transfer learning methods still face several unsolved and challenging problems in real world. These problems are closely related to 1) two-sample test and 2) domain adaptation, which are

reviewed in the following sections in this chapter.

2.2 Two-sample Test

Two-sample test is a fundamental task in both statistical data processing [Chen and Friedman, 2017; Ghoshdastidar et al., 2017; Gibbons and Chakraborti, 2011] and machine learning [Ghifary et al., 2017; Lu et al., 2016; Pan et al., 2011]. It is a hypothesis test aiming to determine whether two sets of samples are drawn from the same distribution. The statistic used in testing is an useful tool in these tasks across a wide range of research topics, such as two-sample tests [Chwialkowski et al., 2015; Gretton et al., 2012; Jitkrittum et al., 2016], domain adaptation [Hoffman et al., 2013; Long et al., 2017; Yamada et al., 2011] and concept drift [Lu et al., 2016, 2014; Reis et al., 2016].

There are two main types of two-sample tests: parametric and nonparametric. Parametric tests are based on the assumption that two distributions have a known form, e.g., a normal distribution. The power of parametric tests, however, quickly decrease as dimensionality increases unless strong assumptions are made when attempting to estimate a large number of parameters, such as a covariance matrix [Chen and Friedman, 2017]. Additionally, since it is difficult to confirm whether one sample has a specific form of distribution, parametric tests have limited applications in the real world.

In contrast, nonparametric tests allow for a direct comparison between two distributions without making any assumptions about the probability density functions (pdf) of the two distributions. Some well-known examples include the Kolmogorov-Smirnov test, the Wilcoxon test, the Wald-Wolfowitz run test (see

[Gibbons and Chakraborti, 2011] for a survey). For this reason, nonparametric tests have attracted the attention of many machine-learning researchers over the last decade. The definition of Nonparametric tests is demonstrated as follows.

Definition 2.1 (Nonparametric two-sample test)**.** *Let x_1 and x_2 be multivariate random variables defined on a topological space $\mathcal{X} \subseteq \mathbb{R}^d$, with respective p and $q \in \mathcal{P}(\mathcal{X})$, where $x_1 : \mathcal{X} \to \mathcal{X}$, $x_2 : \mathcal{X} \to \mathcal{X}$, $\mathcal{P}(\mathcal{X})$ consists of all Borel probability measures on $\mathcal{X}$. Given i.i.d. observations $X_1 := \{x_{11},...,x_{1n_1}\}$ and $X_2 := \{x_{21},...,x_{2n_2}\}$ from p and q, respectively, can we decide whether $p \neq q$ through X_1 and X_2?*

Nonparametric tests can be included in three main categories: f-divergence-based tests [Kanamori et al., 2009; Nguyen et al., 2010; Sugiyama et al., 2008], function-based tests [Gretton et al., 2012; Jitkrittum et al., 2016] and subset-based tests [Liu et al., 2018a; Lu et al., 2016, 2014].

2.2.1 F-divergence based Non-parametric Two-sample Tests

f-divergence-based nonparametric statistics are also known as Ali-Silvey-Csiszár divergences. Some well-known examples include *Kullback-Leibler* (KL) divergence and *Pearson* (PE) divergence. Benefiting from the development of kernel density estimation, the ratio of two distributions is estimated without knowing the pdfs of the distributions [Nguyen et al., 2010; Sugiyama et al., 2008]. Using the estimations, KL divergence and PE divergence [Kanamori et al., 2009; Yamada et al., 2011, 2013] calculate the distance between two distributions.

However, f-divergence-based statistics assume that the two probability measures are absolutely continuous, which indicates that the support sets for the two distributions are the same. Without this assumption, density-ratios may diverge to infinity even in rather simple settings [Cortes et al., 2010]. In [Yamada et al., 2011], Yamada et al. proposed a relative density-ratio estimation method to overcome the drawback of f-divergence-based nonparametric statistics but a hyperparameter is needed to confirm weights of two distributions.

2.2.2 Function based Non-parametric Two-sample Tests

Function-based statistics (also known as integral probability metrics [Sriperumbudur et al., 2010]) are based on Lemma 9.3.2 in [Dudley, 2002], which indicates that two distributions can be compared via their images of a series of functions. These functions map two distributions onto other spaces, and their images of these functions are regarded as features of two distributions. Two types of functions can be used as a basis for function-based nonparametric statistics: i) reproducing kernel functions and ii) q-Lipschitz functions.

2.2.2.1 Reproducing kernel functions

A popular class of function based non-parametric two-sample tests is based on kernel methods [Smola and Schölkopf, 2001]: such tests construct a *kernel mean embedding* [Berlinet and Thomas-Agnan, 2004; Muandet et al., 2017] for each distribution, and measure the difference in these embeddings.

For any *characteristic* kernel, two distributions are the same if and only if their mean embeddings are the same; the distance between mean embeddings is

the *maximum mean discrepancy* (MMD) [Gretton et al., 2012]. There are also several closely related methods, including tests based on checking for differences in mean embeddings evaluated at specific locations [Chwialkowski et al., 2015; Jitkrittum et al., 2016] and kernel Fisher discriminant analysis [Harchaoui et al., 2007]. These tests all work well for samples from simple distributions when using appropriate kernels.

2.2.2.2 Lipschitz functions

With the exception of reproducing kernels, q-Lipschitz functions are used to construct features to compare two distributions. Wasserstein statistic and earth-mover statistic [Arjovsky et al., 2017] are representative examples of statistics that are built on q-Lipschitz functions. The Lipschitz semi-norm of this kind of functions is less than or equal to q.

Function-based statistics have been widely used in machine learning problems, e.g., domain adaptation [Ghifary et al., 2017; Pan et al., 2011] and generative adversarial networks [Arjovsky et al., 2017]. The estimators of function-based statistics can be found in [Sriperumbudur et al., 2010]; the paper [Sriperumbudur et al., 2010] also highlights function-based statistics are more suitable for comparing two distributions than f-divergence-based statistics in general cases, say, where two distributions have a disjointed support set.

2.2.2.3 Limitations

However if distributions are with complex structure, simple kernels will often map distinct distributions to nearby (and hence hard to distinguish) mean embeddings. Figure 4.1a shows an example of a multimodal dataset, where the

overall modes align but the sub-mode structure varies differently at each mode. A translation-invariant Gaussian kernel only "looks at" the data uniformly within each mode (see Figure 4.1b), requiring many samples to correctly distinguish the two distributions. The distributions can be distinguished more effectively if we understand the structure of each mode, as with the more complex kernel illustrated in Figure 4.1c.

2.2.3 Subset based Non-parametric Two-sample Tests

Subset-based tests measure two probability distributions by discovering representative subsets (regions) in the sample space and then comparing the number of observations drawn from each of the distributions within each subset. These tests are based on the definition of probability density: if two distribution are the same, the number of observations drawn from two distributions will be the same within each subset of the sample space. Existing subset-based nonparametric tests are divided into two types: nearest neighbor and space partitioning.

2.2.3.1 Nearest neighbor based tests

Schilling [1986] proposed a multivariate two-sample test based on the possibly weighted proportion of all k-nearest neighbor comparisons in which observations and their neighbors belong to the same sample. Asymptotic null distributions are explicitly determined and shown to involve certain nearest neighbor interaction probabilities.

Henze [1988] used a similar k-nearest neighbor-based approach. Subsequently, Rosenbaum [2005] considered a k-nearest neighbor-based multivariate

two-sample test from the interpoint distance perspective. Interpoint distances are used to construct an optimal non-bipartite matching, i.e., a matching of the observations into disjoint pairs to minimize the total distance within pairs. The resulting cross-match statistic is the number of pairs containing one observation from the first distribution and one from the second.

2.2.3.2 Space partitioning based tests

Biau and Gyorfi [2005] proposed a non-parametric test based on the ℓ_1 distance between the two empirical distributions restricted to a finite partition. However, the authors conclude that choosing a perfect partition is difficult. Hence, subsequent researchers proposed space partitioning from a competence model perspective, where the partition is constructed in terms of related closure and related sets [Lu et al., 2016, 2014]. The distance is then defined based on the total variation.

Liu et al. [2018a] proposed a nearest neighbor-based partitioning scheme at the instance level with an accompanying partition optimization method along with proof that the proposed distance follows a normal distribution. Another topic of recent interest was studied by Lhéritier and Cazals [2018], who developed a generic sequential nonparametric testing framework in which the sample size does not need to be fixed in advance. This means the test is in effect a sequential non-parametric multivariate two-sample test, and the combination of nearest neighbor regressors and Bayesian mixtures with switching distributions yields automatic scale selection with competitive results.

2.2.3.3 Limitations

However, subset-based statistics are usually discrete functions [Lu et al., 2016, 2014; Schilling, 1986], which means the machine learning problems it can solve are limited. In the field of domain adaptation, subset-based statistics cannot be minimized with a gradient descent algorithm, so they cannot be easily used as a regularizer in the loss function to measure the distance between two distributions. In contrast, MMD can be used and, therefore, have become one of the most common function-based test statistics [Rozantsev et al., 2019].

2.3 Domain Adaptation

Of the proposed transfer learning methods, domain adaptation methods have demonstrated good success in various practical applications in recent years [Courty et al., 2017; Ghifary et al., 2017]. Most domain adaptation methods focus on *homogeneous unsupervised domain adaptation* (HoUDA); that is, where the source and target domains have similar, same-dimensionality feature spaces and there are no labeled instances the target domain [Ben-David et al., 2010a]. Nevertheless, given the time and cost associated with human labeling, target domains are heterogeneous[1] and unlabeled, which means most existing HoUDA methods do not perform well on the majority of target domains. Thus, heterogeneous domain adaptation methods are proposed to handle the situation where target domain is heterogeneous.

[1] In the field of domain adaptation, “heterogeneity" often represents that 1) dimensionality of source and target domains are different and 2) features of two domains are disjoint.

2.3.1 Homogeneous Unsupervised Domain Adaptation

We first present the definition of homogeneous unsupervised domain adaptation.

Definition 2.2 (Homogeneous unsupervised domain adaptation)**.** *Let x_s and x_t be multivariate random variables defined on a topological space $\mathcal{X} \subseteq \mathbb{R}^d$, with respective p and $q \in \mathcal{P}(\mathcal{X})$, where $x_s : \mathcal{X} \to \mathcal{X}$, $x_t : \mathcal{X} \to \mathcal{X}$, $\mathcal{P}(\mathcal{X})$ consists of all Borel probability measures on $\mathcal{X}$. Given i.i.d. observations $X_s := \{x_{s1}, ..., x_{sn_s}\}$, $X_t = \{x_{t1}, ..., x_{tn_t}\}$ from p and q, respectively and $Y_s := \{y_{s1}, ..., y_{sn_s}\}$ are ground-truth labels corresponding to X_s, where $y_{si} \in \mathcal{Y} = \{1, 2, ..., C\}$. $\mathbf{D}_s = (X_s, Y_s)$ is denoted by a source domain and $\mathbf{D}_t = (X_t)$ is denoted by a target domain. We aim to train a classifier with $\mathbf{D}_s$ and $\mathbf{D}_t$ to accurately label each instance i.i.d. drawn from q.*

To address HoUDA problem, there are four main techniques: the Grassmann manifold [Fernando et al., 2013; Gong et al., 2014; Gopalan et al., 2014], the integral probability metric [Cao et al., 2018; Gong et al., 2016; Long et al., 2016a,b; Rozantsev et al., 2019], the adversarial-training strategy [Ganin et al., 2016a; Hoffman et al., 2018b; Li et al., 2018c; Saito et al., 2018; Tzeng et al., 2017], and the pseudo-labeling strategy [Behbood et al., 2015; Long et al., 2013; Saito et al., 2017].

2.3.1.1 Grassmann-manifold-based methods

The *geodesic flow kernel* (GFK) method, as a Grassmann-manifold-based method, seeks the best of all subspaces between the source and target domains, using the geodesic flow of a Grassmann manifold to find latent spaces through integration [Gong et al., 2014]. Gopalan et al. [2014] utilized the underlying geometry of

the space of these subspaces, the Grassmann manifold, to obtain a "shortest" geodesic path between the two domains. Then they sampled points along the geodesic to obtain intermediate cross-domain data representations, using which a discriminative classifier is learnt to estimate the labels of the target-domain data.

2.3.1.2 Integral-probability-metric-based methods

For the integral-probability-metric-based methods, the first one is *transfer component analysis* (TCA) method which [Pan et al., 2011] applies MMD ([Gretton et al., 2012], an integral probability metric) to measure the distance between the source and target feature spaces, and optimizes this distance to make sure the two domains are closer than before.

The *joint distribution adaptation* (JDA) method [Long et al., 2013] improves TCA by jointly matching marginal distributions and conditional distributions. Scatter component analysis [Ghifary et al., 2017] extends TCA and JDA methods, and considers the between and within class scatter. [Wang et al., 2019] exploits intra-domain information to get a non-parametric feature and the classifier.

The *correlation alignment* (CORAL) method [Sun et al., 2016] aligns second-order statistics of source and target domain to minimize domain divergence. Manifold embedded distribution alignment [Wang et al., 2018] performs a dynamic distribution alignment in a Grassmann manifold subspace.

The wasserstein distance guided representation learning method [Shen et al., 2018] minimizes the distribution discrepancy by employing Wasserstein Distance in neural networks. *Deep adaptation networks* (DAN) [Long et al., 2019] and *joint adaptation networks* (JAN) [Long et al., 2017] employ MMD and deep neural

networks to learn the best domain-invariant representations of two domains. Deep CORAL Correlation is the extension of shallow method CORAL in deep neural network.

2.3.1.3 Adversarial-training-based methods

For the adversarial-training-based methods, representive works are *domain-adversarial training of neural networks* (DANN) [Ganin et al., 2016a] and *conditional adversarial domain adaptation* (CDAN) [Long et al., 2018].

DANN is directly inspired by the theory on domain adaptation, suggesting that predictions must be made based on features that cannot discriminate between the training (source) and test (target) domains. DANN employs a domain discriminator to recognize which domain data come from and deceives the domain discriminator by changing feature such that a invariant representation can be learn during the adversarial procession. Furthermore, CDAN utilizes the tensor product between feature and classifier predict to grasp the multimodal information and a entropy condition to control the uncertainty of classifier.

2.3.1.4 Pseudo-labeling-based method

Asymmetric Tri-training domain adaptation [Saito et al., 2017], as a pseudo-labeling-based method, is trained with labeled instances from a source domain and a pseudo-labeled target domain.

2.3.1.5 Limitations

However, in the wild, the data volume of source domain tends to be large [Tan et al., 2014]. To avoid the expensive labeling cost, labeled data in source domain

normally come from amateur annotators or the Internet [Lee et al., 2018; Schroff et al., 2011; Tommasi and Tuytelaars, 2014]. Unfortunately, existing HoUDA methods share an implicit assumption that *there are no noisy source data* [Shu et al., 2019; Yu et al., 2017]. Namely, these methods focus on transferring knowledge from clean source data to unlabeled target data. Therefore, these methods cannot well handle HoUDA in the noisy scenario. To validate this fact, we empirically reveal the deficiency of existing HoUDA methods (Figure 5.2, e.g., DAN [Long et al., 2015] and DANN [Ganin et al., 2016a]).

2.3.2 Heterogeneous Domain Adaptation

There are three types of heterogeneous domain adaptation methods: *heterogeneous supervised domain adaptation* (HeSDA), *heterogeneous semi-supervised domain adaptation* (HeSSDA), and *heterogeneous unsupervised domain adaptation* (HeUDA). Following Li et al. [2014] and Xiao and Guo [2015], “heterogeneity" in the domain adaptation field often represents the source and target features as having different dimensionality and being disjoint. For example, if 1) German credit record has 24 features and Australian credit record has 14 features and 2) features from German credit record and Australian credit record are disjoint, then we say that that German credit record and Australian credit record are heterogeneous.

HeSDA/HeSSDA aims to transfer knowledge from a source domain to a heterogeneous target domain, in which dimensions of two domains are different Duan et al. [2012a,b]. There is less literature on this setting than there is for homogeneous situations. The main methods are *heterogeneous spectral mapping*

(HeMap) [Shi et al., 2013], *manifold alignment* (MA) [Wang and Mahadevan, 2011], *asymmetric regularized cross-domain transformation* (ARC-t) [Kulis et al., 2011], *heterogeneous feature augmentation* (HFA) [Li et al., 2014], co-regularized online transfer learning [Zhao et al., 2014], *semi-supervised kernel matching for domain adaptation* (SSKMDA) [Xiao and Guo, 2015], the DASH-N method [Nguyen et al., 2015], Discriminative correlation subspace method [Yan et al., 2017] and semi-supervised entropic Gromov-Wasserstein discrepancy [Yan et al., 2018].

2.3.2.1 Heterogeneous supervised domain adaptation

We first present the definition of heterogeneous supervised domain adaptation.

Definition 2.3 (Heterogeneous supervised domain adaptation)**.** *Let x_s be multivariate random variables defined on a topological space $\mathcal{X} \subseteq \mathbb{R}^d$ with respective $p \in \mathcal{P}(\mathcal{X})$, and and x_t be multivariate random variables defined on a topological space $\mathcal{X}' \subseteq \mathbb{R}^{d'}$ with respective $q \in \mathcal{P}'(\mathcal{X}')$, where $d \neq d'$. $\mathcal{P}(\mathcal{X})$ consists of all Borel probability measures on $\mathcal{X}$, and $\mathcal{P}'(\mathcal{X}')$ consists of all Borel probability measures on $\mathcal{X}'$. Given i.i.d. observations $X_s := \{x_{s1}, ..., x_{sn_s}\}$, $X_t = \{x_{t1}, ..., x_{tn_t}\}$ from p and q, respectively, and $Y_s := \{y_{s1}, ..., y_{sn_s}\}$ are ground-truth labels corresponding to X_s, and $Y_t := \{y_{t1}, ..., y_{tn_t}\}$ are ground-truth labels corresponding to X_t, where $y_{si}, y_{tj} \in \mathcal{Y} = \{1, 2, ..., C\}$. $\mathbf{D}_s = (X_s, Y_s)$ is denoted by a source domain and $\mathbf{D}_t = (X_t, Y_t)$ is denoted by a target domain. We aim to train a classifier with $\mathbf{D}_s$ and $\mathbf{D}_t$ to accurately label each instance i.i.d. drawn from q.*

ARC-t, HFA and co-regularized online transfer learning are representative HeSDA methods. ARC-t learns an asymmetric transformation metric between

different feature spaces by exploiting labeled data from both domains [Kulis et al., 2011]. HFA first maps the data from the two domains into a common subspace in a way that allows the similarity between instances across the different domains to be measured. Then two feature transforming functions are derived, one for each domain, to augment the transformed source and the target instances with their original features and padding zeroes [Li et al., 2014]. Co-regularized online transfer learning aims to transfer knowledge from several source domains to an online learning task in a target domain [Zhao et al., 2014].

2.3.2.2 Heterogeneous semi-supervised domain adaptation

We first present the definition of heterogeneous semi-supervised domain adaptation, and then some representative methods are introduced.

Definition 2.4 (Heterogeneous semi-supervised domain adaptation)**.** *Let x_s be multivariate random variables defined on a topological space $\mathcal{X} \subseteq \mathbb{R}^d$ with respective $p \in \mathcal{P}(\mathcal{X})$, and and x_t be multivariate random variables defined on a topological space $\mathcal{X}' \subseteq \mathbb{R}^{d'}$ with respective $q \in \mathcal{P}'(\mathcal{X}')$, where $d \neq d'$. $\mathcal{P}(\mathcal{X})$ consists of all Borel probability measures on $\mathcal{X}$, and $\mathcal{P}'(\mathcal{X}')$ consists of all Borel probability measures on $\mathcal{X}'$. Given i.i.d. observations $X_s := \{x_{s1},...,x_{sn_s}\}$, $X_t = \{x_{t1},...,x_{tn_t}\}$ from p and q, respectively, and $Y_s := \{y_{s1},...,y_{sn_s}\}$ are ground-truth labels corresponding to X_s, and $Y'_t := \{y_{t1},...,y_{tn'_t}\}$ are ground-truth labels corresponding to $X'_t = \{x_{t1},...,x_{tn'_t}\}$, where $y_{si}, y_{tj} \in \mathcal{Y} = \{1,2,...,C\}$. $\mathbf{D}_s = (X_s, Y_s)$ is denoted by a source domain and $\mathbf{D}_t = (X_t) \cup (X'_t, Y'_t)$ is denoted by a target domain. We aim to train a classifier with $\mathbf{D}_s$ and $\mathbf{D}_t$ to accurately label each instance i.i.d. drawn from q.*

HeMap works by using spectral embedding to unify different feature spaces across the target and source domains, even when the feature spaces are completely different [Shi et al., 2013]. Manifold alignment derives its mapping by dividing the mapped instances into different categories according to the original observations [Wang and Mahadevan, 2011]. SSKMDA maps the target domain points to similar source domain points by matching the target kernel matrix to a submatrix of the source kernel matrix based on a Hilbert Schmidt Independence Criterion [Xiao and Guo, 2015].

DASH-N is proposed to jointly learn a hierarchy of features combined with transformations that rectify any mismatches between the domains and has been successful in object recognition [Xiao and Guo, 2015]. A discriminative correlation subspace method is proposed to find the optimal discriminative correlation subspace for the source and target domain.

Yan et al. [2018] presents a novel HeSSDA method by exploiting the theory of optimal transport, a powerful tool originally designed for aligning two different distributions. *Progressive alignment* (PA) [Li et al., 2019b] is implemented to learn representations of two heterogeneous domains with an unsupervised algorithm, but it still needs labeled instances from the target domain to train a final classifier which can handle possible negative transfer situations.

Soft transfer network [Yao et al., 2019], which jointly learns a domain-shared classifier and a domain-invariant subspace in an end-to-end manner, for addressing the HeSSDA problem. The proposed method not only aligns the discriminative directions of domains but also matches both the marginal and conditional distributions across domains.

2.3.2.3 Heterogeneous unsupervised domain adaptation

Unsupervised domain adaptation methods based on homogeneous feature spaces have been widely researched. However, HeUDA methods are rarely studied due to two shortcomings of current domain adaptation methods: the feature spaces must be homogeneous, and there must be at least some labeled instances in the target domain (or there must be a parallel set in both domains). The detailed definition of HeUDA is expressed below.

Definition 2.5 (Heterogeneous unsupervised domain adaptation)**.** *Let x_s be multivariate random variables defined on a topological space $\mathcal{X} \subseteq \mathbb{R}^d$ with respective $p \in \mathcal{P}(\mathcal{X})$, and and x_t be multivariate random variables defined on a topological space $\mathcal{X}' \subseteq \mathbb{R}^{d'}$ with respective $q \in \mathcal{P}'(\mathcal{X}')$, where $d \neq d'$. $\mathcal{P}(\mathcal{X})$ consists of all Borel probability measures on $\mathcal{X}$, and $\mathcal{P}'(\mathcal{X}')$ consists of all Borel probability measures on $\mathcal{X}'$. Given i.i.d. observations $X_s := \{x_{s1}, ..., x_{sn_s}\}$, $X_t = \{x_{t1}, ..., x_{tn_t}\}$ from p and q, respectively, and $Y_s := \{y_{s1}, ..., y_{sn_s}\}$ are ground-truth labels corresponding to X_s, where $y_{si} \in \mathcal{Y} = \{1, 2, ..., C\}$. $\mathbf{D}_s = (X_s, Y_s)$ is denoted by a source domain and $\mathbf{D}_t = (X_t)$ is denoted by a target domain. We aim to train a classifier with $\mathbf{D}_s$ and $\mathbf{D}_t$ to accurately label each instance i.i.d. drawn from q.*

The hybrid heterogeneous transfer learning method [Zhou et al., 2014] uses the information of the parallel set of both domains to transfer knowledge across domains. The domain specific feature transfer [Wei et al., 2019] method is designed to address the HeUDA problem when two domains have common features. The *kernel canonical correlation analysis* (KCCA) [Yeh et al., 2014] method was proposed to address HeUDA problems when there are paired instances in

the source and target domains, but KCCA is not valid when paired instances unavailable.

2.3.2.4 Limitations

Although existing HeUDA methods can address HeUDA problem, they still need parallel sets to bridge two heterogeneous domains, i.e., there are very similar instances in both heterogeneous domains, which is not realistic. For example, credit assessment data are confidential and private, and the information of each sample cannot be accessed. Thus, we cannot find similar instances between two credit-assessment domains. Namely, parallel sets (needed by existing HeUDA methods) do not often exist. To the best of our knowledge, little theoretical discussion has taken place in regard to the absence of a parallel set in the HeUDA. This limits HeUDA methods to be used in more scenarios.

CHAPTER 3

DISCREPANCY OF DIVERSE SUBSETS: A NON-PARAMETRIC TWO-SAMPLE TEST FOR LOW-DIMENSION DATA

3.1 Introduction

As introduced in Section 2.2, all function-based tests suffer from a common issue: their performance depends on the kernel width used [Gretton et al., 2012; Lhéritier and Cazals, 2018]. This issue results in that function-based tests are more difficult to detect local density changes than subset-based tests are [Liu et al., 2018a]. The main reason is that statistic of the subset-based test presented in [Liu et al., 2018a] automatically confirms size of each subset and compares two distributions within each subset. Despite this advantage, statistics of subset-based tests are usually discrete functions [Lu et al., 2014; Schilling,

1986], which means such statistics are hard to help solve machine learning problems. In the field of domain adaptation, statistics of subset-based tests cannot be minimized with a gradient descent algorithm, so they cannot be easily used as a regularizer in the loss function to measure the distance between two distributions. In contrast, *maximum mean discrepancy* (MMD) can be used and, therefore, have become one of the most common function-based tests [Rozantsev et al., 2019].

Yet the ability of subset-based tests to detect local density changes in distributions is enticing, and if their other shortcomings could be overcome, many fields might benefit. To this end, we propose a novel subset-based statistic, called *discrepancy of diverse subsets* (DDS), that is continuous, differentiable, and able to automatically confirm the size of each subset. The proposed statistic naturally has advantages of statistics of subset-based tests and is beneficial for solving two important problems: two-sample test problem and unsupervised domain adaptation problem.

DDS is a nonparametric test statistic to compare two multivariate distributions using two samples drawn from the two distributions. It is based on a corollary of the Radon-Nikodym theorem (see the theorem in chapter 4 of [Cohn, 2013]). That is, within any measurable subset of a topological space $\mathcal{X}$, if the integrations of two measurable functions p and q are the same, then $p = q$ almost everywhere (see this corollary in Appendix A.1).

It is difficult to directly apply this corollary to measure the discrepancy between two distributions as a result of two critical issues: 1) finding representative subsets of $\mathcal{X}$ and 2) defining the discrepancy between two distributions within these subsets. Therefore, we propose β-level diverse subsets to address the first

issue and local impact random variables to address the second issue. Each β-level diverse subset is constructed with the help of an amplification function to determine where two distributions should be compared. The amplification function ensures DDS to be a continuous and differentiable statistic. β controls the size of the subsets and is automatically selected using conditional probabilities between observations. Then, DDS is established to measure discrepancy between two distributions by comparing the total impact of two samples on each β-level subset, where the total impact of one sample (e.g., the sample drawn from p) on one subset is the integration of local impact random variables on this subset. The large value of DDS indicates there is a high probability that p and q are not the same within at least one β-level subset.

In this chapter, we prove that, regardless of whether or not $p = q$, this empirical DDS converges to its population value, which indicates that the empirical DDS is a consistent estimator. We also reveal that using the same β to construct these diverse subsets ensures that the convergence rate of the empirical DDS is independent with the dimension of samples. We then prove that, under a certain condition, 1) $\text{DDS}(p,q) = 0$ if and only if $p = q$ and 2) $\text{DDS}(p,q)$ is a semimetric defined on $\mathcal{P}(\mathcal{X})$ that consists of all Borel probability measures on $\mathcal{X}$. These properties guarantee that DDS is able to distinguish between two different distributions without any prior knowledge on the distributions (e.g., pdfs of two distributions).

To evaluate the performance of DDS when comparing two distributions, we tested it with two types of problems: two-sample test and unsupervised domain adaptation. With the two-sample test problem, we investigated the asymptotic null distribution of DDS, which indicates that the empirical DDS has a normal

behavior when the number of observations goes to infinity. The corresponding DDS-based two-sample test ascertains whether two distributions are the same based on this asymptotic null distribution.

With the unsupervised domain adaptation problem, we investigated a DDS-based generalization bound. This generalization bound highlights that the empirical DDS controls generalization performance in unsupervised domain adaptation problems. Consequently, we propose an unsupervised domain adaptation method based on neural networks and the empirical DDS for transferring knowledge between source and target domains when the target domain does not contain any labeled instances.

The main contributions of this chapter are summarized as follows.

1) A new nonparametric statistic, named DDS, is proposed to measure the discrepancy between two distributions based on subsets of the sample space. Compared to the statistics of current subset-based tests, DDS is continuous and differentiable. These properties mean DDS can be used to derive gradients, which is useful for some machine learning problems, e.g., domain adaptation problems.

2) DDS is more capable of capturing local density discrepancy between two distributions than function-based test statistics. This nature of DDS makes it perform better than common statistics for solving two important problems: two-sample test and unsupervised domain adaptation.

3) The following properties of DDS are proved: a) its empirical form is a consistent estimator and b) it is semimetric in a space consisting only of Borel probability measures.

4) Asymptotic behavior of the empirical DDS is proved, which provides a solid foundation for DDS-based two-sample test. A DDS-based generalization bound

is proved, which highlights that generalization performance in unsupervised domain adaptation problems is controlled by the empirical DDS.

3.2 Problem Setting

We first restate the two-sample test problem as follows.

Problem 1. Let x_1 and x_2 be multivariate random variables defined on a topological space $\mathcal{X} \subseteq \mathbb{R}^d$, with respective p and $q \in \mathcal{P}(\mathcal{X})$, where $x_1 : \mathcal{X} \to \mathcal{X}$, $x_2 : \mathcal{X} \to \mathcal{X}$, $\mathcal{P}(\mathcal{X})$ consists of all Borel probability measures on $\mathcal{X}$. Given i.i.d. observations $X_1 := \{x_{11}, ..., x_{1n_1}\}$ and $X_2 := \{x_{21}, ..., x_{2n_2}\}$ from p and q, respectively, can we decide whether $p \neq q$ through X_1 and X_2?

3.3 Discrepancy of Diverse Subsets

This section presents the definition of DDS and the empirical form of DDS. We prove that empirical DDS is a consistent estimator of DDS and DDS is a semimetric in a space consisting of Borel probability measures.

3.3.1 Population Form of DDS

This subsection presents how to construct representative subsets of $\mathcal{X}$ and how to evaluate the discrepancy between two distributions within each subset. In the following, we use $\mathbb{E}_{x_1 \sim p}(x_1)$ and $\sigma_{x_1 \sim p}(x_1)$ to denote the expectation and standard deviation with respect to p, and $\mathbb{E}_{x_2 \sim q}(x_2)$ and $\sigma_{x_2 \sim q}(x_2)$ to denote the expectation and standard deviation with respect to q. We denote x_1' as an independent copy of x_1 with the distribution p, and x_2' as an independent copy of x_2 with the

distribution q. Given $\omega \in \mathcal{X}$, although there are only four random variables, x_1, x_1', x_2 and x_2', we have many observations drawn from them, i.e., $x_1(\omega)$, $x_1'(\omega)$, $x_2(\omega)$ and $x_2'(\omega)$. In following, we use x_1, x_1', x_2 and x_2' to represent the images of random variables $x_1(\omega)$, $x_1'(\omega)$, $x_2(\omega)$ and $x_2'(\omega)$ for short. Four random variables, called distance random variables, are defined as follows.

$$D_{x_1}(x_1') = \|\phi(x_1) - \phi(x_1')\|^2, D_{x_1}(x_2') = \|\phi(x_1) - \phi(x_2')\|^2, \tag{3.1}$$

$$D_{x_2}(x_1') = \|\phi(x_2) - \phi(x_1')\|^2, D_{x_2}(x_2') = \|\phi(x_2) - \phi(x_2')\|^2, \tag{3.2}$$

where $\phi(\cdot)$ is the function belonging to the reproducing kernel Hilbert space. Below, we introduce the random variable *radius* with respect to x_1 and x_2.

$$r_{x_1} = \frac{\beta}{2}(\sigma_{x_1' \sim p}(D_{x_1}(x_1')) + \sigma_{x_2' \sim q}(D_{x_1}(x_2'))), \tag{3.3}$$

$$r_{x_2} = \frac{\beta}{2}(\sigma_{x_1' \sim p}(D_{x_2}(x_1')) + \sigma_{x_2' \sim q}(D_{x_2}(x_2'))), \tag{3.4}$$

where β is a positive real number and controls the values of r_{x_1} and r_{x_2}. r_{x_1}/β shows the deviation of the distance between x_1 and other observations. To decide whether the values of x_1' or x_2' belong to the same subset, we introduce the amplification function f as follows.

$$f(D, r) = e^{-(\frac{D}{r})^L} \in [0, 1], \tag{3.5}$$

where D corresponds to the images of the distance random variables defined in Eq. (3.1) and Eq. (3.2) and r corresponds to the images of r_{x_1} and r_{x_2} defined in Eq. (3.3) and Eq. (3.4). L is a large integer that amplifies the divergence between D over r and D below r. If L is set to a large number, $f(D, r)$ will reach 0 when $D > r$ and reach 1 when $D < r$.

Next, the indicator random variables are introduced, which show the proximity of x_1 to x_1' (or to x_2') as follows.

$$I_{x_1}(x_1') = f(D_{x_1}(x_1'), r_{x_1}), I_{x_1}(x_2') = f(D_{x_1}(x_2'), r_{x_1}),$$
$$I_{x_2}(x_1') = f(D_{x_2}(x_1'), r_{x_2}), I_{x_2}(x_2') = f(D_{x_2}(x_2'), r_{x_2}).$$

If the images of $I_{x_1}(x_1')$ and $I_{x_1}(x_2')$ are close to 1, then x_1' and x_2' are close to x_1. According to these indicator random variables, the β-level subsets of $\mathcal{X}$ are constructed as follows.

$$S(\mathcal{X}) = \{S_{x_1}, S_{x_2} |\ \forall x_1, x_2 \in \mathcal{X}\},$$

where

$$S_{x_1} = \{x_1' | I_{x_1}(x_1') > \epsilon_0\} \cup \{x_2' | I_{x_1}(x_2') > \epsilon_0\},$$
$$S_{x_2} = \{x_1' | I_{x_2}(x_1') > \epsilon_0\} \cup \{x_2' | I_{x_2}(x_2') > \epsilon_0\},$$

and ϵ_0 is a very small positive real number (we set $\epsilon_0 = 10^{-4}$ in this chapter). Each x_1 (or x_2) can form a subset S_{x_1} containing x_1' and x_2'. If $I_{x_1}(x_1')$ and $I_{x_1}(x_2')$ are over ϵ_0, then x_1' and x_2' have an impact on S_{x_1} and are contained in S_{x_1}. If $I_{x_1}(x_1')$ is equal to $I_{x_1}(x_2')$, then x_1' and x_2' have the same impact on S_{x_1}. The construction of each subset is based on the same β but the size varies according to the distance between observations.

After constructing these subsets of $\mathcal{X}$, we then consider how to evaluate the discrepancy between x_1' and x_2' within one subset. Although we could use the indicator random variables to show whether x_1' and x_2' have the same impact on S_{x_1}, they ignore the impact that x_1' or x_2' has on other subsets. If x_1' or x_2' belongs to several subsets, its local impact on S_{x_1} should be equally divided. For example,

if x'_1 appears in three subsets, its local impact on each subset is 1/3. Motivated by this, we define the local impact random variables to show how many impacts x'_1 and x'_2 have on subsets S_{x_1} and S_{x_2}.

$$L_{x_1}(x'_1) = \frac{I_{x_1}(x'_1)}{V_{x'_1}}, \quad L_{x_1}(x'_2) = \frac{I_{x_1}(x'_2)}{V_{x'_2}}, \tag{3.6}$$

$$L_{x_2}(x'_1) = \frac{I_{x_2}(x'_1)}{V_{x'_1}}, \quad L_{x_2}(x'_2) = \frac{I_{x_2}(x'_2)}{V_{x'_2}}, \tag{3.7}$$

where

$$V_{x'_1} = \gamma\left(\int I_{x_1}(x'_1)dp_{x_1} + \int I_{x_2}(x'_1)dq_{x_2}\right),$$
$$V_{x'_2} = \gamma\left(\int I_{x_1}(x'_2)dp_{x_1} + \int I_{x_2}(x'_2)dq_{x_2}\right).$$

Here, $dp_{x_1} = p(x_1)dx_1$ and $\int I_{x_1}(x'_1)dp_{x_1}$ is the volume of $x_1 \in \{x_1 | I_{x_1}(x'_1) > 0\}$ according to the measure p. γ is the dimension-free factor for ensuring that the convergence rate of the empirical DDS is independent wtih the dimensionality of $\mathcal{X}$ (we show its significance in Remarks 3.1 and 3.2). We set $\gamma = max\{\gamma_{x'_1}, \gamma_{x'_2}\} \geq 1$ where

$$\gamma_{x'_1} = \frac{\max_{x_1}\{\sigma_{x'_1\sim p}(D_{x_1}(x'_1)), \sigma_{x'_2\sim q}(D_{x_1}(x'_2))\}}{\min_{x_1,x_2}\{\sigma_{x'_1\sim p}(D_{x_1}(x'_1)), \sigma_{x'_1\sim p}(D_{x_2}(x'_1))\}} \geq 1,$$
$$\gamma_{x'_2} = \frac{\max_{x_2}\{\sigma_{x'_2\sim q}(D_{x_2}(x'_2)), \sigma_{x'_1\sim p}(D_{x_2}(x'_1))\}}{\min_{x_1,x_2}\{\sigma_{x'_2\sim q}(D_{x_2}(x'_2)), \sigma_{x'_2\sim q}(D_{x_1}(x'_2))\}} \geq 1.$$

Using local impact random variables, DDS with respect to p (DDS_p) and DDS with respect to q (DDS_q) are proposed as follows.

$$\mathrm{DDS}_p(p,q) = \mathbb{E}_{x_1\sim p}\left(\left|\int L_{x_1}(x'_1)dp_{x'_1} - \int L_{x_1}(x'_2)dq_{x'_2}\right|\right),$$
$$\mathrm{DDS}_q(p,q) = \mathbb{E}_{x_2\sim q}\left(\left|\int L_{x_2}(x'_1)dp_{x'_1} - \int L_{x_2}(x'_2)dq_{x'_2}\right|\right).$$

$\int L_{x_1}(x'_1)dp_{x'_1}$ is the total local impact x'_1 has on the subset S_{x_1} (due to ϵ_0 is very small, so we use $\int L_{x_1}(x'_1)dp_{x'_1}$ to approximate $\int_{S_{x_1}} L_{x_1}(x'_1)dp_{x'_1}$), and $\int L_{x_1}(x'_2)dq_{x'_2}$ is the total local impact x'_2 has on the subset S_{x_1}. If $\int L_{x_1}(x'_1)dp_{x'_1} = \int L_{x_1}(x'_2)dq_{x'_2}$, x'_1 and x'_2 have the same total local impact on S_{x_1}, which indicates that there is no discrepancy between x'_1 and x'_2 within this subset. The final statistic is the sum of DDS_p and DDS_q.

$$\mathrm{DDS}(p,q) = \mathrm{DDS}_p(p,q) + \mathrm{DDS}_q(p,q). \tag{3.8}$$

Using notations from measure theory, DDS is expressed as follows.

$$\mathrm{DDS}(p,q) = \int \left| \int L_x(x')d(p_{x'} - q_{x'}) \right| d(p_x + q_x). \tag{3.9}$$

In Eq. (3.9), $\int L_x(x')dq_{x'}$ is equal to $\int L_x(x'_2)dq_{x'_2}$ of Eq. (3.8). When $p = q$, $\mathrm{DDS}(p,q)$ is equal to 0. Based on the definitions of the indicator random variables and the local impact random variables, DDS is sensitive to local-area changes and global-area changes. If $I_{x_1}(x'_1) = 1$ and $I_{x_1}(x'_2) = 0$ (local-area changes), we reach the conclusion that x'_1 and x'_2 are different on S_{x_1}. However, if $\int I_{x_1}(x'_1)dp_{x'_1} = \int I_{x_1}(x'_2)dq_{x'_2}$ but $V_{x'_1} \neq V_{x'_2}$ (global-area changes), we reach the conclusion that x'_1 and x'_2 are different on other subsets (except S_{x_1}). This means that $\mathrm{DDS}(p,q)$ has the capability to detect many differences between p and q.

3.3.2 Empirical DDS

This subsection introduces the empirical form of DDS, $\hat{\mathrm{DDS}}(X_1, X_2)$. $D(x_{1i}, x_{1j})$, $D(x_{1i}, x_{2j})$, $D(x_{2i}, x_{1j})$ and $D(x_{2i}, x_{2j})$ refer to the four random variables defined in Eq. (3.1) and Eq. (3.2) (x_{1i} refers to x_1, x_{1j} refers to x'_1, and likewise for x_{2i}

and x_{2j}). Then, r_{x_1} and r_{x_2} have the following estimated form:

$$\hat{r}_{x_{1i}} = \frac{\beta}{2}(\hat{\sigma}_{x_{1j}}(D(x_{1i}, x_{1j})) + \hat{\sigma}_{x_{2j}}(D(x_{1i}, x_{2j}))),$$

$$\hat{r}_{x_{2i}} = \frac{\beta}{2}(\hat{\sigma}_{x_{1j}}(D(x_{2i}, x_{1j})) + \hat{\sigma}_{x_{2j}}(D(x_{2i}, x_{2j}))),$$

where $\hat{\sigma}_{x_{1j}}(D(x_{1i}, x_{1j}))$ is the unbiased empirical estimator of the standard deviation of $D_{x_1}(x_1')$ with respect to x_1'. Using the amplification function f, $I(x_{1i}, x_{1j})$, $I(x_{1i}, x_{2j})$, $I(x_{2i}, x_{1j})$ and $I(x_{2i}, x_{2j})$ can be obtained. We now have $n_1 + n_2$ of β-level subsets. Using empirical estimates of standard deviation, $\gamma_{x_{1j}}$ and $\gamma_{x_{2j}}$ can be calculated according to $\gamma_{x_1'}$ and $\gamma_{x_2'}$. Next, $\gamma = \max_{j_1, j_2}\{\gamma_{x_{1,j_1}}, \gamma_{x_{2,j_2}}\}$, where $j_1 = 1, ..., n_1$ and $j_2 = 1, ..., n_2$. The estimators of $V_{x_1'}$ and $V_{x_2'}$ are expressed as follows.

$$\hat{V}_{x_{1j}} = \gamma\left(\frac{1}{n_1}\sum_{i=1}^{n_1} I(x_{1i}, x_{1j}) + \frac{1}{n_2}\sum_{i=1}^{n_2} I(x_{2i}, x_{1j})\right),$$
$$\hat{V}_{x_{2j}} = \gamma\left(\frac{1}{n_1}\sum_{i=1}^{n_1} I(x_{1i}, x_{2j}) + \frac{1}{n_2}\sum_{i=1}^{n_2} I(x_{2i}, x_{2j})\right).$$

$L(x_{1i}, x_{1j}), L(x_{1i}, x_{2j}), L(x_{2i}, x_{1j})$ and $L(x_{2i}, x_{2j})$ can thus be easily computed through Eq. (3.6) and Eq. (3.7). Estimators of DDS_p and DDS_q then are given as follows.

$$\hat{\mathrm{DDS}}_{X_1} = \frac{1}{n_1}\sum_{i=1}^{n_1}\left(\left|\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{1i}, x_{1j}) - \frac{1}{n_2}\sum_{j=1}^{n_2} L(x_{1i}, x_{2j})\right|\right),$$
$$\hat{\mathrm{DDS}}_{X_2} = \frac{1}{n_2}\sum_{i=1}^{n_2}\left(\left|\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{2i}, x_{1j}) - \frac{1}{n_2}\sum_{j=1}^{n_2} L(x_{2i}, x_{2j})\right|\right),$$

thus $\hat{\mathrm{DDS}}(X_1, X_2) = \hat{\mathrm{DDS}}_{X_1} + \hat{\mathrm{DDS}}_{X_2}$. $\hat{\mathrm{DDS}}_{X_1}$ measures the discrepancy between two samples within subsets generated by instances in X_1 and $\hat{\mathrm{DDS}}_{X_2}$ measures the discrepancy between two samples within subsets generated by instances in X_2. The summation of both measures total discrepancy between two samples.

3.3.3 Automatic Selection of Beta

According to r_{x_1} and r_{x_2}, β controls the size of the subsets. If β is too big, all the subsets will be the same; if β is too small, there will be no overlap of any of the subsets, so the intersection between them will be empty, which is not considered in our case. In other words, β is a subset divergence controller for the empirical DDS.

To select the best β value, we want to have the maximum average divergence of the subsets. Intuitively, we consider that the higher the average divergence of the subsets, the greater the sensitivity of the distribution discrepancy that will be captured. For example, if two observations x_{1i}, x_{2i} are always observed together in subsets, it is impossible to capture the density discrepancy between them. Therefore, we consider the average conditional probability of observing a subset $S_{x_{2i}}$ giving a subset $S_{x_{1i}}$ as an indicator to quantify the diversity of subsets. According to the definition of β-level subsets, we have the conditional probability of $S_{x_{1j}}$ given $S_{x_{1i}}$ as follows.

$$P(S_{x_{1j}}|S_{x_{1i}}) = \frac{|S_{x_{1j}} \cap S_{x_{1i}}|}{|S_{x_{1i}}|},$$

where $|\{\cdot\}|$ is the cardinality of a set $\{\cdot\}$. We can compute $P(S_{x_{2j}}|S_{x_{1i}})$, $P(S_{x_{1j}}|S_{x_{2i}})$ and $P(S_{x_{2j}}|S_{x_{2i}})$ in a similar way. The diversity factor of $S_{x_{1i}}$ in terms of β-level subsets can then be defined as:

$$\mathcal{D}_\beta(S_{x_{1i}}) = 1 - \frac{1}{2}\Big(\frac{1}{n'_1(x_{1i})}\sum_{j=1}^{n_1} P(S_{x_{1j}}|S_{x_{1i}}) + \frac{1}{n'_2(x_{1i})}\sum_{j=1}^{n_2} P(S_{x_{2j}}|S_{x_{1i}})\Big),$$

where $n'_1(x_{1i}) = \sum_j sign(P(S_{x_{1j}}|S_{x_{1i}}) > 0)$ and $n'_2(x_{1i}) = \sum_j sign(P(S_{x_{2j}}|S_{x_{1i}}) > 0)$.

Algorithm 3.1 Beta Selection

1: Input: The amplified distance matrix, M_Z^I, β grid search range (default $\mathcal{B} = [0.02, 20]$ with 1000 values)
2: Initialize $\beta^{\max} = -1$, $\mathcal{D}_\beta^{\max} = 0$;
for β *in* $\mathcal{B}$ **do**
    3: Compute $\mathcal{D}_\beta$ with β and M_Z^I;
    if $\mathcal{D}_\beta > \mathcal{D}_\beta^{\max}$ **then**
        4: Assign $\beta^{\max} = \beta$;
        5: Assign $\mathcal{D}_\beta^{\max} = \mathcal{D}_\beta$;
    end
end
6: Output: $\beta_{\max}$.

The average diversity of the β-level subsets is defined as follows.

$$\mathcal{D}_\beta(S(\mathcal{X})) = n_1^{-1} \sum_{i=1}^{n_1} \mathcal{D}_\beta(S_{x_{1i}}) + n_2^{-1} \sum_{i=1}^{n_2} \mathcal{D}_\beta(S_{x_{2i}}). \tag{3.10}$$

Therefore, the value of β is selected via $\max_\beta\{\mathcal{D}_\beta(S(\mathcal{X}))\}$. β-level diverse subsets of $\mathcal{X}$ are constructed using the best β. Algorithm 3.1 presents the implementation details of how to select the β for calculating the DDS statistics for a given amplified distance matrix M_Z^I. Line 2 initialize the $\beta^{\max}$ and $\mathcal{D}_\beta^{\max}$. Line 3-5 for loop the predefined $\mathcal{B}$ grid search range to find the $\beta^{\max}$ so that $\mathcal{D}_\beta$ reaches the maximum. The $\mathcal{D}_\beta$ is computed by Eq (3.10). At last, line 6 return the best β value according to the grid search.

3.3.4 Consistency of the empirical DDS

This subsection shows that regardless of whether $p = q$, the empirical DDS is a consistent estimator. First, let

$$g(x_1) = \left| \int L_{x_1}(x_1') dp_{x_1'} - \int L_{x_1}(x_2') dq_{x_2'} \right|,$$

and we give the upper bound of $g(x_1)$ as follows.

Lemma 3.1. *Given p, q, X_1, X_2 defined in Problem 1 and assuming $\sigma_{x_1' \sim p}(D_{x_1}(x_1'))$, $\sigma_{x_1' \sim p}(D_{x_2}(x_1'))$ and $\sigma_{x_2' \sim q}(D_{x_1}(x_2'))$ are finite, the upper bound of $g(x_1)$ is expressed as*

$$g(x_1) < \left(\frac{\beta \max_{x_1}\{\sigma_{x_1' \sim p}(D_{x_1}(x_1')), \sigma_{x_2' \sim q}(D_{x_1}(x_2'))\}}{\beta\gamma \min\{\sigma_{x_1' \sim p}(D_{x_1}(x_1')), \sigma_{x_1' \sim q}(D_{x_2}(x_1'))\}} \right)^d = M_{pq}/\gamma^d,$$

where d is the dimension of x_1 and $M_{pq} = \max\{\|p\|_\infty, \|q\|_\infty\}$.

Proof. Without loss of generality, we assume that $\int L_{x_1}(x_1')dp_{x_1'} - \int L_{x_1}(x_2')dq_{x_2'} > 0$. When x_2' is not in any subset containing x_1, $g(x_1)$ has an upper bound (since $L_{x_1}(x_1') > 0$),

$$g(x_1) < \int L_{x_1}(x_1')dp_{x_1'} < M_{pq} \int L_{x_1}(x_1')dx_1',$$

where $M_{pq} = \max\{\|p\|_\infty, \|q\|_\infty\}$. Recall the definition of $V_{x_1'}$, $V_{x_1'}$ means the volume consists of x_1 and x_2, where both S_{x_1} and S_{x_2} contain x_1'. In terms of r_{x_1} and r_{x_2}, we know

$$V_{x_1'} > \frac{\pi^{\frac{d}{2}} \left(\beta\gamma \min_{x_1,x_2}\{\sigma_{x_1' \sim p}(D_{x_1}(x_1')), \sigma_{x_1' \sim p}(D_{x_2}(x_1'))\} \right)^d}{\Gamma\left(\frac{d}{2}+1\right)}. \tag{3.11}$$

Inequality (3.11) is based on the d-dimensional volume of a Euclidean ball. We denote the right part of Inequality (3.11) as V_{min} and we have

$$g(x_1) < M_{pq} \int_{I_{x_1}(x_1')>0} \frac{1}{V_{min}} dx_1'.$$

Then, we need to estimate the maximum volume consisting of x_1', where x_1' is included in S_{x_1}. In terms of r_{x_1}, we know

$$\frac{g(x_1)}{M_{pq}} < \frac{\pi^{\frac{d}{2}} \left(\beta \max_{x_1}\{\sigma_{x_1' \sim p}(D_{x_1}(x_1')), \sigma_{x_2' \sim q}(D_{x_1}(x_2'))\} \right)^d}{\Gamma\left(\frac{d}{2}+1\right) V_{min}}.$$

Thus, based on the definition of γ, $g(x_1) < M_{pq}/\gamma^d$. ■

Remark 3.1. *Lemma 3.1 indicates that β and γ are significant in keeping convergence rate of the empirical DDS benefit from the dimensionality of x_1 (shown in Theorem 3.1). If β is not the same for different x_1, $g(x_1)$ may have a infinite upper bound when increasing the dimensionality of observations. Similarly, if we do not introduce γ, the upper bound of $g(x_1)$ approaches infinity when the dimensionality of observations increases.*

The uniform convergence bound for the empirical DDS is given in the following theorem.

Theorem 3.1. *Given p, q, X_1, X_2 defined in Problem 1 and assuming that values of $\sigma_{x_1' \sim p}(D_{x_1}(x_1'))$, $\sigma_{x_1' \sim p}(D_{x_2}(x_1'))$, $\sigma_{x_2' \sim q}(D_{x_1}(x_2'))$ and $\sigma_{x_2' \sim q}(D_{x_2}(x_2'))$ are finite, then $\forall \epsilon > 0$ we have*

$$Pr_{X_1,X_2}\{|DDS(p,q) - D\hat{D}S(X_1,X_2)| - \mathcal{O}(n^{-\frac{1}{2}}) > \epsilon\} \leq 2exp\left(\frac{-\epsilon^2 \gamma^{2d} n_1 n_2}{2M_{pq}^2(n_1+n_2)}\right),$$

where Pr_{X_1,X_2} denotes the probability over the n_1-sample X_1 and n_2-sample X_2, and $M_{pq} = \max\{\|p\|_\infty, \|q\|_\infty\}$.

The completed proof is provided in the Appendix A.1. The basic idea is to use McDiarmid's inequality and Lemma 3.1.

Remark 3.2. *Theorem 3.1 shows that the empirical DDS approaches its population value at rate $\mathcal{O}(M_{pq}(n_1^{-1} + n_2^{-1})^{1/2}/\gamma^d)$. This rate benefits from the dimensionality of x_1 and x_2.*

3.3.5 DDS as a Semimetric in P(X)

This subsection presents two important properties of DDS(p,q): when $L \to +\infty$, 1) DDS(p,q) = 0, if and only if $p = q$; and 2) it can be proved that DDS(p,q) is

a semimetric defined on $\mathcal{P}(\mathcal{X})$, where L is the parameter of the amplification function defined in Eq. (3.5). Formally, we have following theorems.

Theorem 3.2. *Given $p, q \in \mathcal{P}(\mathcal{X})$ and $L \to +\infty$, then $p = q$ if and only if DDS(p, q) = 0.*

The completed proof is provided in the Appendix A.1. The basic idea is to use properties of indicator functions. Theorem 3.2 shows that DDS(p, q) is a fine measurement for two Borel probability measures. If the DDS(p, q) is close to 0, p and q are also probably close to each other. At the end of this subsection, we show that DDS(p, q) is a semimetric defined on $\mathcal{P}(\mathcal{X})$.

Theorem 3.3. *Given $p, q \in \mathcal{P}(\mathcal{X})$ and $L \to +\infty$, DDS(p, q) is a semimetric defined on $\mathcal{P}(\mathcal{X})$.*

Proof. $\forall p, q \in \mathcal{P}(\mathcal{X})$, we know 1) DDS$(p, q) \geq 0$; 2) DDS$(p, q)$ = DDS(q, p) and 3) DDS$(p, q) = 0$ if and only if $p = q$. This means that DDS(p, q) is a semimetric defined on $\mathcal{P}(\mathcal{X})$. ■

3.4 DDS for Two-sample Test

This section introduces how to apply DDS to decide whether two samples come from the same distribution. To obtain a DDS based two-sample test, we study the asymptotic behavior of DDS in the following subsection.

3.4.1 Asymptotic Distribution of DDS

Asymptotic behavior is an important property of the test statistics, because it precisely demonstrates the behavior of a statistic when obtaining infinite observa-

tions. For example, MMD has a Normal asymptotic behavior [Gretton et al., 2012], while ME has a chi-square asymptotic behavior [Jitkrittum et al., 2016]. This subsection shows that the permutation null distribution of $\hat{\text{DDS}}(X_1, X_2)$ approaches a normal distribution when $n_1 \to \infty$, $n_2 \to \infty$ and $n_1/(n_1+n_2) \to \lambda_1 \in (0,1)$. Before stating the theorem, we define a new random variable z with observations z_i in which

$$z_i = x_{1i}, i = 1,...,n_1, \quad z_{j+n_1} = x_{2j}, j = 1,...,n_2. \tag{3.12}$$

Let $n = n_1 + n_2$. In the null hypothesis test, we assume that $p = q$. The estimates of r_{x_1} and r_{x_2} are thus expressed as follows. $\hat{r}^{\mathcal{H}_0}_{x_{1i}} = \beta\hat{\sigma}_{z_j}(D(x_{1i}, z_j))$ and $\hat{r}^{\mathcal{H}_0}_{x_{2i}} = \beta\hat{\sigma}_{z_j}(D(x_{2i}, z_j))$.

First, we state the following lemma.

Lemma 3.2. *Given X_1 and X_2 defined in Problem 2 and z defined in* (3.12)*, when $n \to \infty$, under the permutation null,*

$$\frac{1}{n_1\sigma_{x_{1i}}}\left(\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{1i}, x_{1j}) - \frac{1}{n_2}\sum_{j=1}^{n_2} L(x_{1i}, x_{2j})\right) \xrightarrow{\mathcal{D}} \mathcal{N}(0,1),$$

where $\lambda_1 = \lim_{n\to\infty} n_1/n$, $\lambda_2 = \lim_{n\to\infty} n_2/n$, $\sigma^2_{x_{1i}} = n_1^{-2}(\lambda_1 n_1^{-2} + \lambda_2 n_2^{-2})\sum_{j=1}^{n} L(x_{1i}, z_j)^2$ and $\mathcal{D}$ is "converges in distribution".

The completed proof is provided in the Appendix A.1. To prove this lemma, the basic idea is to construct a new random variable as follows.

$$P(b_j = n_1^{-1}L(x_{1i}, z_j)) = \lambda_1, P(b_j = -n_2^{-1}L(x_{1i}, z_j)) = \lambda_2,$$

where P means the probability and $\lambda_1 + \lambda_2 = 1$. Under the permutation null ($p = q$), we know

$$n_1^{-1}\sum_{j=1}^{n_1} L(x_{1i}, x_{1j}) - n_2^{-1}\sum_{j=1}^{n_2} L(x_{1i}, x_{2j}) = \sum_{j=1}^{n} b_j.$$

Applying Lyapunov's central limit theorem to $\sum_{j=1}^{n} b_j$, this theorem is proved.

According to Lemma 3.1, we know $n^{-2}\sum_j L(x_{1i}, z_j)^2 \propto \mathcal{O}(1/n)$, which means that $\sigma_{x_{1i}}$ converges to zero at a rate of $\mathcal{O}\big(1/\sqrt{\lambda_1^{-1} n_1^{-1}(\lambda_1 n_1^{-2} + \lambda_2 n_2^{-2})}\big)$. Next, we state the main theorem for this section.

Theorem 3.4. *Given X_1 and X_2 defined in Problem 2, when $n \to \infty$, under the permutation null,*

$$\frac{n_1^{\frac{1}{2}} D\hat{D}S_{X_1} - \frac{\sqrt{2}}{\sqrt{\pi}} \sum_{i=1}^{n_1} \sigma_{x_{1i}}}{\sqrt{\big(1 - \frac{2}{\pi}\big) \sum_{i=1}^{n_1} \sigma_{x_{1i}}^2}} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1),$$

where $\sigma_{x_{1i}}^2 = n_1^{-2}(\lambda_1 n_1^{-2} + \lambda_2 n_2^{-2}) \sum_{j=1}^{n} L(x_{1i}, z_j)^2$ and $\mathcal{D}$ is "converges in distribution".

The completed proof is provided in the Appendix A.1. To prove this theorem, the basic idea is to construct a new random variable $a_{x_{1i}}$ as per the following.

$$a_{x_{1i}} = \frac{1}{n_1}\Big(\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{1i}, x_{1j}) - \frac{1}{n_2}\sum_{j=1}^{n_2} L(x_{1i}, x_{2j})\Big).$$

So, we know $\hat{\mathrm{DDS}}_{X_1} = \sum_i |a_{x_{1i}}|$. Applying Lyapunov's central limit theorem to $\sum_i |a_{x_{1i}}|$, this theorem is proved.

The following corollaries are obtained according to Theorem 3.4.

Corollary 3.1. *Given X_1 and X_2 defined in Problem 2 and z defined in* (3.12)*, when $n \to \infty$, under the permutation null,*

$$\frac{n_2^{\frac{1}{2}} D\hat{D}S_{X_2} - \frac{\sqrt{2}}{\sqrt{\pi}} \sum_{i=1}^{n_2} \sigma_{x_{2i}}}{\sqrt{\big(1 - \frac{2}{\pi}\big) \sum_{i=1}^{n_2} \sigma_{x_{2i}}^2}} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1),$$

where $\sigma_{x_{2i}}^2 = n_2^{-2}(\lambda_1 n_1^{-2} + \lambda_2 n_2^{-2}) \sum_{j=1}^{n} L(x_{2i}, z_j)^2$.

Corollary 3.2. *Given X_1 and X_2 defined in Problem 2 and z defined in* (3.12)*, when $n \to \infty$, under the permutation null,*

$$\frac{n^{\frac{1}{2}} D\hat{D}S - \mu}{\sigma} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1), \tag{3.13}$$

where

$$\mu = \frac{\sqrt{2}}{\sqrt{\pi}}\Big(\lambda_1^{-\frac{1}{2}} \sum_{i=1}^{n_1} \sigma_{x_{1i}} + \lambda_2^{-\frac{1}{2}} \sum_{i=1}^{n_2} \sigma_{x_{2i}}\Big),$$
$$\sigma^2 = \Big(1 - \frac{2}{\pi}\Big)\Big(\lambda_1^{-1} \sum_{i=1}^{n_1} \sigma_{x_{1i}}^2 + \lambda_2^{-1} \sum_{i=1}^{n_2} \sigma_{x_{2i}}^2\Big),$$

$\sigma_{x_{1i}}$ is defined in Lemma 3.2 and $\sigma_{x_{2i}}$ is defined in Corollary 3.1.

From Corollary 3.2, we know that empirical DDS has Normal behavior when obtaining infinite observations.

3.4.2 Two Hypothesis Tests based on DDS

This subsection introduces two hypothesis tests based on DDS. The first one is based on Theorem 3.1 and the second one is based on Corollary 3.2.

Test I. A consistent statistical test is obtained by using the uniform convergence bound in Theorem 3.1,

Corollary 3.3. *A hypothesis test of level α for the null hypothesis $\mathcal{H}_0 : p = q$ has the acceptance region $D\hat{D}S(X_1, X_2) < \hat{M}_{pq} \gamma^{-d} \sqrt{(2n_1^{-1} + 2n_2^{-1}) ln(\alpha^{-1})}$.*

The hypothesis testing in Corollary 3.3 is easily computed ($O(1)$) but M_{pq} is hard to estimate. The another drawback of Test I is that it does not involve the asymptotic behavior of $\hat{\text{DDS}}(X_1, X_2)$.

Test II. Corollary 3.4 gives the second hypothesis test based on the asymptotic distribution of $\hat{\text{DDS}}(X_1, X_2)$.

Corollary 3.4. *A hypothesis test of level α for the null hypothesis $\mathcal{H}_0 : p = q$ has the acceptance region $D\hat{D}S(X_1, X_2) < n^{-\frac{1}{2}}(\mu + \sigma\sqrt{2}erf^{-1}(1-2\alpha))$, where $erf(\cdot)$ is the error function in statistics, μ and σ^2 are defined in Formula* (3.13)*.*

According to convergence rates of $\sigma_{x_{1i}}$ and $\sigma_{x_{2i}}$, we know that the expectation value and the standard deviation of $\hat{\text{DDS}}(X_1, X_2)$ converge to zero at a rate of $\mathcal{O}(2\sqrt{2}c/\sqrt{\lambda_1\lambda_2 n\pi})$ and $\mathcal{O}(C/n)$, respectively (assuming $\lambda_1 < \lambda_2$), where c is a constant depending on how rapidly $n^{-2}L(x_{1i}, z_j)^2$ or $n^{-2}L(x_{2i}, z_j)^2$ converge to zero and $C = c\sqrt{(1-2/\pi)(\lambda_1^{-1}\lambda_2^{-2} + \lambda_1^{-2}\lambda_2^{-1})}$. In experiments, we show that sample sizes in the hundreds are large enough to use the approximated acceptance threshold based on the asymptotic distribution.

3.5 DDS for Unsupervised Domain Adaptation

This section introduces how to apply DDS to help transfer knowledge from a source domain, which has sufficient labeled instances, to a target domain, which has no labeled instance. We restate the unsupervised domain adaptation problem first:

Problem 2. Let x_s and x_t be multivariate random variables defined on a topological space $\mathcal{X} \subseteq \mathbb{R}^d$, with respective p and $q \in \mathcal{P}(\mathcal{X})$, where $x_s : \mathcal{X} \to \mathcal{X}$, $x_t : \mathcal{X} \to \mathcal{X}$, $\mathcal{P}(\mathcal{X})$ consists of all Borel probability measures on $\mathcal{X}$. Given i.i.d. observations $X_s := \{x_{s1}, ..., x_{sn_s}\}$, $X_t = \{x_{t1}, ..., x_{tn_t}\}$ from p and q, respectively and $Y_s := \{y_{s1}, ..., y_{sn_s}\}$ are ground-truth labels corresponding to X_s, where $y_{si} \in \mathcal{Y} = \{1, 2, ..., C\}$. $\mathbf{D}_s = (X_s, Y_s)$ is denoted by a source domain and $\mathbf{D}_t = (X_t)$ is denoted by a target domain. Can we use knowledge from $\mathbf{D}_s$ to label each instance x_{ti} in $\mathbf{D}_t$?

We begin by studying the generalization bound for DDS based unsupervised domain adaptation and then show how to use DDS to transfer knowledge across two domains.

3.5.1 Generalization Bound for DDS based Unsupervised Domain Adaption

We denote the labeling functions of the source and target domains as by $f_p(x_s)$ and $f_q(x_t)$, respectively. It is clear that the dissimilarity between $f_p(x_s)$ and $f_q(x_t)$ need to be small for adaptation to be possible.

We denote $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ a loss function defined over pairs of labels and $\mathcal{L}_p(f,g)$ the expected loss for any two functions $f,\ g : X_s \to \mathcal{Y}$, which can be expressed by $\mathcal{L}_p(f,g) = \mathbb{E}_{x_s \sim p}(\ell(f(x_s), g(x_s)))$. The domain adaptation problem consists of selecting a hypothesis h out of a hypothesis set H with a small expected loss according to the distribution q, $\mathcal{L}_q(h,f)$ [Mansour et al., 2009]. Let $h_p^* = \text{argmin}_{h \in H} \mathcal{L}_p(h, f_p)$, $h_q^* = \text{argmin}_{h \in H} \mathcal{L}_q(h, f_q)$ and we assume the loss function ℓ is symmetric and obeys the triangle inequality. Then, we have, $\forall h \in H$

$$\begin{aligned} \mathcal{L}_q(h, f_q) &\leq \mathcal{L}_q(h, h_p^*) + \mathcal{L}_q(h_p^*, h_q^*) + \mathcal{L}_q(h_q^*, f_q) \\ &\leq \mathcal{L}_q(h_q^*, f_q) + |\mathcal{L}_q(h, h_p^*) - \mathcal{L}_p(h, h_p^*)| + \mathcal{L}_p(h, h_p^*) + \mathcal{L}_q(h_p^*, h_q^*). \end{aligned} \tag{3.14}$$

In Inequality (3.14), $|\mathcal{L}_q(h, h_p^*) - \mathcal{L}_p(h, h_p^*)|$ represents the discrepancy between p and q if we give h and h_p^*, which can be bounded by DDS if the following assumption exists.

Assumption (discrepancy assumption). If $p \neq q$, then $\forall x, x' \in \{x | p(x) \neq q(x), x \in \mathcal{X}\}$, $\exists c > 1\ s.t.\ \gamma | \int_{S_x} L_x(x') dp_{x'} - \int_{S_x} L_x(x') dq_{x'} | \geq 1/c > 0$, where c is a finite real

number and $S_x \subset \mathcal{X}$ is a subset derived by x.

If $p \neq q$, then we have $\mathrm{DDS}(p,q) \neq 0$, meaning that there should be some subsets in which two samples have different local impacts. This assumption indicates: if $p \neq q$, then there is a discrepancy between $\int_{S_x} L_x(x')dp_{x'}$ and $\int_{S_x} L_x(x')dq_{x'}$ within every subset of the set $\{x|p(x) \neq q(x), x \in \mathcal{X}\}$ (the support set of $p(x)-q(x)$. We could decrease β to satisfy this assumption. In the empirical DDS, the discrepancy assumption means that $\forall x_{1i}$,

$$\gamma|n_1^{-1}\sum_{j=1}^{n_1} L(x_{1i},x_{1j}) - n_2^{-1}\sum_{j=1}^{n_2} L(x_{1i},x_{2j})| \geq 1/c > 0.$$

Let $U = \{x|\ \forall x' \in S_x,\ p(x') > q(x')\} \cup \{x|\ \forall x' \in S_x,\ p(x') < q(x')\}$ and $\mathcal{X}_{Supp} = \{x|p(x)-q(x) \neq 0, x \in \mathcal{X}\}$. If ℓ is bounded by a finite real number $M > 0$, based on the definition of $\mathcal{L}_q(h,h_p^*)$ and the discrepancy assumption, we have

$$\begin{aligned}
|\mathcal{L}_q(h,h_p^*) - \mathcal{L}_p(h,h_p^*)| &= |\mathbb{E}_{x_s\sim p}(\ell(h(x_s),h_p^*(x_s))) - \mathbb{E}_{x_t\sim q}(\ell(h(x_t),h_p^*(x_t)))| \\
&= \int \ell(h(x'),h_p^*(x'))d(p_{x'}-q_{x'}) \\
&= \int \frac{\int I_{x_1}(x')d(p_x+q_x)}{\int I_{x_1}(x')d(p_x+q_x)}\ell(h(x'),h_p^*(x'))d(p_{x'}-q_{x'}) \\
&= \gamma\int\Big(\int L_x(x')\ell(h(x'),h_p^*(x'))d(p_{x'}-q_{x'})\Big)d(p_x+q_x) \\
&\leq \gamma\int\Big|\int L_x(x')\ell(h(x'),h_p^*(x'))d(p_{x'}-q_{x'})\Big|d(p_x+q_x) \\
&\leq M\gamma\int_U\Big|\int L_x(x')d(p_{x'}-q_{x'})\Big|d(p_x+q_x) \\
&\quad + Mc\gamma\int_{\mathcal{X}_{Supp}/U}\Big|\int L_x(x')d(p_{x'}-q_{x'})\Big|d(p_x+q_x) \\
&= M\gamma((c-1)\theta+1)\cdot\mathrm{DDS}(p,q), \qquad (3.15)
\end{aligned}$$

where $\theta = \int_{\mathcal{X}_{Supp}/U} |\int L_x(x')d(p_{x'} - q_{x'})|d(p_x + q_x)/\text{DDS}(p\ ,q) \in (0,1]$. Thus, we have

(3.16)

$$\mathcal{L}_q(h, f_q) - \mathcal{L}_q(h_q^*, f_q) \leq \mathcal{L}_p(h, h_p^*) + \mathcal{L}_q(h_p^*, h_q^*) + M\gamma((c-1)\theta + 1)\text{DDS}(p,q).$$

Because p and q are continuous functions, smaller β indicates smaller θ and smaller upper bound in Inequality (3.16). Before we stating the generalization bound for DDS based unsupervised domain adaptation, we introduce the Rademacher complexity [Bartlett and Mendelson, 2002] which measures the degree to which a class of functions can fit random noise. This measure is the basis of bounding the empirical loss and expected loss.

Definition 3.1 (Rademacher Complexity)**.** *Let H be a set of real-valued functions defined over $\mathcal{X}$. Given a sample $X = \{x_1, ..., x_m\}$, the empirical Rademacher complexity of H is defined as follows.*

$$\hat{\Re}_X(H) = \frac{2}{m}\mathop{\mathbb{E}}_{\sigma}\Big(\sup_{h\in H}\Big|\sum_{i=1}^{m}\sigma_i h(x_{1i})\Big|\Big).$$

The expectation is taken over $\sigma = (\sigma_1, ..., \sigma_m)$ where σ_is are independent uniform random variables taking values in {-1,+1}. The Rademacher complexity of a hypothesis set H is defined as the expectation of $\hat{\Re}_X(H)$ over all samples of size m:

$$\Re_m(H) = \mathop{\mathbb{E}}_{X}\Big(\hat{\Re}_X(H)\Big||X| = m\Big).$$

Based on Rademacher complexity and Theorem 2 in [Mansour et al., 2009], with probability at least $1-\delta$ over X_s of size n_s drawn from p, upper bound of $\mathcal{L}_p(h, h_p^*)$ can be expressed as follows.

(3.17)
$$\mathcal{L}_p(h, h_p^*) \leq \hat{\mathcal{L}}_p(h, h_p^*) + \hat{\Re}_{X_s}(\ell_H) + 3M\sqrt{\frac{log\frac{2}{\delta}}{2n_s}},$$

where $\ell_H = \{x \to \ell(h(x), h_p^*(x)) | h \in H\}$ and $\hat{\mathcal{L}}_p(h, h_p^*)$ is the empirical loss in the source domain. According to Theorem 3.1 and Inequalities (3.15), (3.16) and (3.17), the generalization bound for DDS based unsupervised domain adaptation is stated as follows.

Theorem 3.5 (Adaptation bound with DDS)**.** *Let H be a family of functions mappings from $\mathcal{X}$ to $\mathbb{R}$. Given X_s, X_t, p, q defined in Problem 3 and loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$, if discrepancy assumption is satisfied and ℓ is symmetric, bounded by a positive finite real number M and obeys the triangle inequality, for any hypothesis $h \in H$, with probability at least $1 - 2\delta$, the following adaptation bound holds:*

$$\begin{aligned}\mathcal{L}_q(h, f_q) - \mathcal{L}_q(h_q^*, f_q) \leq & \hat{\mathcal{L}}_p(h, h_p^*) + \hat{\Re}_{X_s}(\ell_H) + 3M\sqrt{\frac{log\frac{2}{\delta}}{2n_s}} + \mathcal{L}_q(h_p^*, h_q^*) \\ & + M\gamma\rho\left(\mathrm{D\hat{D}S}(X_s, X_t) + \frac{M_{pq}}{\gamma^d}\sqrt{\frac{2(n_s+n_t)log(\frac{2}{\delta})}{n_s n_t}} + \mathcal{O}(n^{-\frac{1}{2}})\right),\end{aligned} \tag{3.18}$$

where $\rho = (c-1)\theta + 1$, c is defined in the discrepancy assumption, θ is defined in Inequality (3.15)*, $\hat{\Re}_{X_s}(\ell_H)$ is defined in Inequality* (3.17) *and $M_{pq} = \max\{\|p\|_\infty, \|q\|_\infty\}$.*

Proof. Recall Inequality (3.14):

$$\begin{aligned}\mathcal{L}_q(h, f_q) \leq & \mathcal{L}_q(h, h_p^*) + \mathcal{L}_q(h_p^*, h_q^*) + \mathcal{L}_q(h_q^*, f_q) \\ \leq & \mathcal{L}_q(h_q^*, f_q) + |\mathcal{L}_q(h, h_p^*) - \mathcal{L}_p(h, h_p^*)| + \mathcal{L}_p(h, h_p^*) + \mathcal{L}_q(h_p^*, h_q^*).\end{aligned}$$

Due to Theorem 3.1, we know

$$\begin{aligned}\mathrm{DDS}(p, q) \leq & |\mathrm{DDS}(p, q) - \mathrm{D\hat{D}S}(X_s, X_t)| + \mathrm{D\hat{D}S}(X_s, X_t) \\ \leq & \mathrm{D\hat{D}S}(X_s, X_t) + \frac{M_{pq}}{\gamma^d}\sqrt{\frac{2(n_s+n_t)log(\frac{2}{\delta})}{n_s n_t}} + \mathcal{O}(n^{-\frac{1}{2}}).\end{aligned} \tag{3.19}$$

Combining Inequalities (3.14), (3.15), (3.17) and (3.19), this theorem is proved.

■

As discussed in [Mansour et al., 2009], $\hat{\Re}_{X_s}(\ell_H)$ in Inequality (3.18) has different upper bounds if a different ℓ is selected. If ℓ is a q-Lipschitz function, $\hat{\Re}_{X_s}(\ell_H) \leq 2q\hat{\Re}_{X_s}(H)$; if ℓ is 0-1 loss, $\hat{\Re}_{X_s}(\ell_H) \leq 2\hat{\Re}_{X_s}(H)$.

The significance of Theorem 3.5 is twofold. First, it highlights that the $\hat{\text{DDS}}(X_s, X_t)$ controls generalization performance in domain adaptation. Second, the bound shows a direct connection between DDS and the domain adaptation theory proposed in [Mansour et al., 2009].

3.5.2 DDS based Unsupervised Domain Adaption

This subsection presents an unsupervised domain adaptation method based on DDS, called diverse subsets adaptation (DSA). Since this chapter focuses on statistics to compare two distributions, DSA mainly focuses on finding a map T and a hypothesis h such that $\hat{\text{DDS}}(T(X_s), T(X_t))$ and $\ell(h(T(X_s)), f_p(X_s))$ approach zero, indicating that the aim of DSA is to solve the following optimization problem.

$$\min_{T,h} \ \ell(h(T(X_s)), f_p(X_s)) + \lambda \hat{\text{DDS}}(T(X_s), T(X_t)), \tag{3.20}$$

where λ is the penalty parameter of $\hat{\text{DDS}}(T(X_s), T(X_t))$. Because neural networks (NN) have a strong ability to find domain representations, we use an NN to find the map T and the hypothesis h in Formula (3.20). Thus, T and h has the following expressions.

$$T(x_{si}) = W_2 a(W_1 x_{si} + b_1) + b_2,$$

$$h(T(x_{si})) = W_3 T(x_{si}) + b_3,$$

where $a(\cdot)$ is a rectifier linear unit (ReLU) function, $a(x) = \max(0, x)$, W_1 is a d-by-d matrix, W_2 is a d_{rep}-by-d matrix, W_3 is a C-by-d_{rep} matrix, b_1 is a d-by-1 vector, b_2 is a d_{rep}-by-1 vector, b_3 is a C-by-1 vector. d_{rep} is a parameter that we need to select for different datasets. The cross-entropy loss function for the softmax regression is selected as ℓ:

$$\ell_c(h(T(x_{si})), y_{si}) = -\sum_{j=1}^{C} 1_{y_{si}=j} \log(h_j(x_{si})),$$

where $h_j(x_{si}) = h_j(T(x_{si}))/\sum_{j'} h_{j'}(T(x_{si}))$ is the softmax function that computes the probability of predicting sample x_{si} for the j^{th} class [Long et al., 2016a]. The solution of Formula (3.20) is obtained using Adam gradient decent algorithm: $\{W_i^*\}_{i=1}^3$ and $\{b_i^*\}_{i=1}^3$. Thus, we predict labels for $\mathbf{D_t}$ using

$$\hat{y}_{ti} = \arg\max_j \{(W_3^*(W_2^* a(W_1^* x_{ti} + b_1^*) + b_2^*) + b_3^*)_j\},$$

where $j = 1, ..., C$. Figure 3.1 shows the DSA procedures. It is clear that DSA is a four-layer neural network with a regularizer in the third layer. The discrepancy between distributions of two domains in the third layer is minimized because $\hat{\text{DDS}}(T^*(X_s), T^*(X_t))$ approaches zero, where $T^*(x_{si}) = W_2^* a(W_1^* x_{si} + b_1^*) + b_2^*$ and $T^*(x_{ti}) = W_2^* a(W_1^* x_{ti} + b_1^*) + b_2^*$. Thus, $T^*(X_s)$ and $T^*(X_t)$ can be seen as representations of two domains and other classifiers trained on $T^*(X_s)$ can be used to predict the labels of $\mathbf{D_t}$. In the Experiments section, we also report the classification performance of $T^*(X_s)$ combined with 1-nearest neighbors (1NN) and support vector machine (SVM).

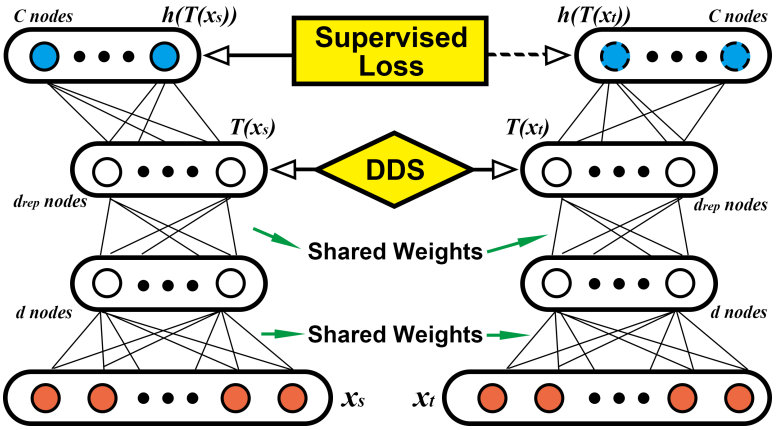


Figure 3.1: The proposed DDS based unsupervised domain adaptation (DSA). Red circles with the solid line represent the features of both domains, and blue circles with solid line represent in-sample (source domain) outputs of DSA, and blue circles with dash line represent out-sample (target domain) outputs of DSA. Other circles represent hidden neurons of DSA. In the third layer of DSA, representations of instances of both domains are obtained: $T^(X_s)$ and $T^(X_t)$.

3.6 Experiments

This section presents the experiments that applied DDS to solve two problems: two-sample test and unsupervised domain adaptation.

3.6.1 Evaluation of DDS for Two-sample Test

In our experiments, we first compared the Type I and Type II errors for DDS against other state-of-the-art tests on three artificial data sets, and then evaluated all four on one benchmark dataset.

3.6.1.1 Datasets

Following Gretton et al. [2012]; Jitkrittum et al. [2016]; Zaremba et al. [2013], we use Gaussian and Gaussian mixture distributions to construct our synthetic

data.

Syn-I. Synthetic data set I (Gaussian data with a drifting mean): In this experiment, we evaluated the power of DDS on a 2D Gaussian data with a covariance of 0. Drift severity was controlled by a predefined drift margin, denoted as $\Delta = \{0, 0.2, 0.4\}$, in which $|\mu_p - \mu_q| = \Delta$.

Syn-II. Synthetic data set II (mixture Gaussian data with a drifting mean): To make the distribution more complicated, we generated data from the 2D mixture Gaussian distributions with $\mu_1 = [0, 0]$, $\mu_2 = [0, 10]$. To simulate the mean drift, we added a margin $\Delta = \{0, 0.2, 0.4\}$ to both μ_1, μ_2, so that the total mean drift was controlled by Δ.

Higgs data. Similar to [Chwialkowski et al., 2015; Lhéritier and Cazals, 2018], we evaluated DDS on the Higgs data set. The objective is to distinguish the signature of processes producing Higgs bosons from those background processes that do not. We use four low-level features, azimuthal angular momenta φ for four particle jets, as described in [Chwialkowski et al., 2015; Lhéritier and Cazals, 2018], which means the distributions are on $\mathbb{R}^4$. We denote the jet φ momenta distribution of the background process as P, and the process that produces Higgs bosons as Q. The total sample size of mixed P and Q is 1.1×10^7, we randomly selected 1000 samples with replacement from each distribution for data integration. There are three types of data integration: Real-I where both two sample sets are drawn from P; Real-II, where both two sample sets are drawn from Q; and Real-III, where one sample is drawn from P and one from Q.

3.6.1.2 Experiments setup

The comparison tests and configurations are described below:

Compared tests. Maximum mean discrepancy (**MMD**) [Gretton et al., 2012]: We compared DDS with MMD (biased) using a bootstrap approach to calculate the rejection region with a median heuristic to determine the kernel bandwidth[1]. For a fair comparison, the DDS threshold was also chosen via bootstrap. Kernel optimized MMD (**OPT**) [Gretton et al., 2012]: OPT selects the kernel bandwidth and types of kernels according to the datasets. In this experiment the kernels used were Gaussian with an automatically selected bandwidth. The test step used MMD with this combination of kernels[2]. Low variance kernel two-sample tests (**B-tests**) [Zaremba et al., 2013]: B-tests also belongs to the family of MMD kernel two-sample tests[3]. As suggested in [Zaremba et al., 2013], we set B $=\sqrt{n_1}$. Smooth characteristic functions (**SCF**) and mean embedding (**ME**) [Chwialkowski et al., 2015; Jitkrittum et al., 2016][4]: SCF and ME build a distance between distributions based on characteristic functions at J random Fourier frequencies. As suggested in [Lhéritier and Cazals, 2018] and [Chwialkowski et al., 2015], we set $J = 10$, except in situations where the number of observations was less than 100, $J = 10$ causes high Type I errors; thus, we set $J = 1$ ($J = 1$ was selected in [Jitkrittum et al., 2016]).

Configurations. A Common configuration is a significance level of $\alpha = 0.05$ and 200 bootstrap sampling times. First, we generated several toy examples to evaluate the performance of DDS. For each test, we evaluated two samples $X_1 \sim p$ and $X_2 \sim q$, and defined the null hypothesis $H_0 : p = q$. To evaluate the power of the test with a different sample size, we changed the sample size

[1]The source code of MMD is available at http://www.gatsby.ucl.ac.uk/ gretton/mmd/mmd.htm.

[2]The Matlab implementation provided at http://www.gatsby.ucl.ac.uk/ gretton/adaptMMD/adaptMMD.htm.

[3]The Matlab implementation of B-tests is available at https://github.com/wojzaremba/btest.

[4]The source code of SCF tests is available at https://github.com/wittawatj/interpretable-test.

$m = \{50, 100, 500\}$, where $n_1 = n_2 = m$. We ran the test 150 times for each data type. L in the amplification function f defined in Eq. (3.5) is set to 20. β was selected from the range $[0.001, 20]$ by a grid search to maximize the average diversity of β-level subsets, as defined in (3.10).

3.6.1.3 Experimental results

This subsection presents the Type I and Type II errors for DDS and the other two-sample tests and shows how parameter L in amplification f influences the performance of DDS in the two-sample test task.

According to the results shown in Table 3.1, DDS achieved similar results to MMD on Gaussian mean drifting data and performed better on the other two types of distribution drift. We consider the main reason for the failure of MMD on the SynII data set was the choice of the kernel bandwidth. Evidence for this is that OPT and SCF outperformed MMD on this data set, both of which have kernel bandwidth selection advantage over MMD. However, the available combinations of kernel and kernel bandwidth may not be enough to cover all circumstances. The SCF test performed the better than MMD, OPT and B-test on Syn-II, which indicates that the optimized random Fourier features could help improve the testing performance. Table 3.2 and Table 3.3 report the average errors of DDS and the other two sample tests. Looking at both tables, MMD has the lowest Type I error, and DDS has the lowest Type II error among these tests. The average number of Type I errors for OPT and ME was greater than 5% (the threshold α), which means that both may not be qualified tests in three synthetic datasets. Comparing averages for Type II errors, DDS outperformed other tests, especially on Syn-II, indicating that DDS has more ability to handle complex

situations, such as mixture distributions. Table 3.4 shows the performance of these tests on real data. The results show that DDS performed better in terms of both Type I and Type II errors, but took more time than OPT, B-Test, ME and SCF. DDS is faster than MMD (DDS only needs 53% running time of MMD).

If parameter L were to be increased, DDS would have more power to distinguish between two distributions. Hence, we conducted an experiment to show how the test power of DDS changes when L is set to "20, 40, 60, 80". The average for the Type I errors changed from 4.813 ($L = 20$), 4.667 ($L = 40$), 4.741 ($L = 60$), to 4.593 ($L = 80$). The average for the Type II errors changed from 59.259 ($L = 20$), 59.370 ($L = 40$), 59.222 ($L = 60$), to 59.259 ($L = 80$). It is clear that increasing L slightly reduced the Type I error but did not significantly influence the Type II errors for DDS.

Table 3.1: Synthetic data set results (%). The null hypothesis H_0 is $p = q$. The $H_1^{\Delta_i}$ indicates the percentage of the test reject the null hypothesis, while the magnitude between p and q is Δ_i. Since Δ_1 is equal to 0, $H_1^{\Delta_1}$ is the Type I error (the lower the better), while $H_1^{\Delta_2}$ and $H_1^{\Delta_3}$ are the Type II error (the lower the better). Bold values represent the lowest error.

		$H_1^{\Delta_1}$			$H_1^{\Delta_2}$			$H_1^{\Delta_3}$		
		m_1	m_2	m_3	m_1	m_2	m_3	m_1	m_2	m_3
Syn-I	DDS	7.33	**4.00**	**3.33**	84.00	**70.00**	6.67	32.67	12.00	**0.00**
	MMD	5.33	4.67	4.67	83.33	**70.00**	**3.33**	30.67	**10.00**	**0.00**
	OPT	7.33	6.67	4.00	94.67	92.00	90.67	89.33	82.00	66.00
	B-test	9.33	11.33	6.67	90.67	86.67	68.67	74.67	61.33	4.67
	ME	**3.33**	6.67	5.33	**81.33**	85.33	10.00	**30.00**	18.67	1.33
	SCF	2.67	7.33	6.00	82.67	72.67	8.67	48.00	22.00	1.33
Syn-II	DDS	2.00	1.33	5.33	96.00	85.33	**22.00**	**78.67**	**33.33**	**0.00**
	MMD	**0.00**	**0.00**	**0.00**	100	100.00	100.00	100.00	100.00	79.33
	OPT	2.67	4.00	2.00	98.00	95.33	94.00	96.00	92.00	66.00
	B-test	7.33	4.67	3.33	94.00	96.00	89.33	90.00	84.67	54.00
	ME	2.67	7.33	4.00	96.00	**77.33**	33.33	80.67	46.67	1.33
	SCF	0.67	0.00	2.67	**92.00**	86.00	36.00	80.67	49.33	0.67

Table 3.2: Average Type I error (%) and corresponding rankings. This table shows the average Type I error for each test and the average values of Type I errors obtained under m_1, m_2, m_3 (see Table 3.1). Bold values represent the lowest error.

Datasets	Criterion	DDS	MMD	OPT	B-test	ME	SCF
Syn-I	Type I error	**4.887 (1)**	4.890 (2)	6.000 (5)	9.110 (6)	5.110 (3)	5.333 (4)
Syn-II	Type I error	4.887 (4)	**4.013 (2)**	4.890 (5)	5.110 (6)	4.667 (3)	3.993 (1)

Table 3.3: Average Type II error (%) and corresponding rankings. This table shows the average Type II error for each test and the average values of Type II errors obtained under m_1, m_2, m_3 (see Table 3.1). Bold values represent the lowest error.

Datasets	Criterion	DDS	MMD	OPT	B-test	ME	SCF
Syn-I	Type II error	34.223 (2)	**32.888 (1)**	85.778 (6)	64.447 (5)	37.777 (3)	39.223 (4)
Syn-II	Type II error	**52.555 (1)**	96.555 (6)	90.222 (5)	84.667 (4)	55.888 (2)	57.445 (3)

Table 3.4: Higgs data set results (%) and corresponding rankings. Real-I has both two-sample sets drawn from P distribution. Real-II has bath two sample sets drawn from Q distribution. Real-III has one sample set drawn from P and the other drawn from Q. Bold values represent the lowest error and lowest running time.

	Real-I (Type I)	Real-II (Type I)	Real-III (Type II)	Average Time (s)
DDS	**2.67 (1)**	**2.67 (1)**	**82.00 (1)**	3.631 (5)
MMD	6.00 (4)	5.33 (3)	90.00 (4)	6.840 (6)
OPT	6.67 (6)	5.33 (3)	91.33 (5)	0.084 (4)
B-test	4.67 (3)	8.00 (6)	92.67 (6)	0.038 (3)
ME	**2.67 (1)**	3.33 (2)	85.33 (3)	**0.016 (1)**
SCF	6.00 (4)	5.33 (3)	83.33 (2)	**0.016 (1)**

3.6.2 Evaluation of DDS for Unsupervised Domain Adaptation

We compared the DSA method to competitive test statistics used in unsupervised domain adaptation on three benchmark datasets.

3.6.2.1 Datasets

COIL20, PIE, and Office31 are three benchmark datasets widely adopted to evaluate domain adaptation methods.

COIL20 contains 20 objects with 1440 images. The images of each object were taken 5 degrees apart as the object is rotated on a turntable and each object has 72 images. Following [Long et al., 2013], COIL20 was separated into two subsets **COIL1** and **COIL2**: COIL1 and COIL2 contains images taken in different directions. In this way, subsets COIL1 and COIL2 follow relatively different distributions. One dataset COIL1 vs COIL2 was constructed by selecting all 720 images in COIL1 to form the source data, and all 720 images in COIL2 to

form the target data. We switched the source/target to create another dataset COIL2 vs COIL1.

PIE, which stands for “Pose, Illumination, Expression", is a benchmark face database [Long et al., 2013]. Following [Long et al., 2013], we selected **PIE1** (C05, left pose), **PIE2** (C07, upward pose), **PIE3** (C09, downward pose), **PIE4** (C27, frontal pose) and **PIE5** (C29, right pose). In each subset (pose), all the face images were taken under different lighting, illumination and expression conditions. We can construct 20 domain adaptation tasks, e.g., PIE1 vs PIE2, PIE1 vs PIE3, ..., PIE5 vs PIE4. In this way, the source and target data were constructed using face images from different poses, and, thus, follow significantly different distributions.

Office-31 contains three object domains, i.e., Amazon (images downloaded from online merchants), webcam (low-resolution images by a web camera), and DSLR (high-resolution images by a digital SLR camera). It consists of 4,652 images with 31 categories. Specifically, we have three domains, A (Amazon), W (webcam) and D (DSLR). By randomly selecting two different domains as the source domain and the target domain, we constructed 3×2=6 cross-domain object data sets, e.g., $A \rightarrow W$, $A \rightarrow D$, ..., $D \rightarrow W$. We used DeCAF$_7$ features for the Office-31 dataset.

The characteristics of each dataset is summarized in Table 3.5.

Table 3.5: Real-world datasets for transfer learning

Dataset	#Instances	#Attributes	#Class	Description
COIL20	1,440	1,024	20	COIL1, COIL2
Office31	4,652	4,096	31	A, W, D
PIE	11,554	1,024	68	PIE1, ..., PIE5

3.6.2.2 Experiments setup

The compared domain adaptation methods and configurations are described below:

Compared test statistics. Maximum mean discrepancy (**MMD**) [Gretton et al., 2012] is the most common statistic used in the domain adaptation field [Ghifary et al., 2017; Long et al., 2015, 2016a, 2017; Pan et al., 2011]. It was proposed to test whether two samples come from the same distribution [Gretton et al., 2012] and has similar properties to DDS including consistency [Gretton et al., 2012], semimetric/metric [Gretton et al., 2012], asymptotic behavior [Gretton et al., 2012], and a generalization bound for domain adaptation (under a specific hypothesis set) [Ghifary et al., 2017]. Thus, our main comparator was MMD. To fairly compare DDS to MMD in the domain adaptation tasks, "MMD+NN" was implemented to minimize discrepancies between two domains (via MMD) for transferring knowledge. "MMD+NN" aims to solve the following optimization problem.

$$\min_{T,h} \ \ell(h(T(X_s)), f_p(X_s)) + \lambda \text{MMD}(T(X_s), T(X_t)), \tag{3.21}$$

where ℓ, T and h are the same as ℓ, T and h of "DDS+NN" (DSA). Adam gradient decent algorithm is applied to solve Formula (3.21). Comparing DSA to "MMD+NN" demonstrates that DDS outperforms MMD in domain adaptation tasks. Similar to DSA, "MMD+NN" also constructs representations of two domains. Combining the constructed representations with 1-NN and SVM forms other competitive methods. 1 nearest neighbor (1-NN) and SVM are two non-transfer methods that show the effectiveness of domain adaptation methods. We also compared DSA to common domain adaptation methods, including *geodesic*

flow kernel (GFK) [Gong et al., 2014], *transfer component analysis* (TCA) [Pan et al., 2011], *domain invariant projection* (DIP-CC) [Baktashmotlagh et al., 2013], *correlation alignment* (CORAL) [Sun et al., 2016], *deep domain confusion* (DDC) [Tzeng et al., 2014], *joint adaptation networks* (JAN) [Long et al., 2017] and *domain adversarial neural networks* (DANN) [Ganin et al., 2016b].

Configurations. For all datasets, L was set to 4 for DSA. We used Adam gradient decent algorithm with full batches and set the parameter $\epsilon = 10^{-8}$ and the exponential decay rates for the moment estimates $\beta_1 = 0.5$, $\beta_2 = 0.9$. Since the target domains do not contain any labeled data, it was impossible to automatically tune the optimal parameters for the target classifier using cross-validation. Thus, we evaluated all methods by empirically searching the parameter space for the optimal parameter settings, and report the best results for each method and task. We used grid-search to tune the free parameters for DSA and "MMD+NN". We searched λ from $\{0.01, 0, 1, 1, 5, 10, 30, 50\}$ and the learning rate η of Adam algorithm from $\{10^{-5}, 1.5 \times 10^{-5}, 2 \times 10^{-5}, ..., 10 \times 10^{-5}\}$. We selected β for DSA from $\{0.5 \times \beta_{best}, 0.6 \times \beta_{best}, ..., 2 \times \beta_{best}\}$, where $\beta_{best} = arg\max_{\beta}\{\mathcal{D}_{\beta}(S(\mathcal{X}))\}$ as demonstrated in Section 3.3.3. We did not calculate γ in every loop. γ was set to 2 in domain adaptation tasks, which is large enough for all involved datasets. d_{rep} for COIL20 dataset is set to 1024 and for the other two datasets is set to 2000.

Following [Li et al., 2014; Pan et al., 2011; Pan and Yang, 2010; Xiao and Guo, 2015], SVM was trained on the representations ($T^*(x_{si})$) of the source domain, then tested on the unlabeled target domain. We used LIBSVM's default parameters for all classification tasks: the bandwidth of the radial basis function kernel was set to $1/d_{rep}$; the cost C was set to 1; and the tolerance of the termination criterion was set to 0.001. Similar to SVM, 1-NN was trained on

the representations ($T^*(x_{si})$) of the source domain, then tested on the unlabeled target domain. $Accuracy$ was used as the test metric as it has been widely adopted in the literature [Ghifary et al., 2017; Li et al., 2014; Pan et al., 2011]. The definition follows.

$$Accuracy = \frac{|x \in X_t : h(x) = y(x)|}{|x \in X_t|},$$

where $y(x)$ is the ground truth label of x, while $h(x)$ is the label predicted by domain adaptation methods.

3.6.2.3 Experimental results

The average accuracy for each method is reported in Table 3.6. From this table, it is clear that DDS performed better than MMD in most tasks. In tasks $W \rightarrow A$ and PIE5 vs PIE1, MMD outperformed DDS when the NN classifier was selected. DDS had much better performance in face recognition transfer tasks, such as PIE2 vs PIE 3 (improved around 25%), PIE2 vs PIE 3 (improved around 18%), PIE5 vs PIE2 (improved around 15%) and PIE5 vs PIE3 (improved around 14%). This may be because there are many local differences in face images and DDS are more suitable for measuring distributions discrepancy than MMD. In general, 1) the NN classifier outperformed the 1-NN and SVM classifiers no matter which statistic was used to measure distance between two distributions; 2) the domain adaptation methods outperformed 1-NN, SVM and NN, which shows that reducing the distribution discrepancy is key to domain adaptation.

We also compare DSA to common unsupervised domain adaptation methods. The following results were observed: 1) "MMD+NN" performed better than TCA and DIP-CC, indicating that neural networks were more suitable for finding

$T(x)$ than linear or kernel maps; 2) DSA outperformed the methods that only consider minimizing the marginal distribution discrepancy, which indicates that classification performance in the target domain benefits more from minimizing DDS than from minimizing other statistics; 3) methods that consider aligning joint distributions of two domains performed better than DSA because aligning joint distributions may reduce $\mathcal{L}_q(h_p^*, h_q^*)$ in Inequality (3.18) with the current benchmark datasets; 4) end-to-end methods (deep domain adaptation methods) had much better performance than shadow methods (including DSA). In the future, we will develop an unsupervised domain adaptation method based on minimizing DDS within end-to-end domain adaptation structure.

3.7 Summary

This chapter presents a test statistic called *discrepancy of diverse subsets* (DDS) to compare two multivariate distributions. The DDS autonomously constructs β-level subsets on a topological space $\mathcal{X}$ for two samples drawn from two distributions, then compares the total impact that the two samples have on each β-level subset to result in the final discrepancy between two samples. DDS is a nonparametric test statistic and the results of the experiments demonstrate its performance on two problems: 1) two-sample test and 2) unsupervised domain adaptation. Several properties of DDS are proved: 1) the empirical DDS is a consistent estimator; 2) DDS is a semimetric for two probability measures; 3) the asymptotic null distribution of the empirical DDS is a Normal distribution; and 4) DDS controls generalization performance in domain adaptation, i.e., the DDS-based generalization bound is proved. These properties theoretically guarantee

Table 3.6: Classification accuracy for DDS-based domain adaptation and MMD-based domain adaptation on 28 transfer recognition tasks (object recognition (the first 8 rows) and face recognition (the other rows)). The results show that DDS outperformed MMD in 26 transfer tasks. Bold values represent the highest accuracy. Please note that 1-NN means the 1 nearest neighbor and NN means the neural network.

Tasks	1-NN	SVM	NN	MMD+NN			DDS+NN (DSA)		
				1-NN	SVM	NN	1-NN	SVM	NN
COIL1 vs COIL2	83.33	81.94	80.83	98.2	95.64	97.77	**99.31**	98.75	95.69
COIL2 vs COIL1	84.72	82.64	75.97	95.69	98.19	95.69	**98.33**	97.78	94.31
W→A	38.41	41.68	44.98	42.56	47.64	**50.51**	41.56	45.12	49.7
A→W	43.65	54.72	57.48	62.77	64.66	64.15	55.09	64.53	**65.03**
W→D	96.59	97.79	95.98	76.51	95.98	97.39	92.57	97.79	**98.59**
D→W	87.55	89.94	93.08	92.58	92.33	93.96	85.79	91.07	**94.21**
A→D	54.01	59.64	62.45	62.65	65.26	63.45	59.64	64.85	**67.27**
D→A	37.88	42.95	44.3	44.48	45.47	47.71	40.65	44.55	**49.2**
PIE1 vs PIE 2	30.82	26.21	25.23	34.81	34.5	38.12	**45**	39.04	44.44
PIE1 vs PIE 3	33.88	28.74	42.89	37.32	37.13	43.87	37.93	37.99	**44.91**
PIE1 vs PIE 4	37.73	29.95	25.8	69	62.06	64.58	**74.07**	64.16	68.46
PIE1 vs PIE 5	13.11	16.05	16.48	22.18	19.42	24.69	29.53	21.14	**37.93**
PIE2 vs PIE 1	26.05	21.54	21.31	35.56	39.14	39.05	40.94	39.56	**41.18**
PIE2 vs PIE 3	53.86	35.48	23.53	35.85	36.39	39.09	**64.46**	50.98	50.37
PIE2 vs PIE 4	49.71	49.95	36.5	52.27	52.27	57.5	58.4	54.88	**59.65**
PIE2 vs PIE 5	31.07	23.47	19	25.67	22.49	23.6	35.72	31.25	**36.58**
PIE3 vs PIE 1	22.69	23.83	20.53	37.64	39.17	45.68	35.65	40.52	**46.31**
PIE3 vs PIE 2	36.96	34.5	29.65	36.16	39.47	41.62	**59.91**	46.78	52.12
PIE3 vs PIE 4	46.26	44.61	46.8	51.48	58.72	60.65	62.06	64.31	**68.91**
PIE3 vs PIE 5	20.89	21.81	25.06	29.29	28.86	30.7	33.02	27.63	**34.5**
PIE4 vs PIE 1	31.9	39.32	43.61	67.62	63.75	69.63	**71.97**	64.44	68.13
PIE4 vs PIE 2	69.49	59.91	54.51	54.51	68.02	71.15	67.22	71.88	**73.11**
PIE4 vs PIE 3	78.49	62.81	54.53	70.21	70.28	72.18	**78.61**	70.96	75.98
PIE4 vs PIE 5	38.17	41.48	35.96	35.23	36.09	40.77	39.15	47.79	**51.72**
PIE5 vs PIE 1	14.2	24.78	27.58	35.32	42.35	**46.61**	37.39	43.94	46.21
PIE5 vs PIE 2	29.59	29.9	27.44	34.99	39.47	41.43	**56.35**	44.63	48.43
PIE5 vs PIE 3	32.41	29.29	28.62	35.97	40.81	43.26	**56.92**	53.49	53.93
PIE5 vs PIE 4	26.73	31.72	44.85	38.81	43.37	50.86	44.15	48.96	51.15
Average	44.65	43.81	43.03	50.55	52.82	55.56	57.19	56.03	**59.57**

good performance when using DDS to help solve above two problems.

CHAPTER 4

MAXIMUM MEAN DISCREPANCY WITH LEARNED DEEP KERNELS: A NON-PARAMETRIC TWO-SAMPLE TEST FOR HIGH-DIMENSION DATA

4.1 Introduction

A popular class of non-parametric two-sample tests is based on kernel methods [Smola and Schölkopf, 2001]: such tests construct a *kernel mean embedding* [Berlinet and Thomas-Agnan, 2004; Muandet et al., 2017] for each distribution, and measure the difference in these embeddings. For any *characteristic* kernel, two distributions are the same if and only if their mean embeddings are the same; the distance between mean embeddings is the *maximum mean discrepancy* (MMD) [Gretton et al., 2012]. There are also several closely related methods, including tests based on checking for differences in mean embeddings evaluated

at specific locations [Chwialkowski et al., 2015; Jitkrittum et al., 2016] and kernel Fisher discriminant analysis [Harchaoui et al., 2007]. These tests all work well for samples from simple distributions when using appropriate kernels.

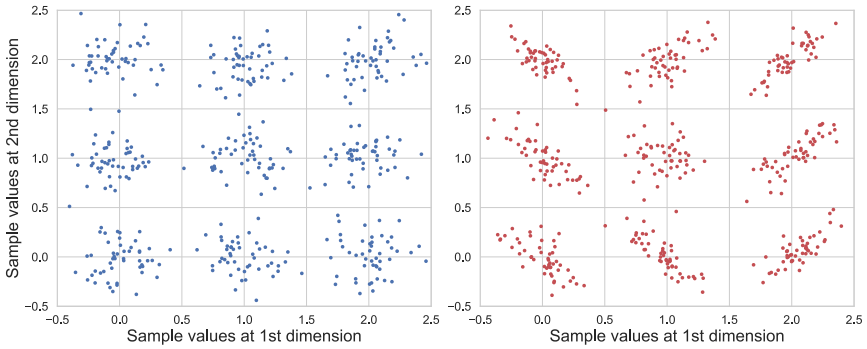
Problems that we care about, however, often involve distributions with complex structure, where simple kernels will often map distinct distributions to nearby (and hence hard to distinguish) mean embeddings. Figure 4.1a shows an example of a multimodal dataset, where the overall modes align but the submode structure varies differently at each mode. A translation-invariant Gaussian kernel only "looks at" the data uniformly within each mode (see Figure 4.1b), requiring many samples to correctly distinguish the two distributions. The distributions can be distinguished more effectively if we understand the structure of each mode, as with the more complex kernel illustrated in Figure 4.1c.

To model these complex functions, we adopt a *deep kernel* approach [Jean et al., 2018; Li et al., 2017; Sutherland et al., 2017; Wenliang et al., 2019; Wilson et al., 2016], building a kernel with a deep network. In this chapter, we use

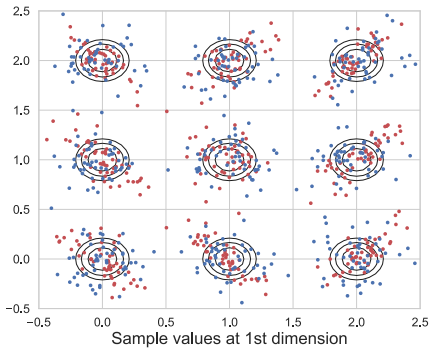
$$k_\omega(x,y) = [(1-\epsilon)\kappa(\phi_\omega(x),\phi_\omega(y)) + \epsilon]q(x,y), \tag{4.1}$$

where the deep neural network ϕ_ω extracts features of samples, and κ is a simple kernel (e.g., a Gaussian) on those features, while q is a simple characteristic kernel (e.g. Gaussian) on the input space. With an appropriate choice of ϕ_ω, this allows for extremely flexible kernels which can learn complex behavior very different in different parts of space. This choice is discussed further in Section 4.5.

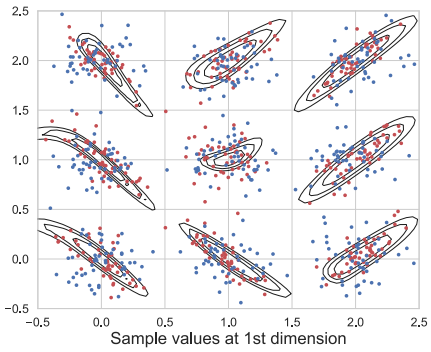
These complex kernels, though, cannot feasibly be specified by hand or simple heuristics, as is typical practice in kernel methods. We select the parameters



(a) Samples drawn from $\mathbb{P}$ (left) and $\mathbb{Q}$ (right).



(b) Contour of Gaussian



(c) Contour of deep kernel

Figure 4.1: Blob dataset (a), with contours of Gaussian kernel (b) and deep kernel (c) evaluated at 9 locations (contour values are $0.7, 0.8$ and 0.9). Each distribution has 9 modes; the central modes have the same shape, but $\mathbb{Q}$ has a different shape at each other mode. A Gaussian kernel (b) compares points isotropically throughout the space; contours show $k(x, \mu)$ for each mode μ. A deep kernel (c) learned by our methods compares points differently in different locations, allowing better identification of differences between $\mathbb{P}$ and $\mathbb{Q}$.

ω by maximizing the ratio of the MMD to its variance, which maximizes test power at large sample sizes. This procedure was proposed by Sutherland et al. [2017], but we establish for the first time that it gives consistent selection of the best kernel in the class, whether optimizing our deep kernels with hundreds of thousands of parameters or simply choosing lengthscales of a Gaussian as did in Sutherland et al. [2017]. Previously, there were no guarantees this procedure would yield a kernel which generalized at all from the training set to a test set.

Another way to compare distributions is to train a classifier between them, and evaluate its accuracy [Lopez-Paz and Oquab, 2017]. We show, perhaps surprisingly, that our framework encompasses this approach, but deep kernels allow for more general model classes which can use the data more efficiently. We also train representations directly to maximize test power, rather than a cross-entropy surrogate.

We test our method on several simulated and real-world datasets, including complex synthetic distributions, high-energy physics data, and challenging image problems. We find convincingly that learned deep kernels outperform simple shallow methods, and learning by maximizing test power outperforms learning through a cross-entropy surrogate loss.

4.2 Concepts and Notations

Two-sample test. Let $\mathcal{X}$ be a separable metric space – in this chapter, typically a subset of $\mathbb{R}^d$ – and $\mathbb{P}$, $\mathbb{Q}$ be Borel probability measures on $\mathcal{X}$. We observe independent identically distributed (*i.i.d.*) samples $S_{\mathbb{P}} = \{x_i\}_{i=1}^n \sim \mathbb{P}^n$ and $S_{\mathbb{Q}} = \{y_j\}_{j=1}^m \sim \mathbb{Q}^m$. We wish to know whether $S_{\mathbb{P}}$ and $S_{\mathbb{Q}}$ come from the same

distribution: does $\mathbb{P} = \mathbb{Q}$?

We use the null hypothesis testing framework, where the null hypothesis $H_0 : \mathbb{P} = \mathbb{Q}$ is tested against the alternative hypothesis $H_1 : \mathbb{P} \neq \mathbb{Q}$. We perform a two-sample test in four steps: select a significance level $\alpha \in [0,1]$; compute a test statistic $\hat{t}(S_{\mathbb{P}}, S_{\mathbb{Q}})$; compute the p-value $\hat{p} = \Pr_{H_0}(T > \hat{t})$, the probability of the two-sample test returning a statistic as large as $\hat{t}$ when H_0 is true; finally, reject H_0 if $\hat{p} < \alpha$.

Maximum mean discrepancy (MMD). We will base our two-sample test statistic on an estimate of a distance between distributions. Our metric, the MMD, is defined in terms of a kernel k giving point-level "similarities" on $\mathcal{X}$.

Definition 4.1 (Gretton et al., 2012)**.** *Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be the kernel of a reproducing kernel Hilbert space $\mathcal{H}_k$, with feature maps $k(\cdot, x) \in \mathcal{H}_k$. Let $X, X' \sim \mathbb{P}$ and $Y, Y' \sim \mathbb{Q}$, and define the* kernel mean embeddings $\mu_{\mathbb{P}} := \mathbb{E}[k(\cdot, X)]$ *and* $\mu_{\mathbb{Q}} := \mathbb{E}[k(\cdot, Y)]$*. Under mild integrability conditions,*

$$\begin{aligned} \mathrm{MMD}(\mathbb{P}, \mathbb{Q}; \mathcal{H}_k) &:= \sup_{f \in \mathcal{H}, \|f\|_{\mathcal{H}_k} \leq 1} |\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]| \\ &= \left\| \mu_{\mathbb{P}} - \mu_{\mathbb{Q}} \right\|_{\mathcal{H}_k} = \sqrt{\mathbb{E}[k(X, X') + k(Y, Y') - 2k(X, Y)]}. \end{aligned}$$

For characteristic *kernels, $\mu_{\mathbb{P}} = \mu_{\mathbb{Q}}$ implies $\mathbb{P} = \mathbb{Q}$, hence $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}; \mathcal{H}_k) = 0$ if and only if $\mathbb{P} = \mathbb{Q}$.*

The first form shows that the MMD is an integral probability metric [Müller, 1997], along with such popular distances as the Wasserstein and total variation.

There are several natural estimators of the MMD from samples. We will assume $n = m$ and use the U-statistic estimator, which is unbiased for MMD^2

and has nearly minimal variance among unbiased estimators [Gretton et al., 2012]:

$$\widehat{\mathrm{MMD}}_u^2(S_{\mathbb{P}}, S_{\mathbb{Q}}; k) = \frac{1}{n(n-1)} \sum_{i \neq j} H_{ij} \tag{4.2}$$

$$H_{ij} = k(X_i, X_j) + k(Y_i, Y_j) - k(X_i, Y_j) - k(Y_i, X_j).$$

The similar $\widehat{\mathrm{MMD}}_b^2 := \frac{1}{n^2} \sum_{ij} H_{ij}$ is the squared MMD between the empirical distributions of $S_{\mathbb{P}}$ and $S_{\mathbb{Q}}$.[1]

Testing with the MMD. It can be shown that under H_0, $n\widehat{\mathrm{MMD}}_u^2$ converges to a distribution depending on $\mathbb{P}$ and k; we thus use this as our test statistic.

Proposition 4.1 (Asymptotics of $\widehat{\mathrm{MMD}}_u^2$)**.** *Under the null hypothesis, $H_0 : \mathbb{P} = \mathbb{Q}$, we have if $Z_i \sim \mathcal{N}(0, \sqrt{2}^2)$,*

$$n\widehat{\mathrm{MMD}}_u^2 \xrightarrow{\boldsymbol{d}} \sum_i \lambda_i (Z_i^2 - 2);$$

here λ_i are the eigenvalues of the $\mathbb{P}$-covariance operator of the centered kernel [Gretton et al., 2012, Theorem 12].

Under the alternative, $H_1 : \mathbb{P} \neq \mathbb{Q}$, a standard central limit theorem holds [Serfling, 1980, Section 5.5.1]:

$$\sqrt{n}(\widehat{\mathrm{MMD}}_u^2 - \mathrm{MMD}^2) \xrightarrow{\boldsymbol{d}} \mathcal{N}(0, \sigma_{H_1}^2)$$

$$\sigma_{H_1}^2 := 4\left(\mathbb{E}[H_{12}H_{13}] - \mathbb{E}[H_{12}]^2\right).$$

Although it is possible to construct a test based on directly estimating this null distribution [Gretton et al., 2009], it is both simpler and, if implemented

[1] Including $k(X_i, Y_i)$ terms in $\widehat{\mathrm{MMD}}_u$ gives the minimal variance unbiased estimator, and allows $m \neq n$. The U-statistic is more convenient for analysis and for efficient permutations; in our settings it behaves similarly to the MVUE and $\widehat{\mathrm{MMD}}_b^2$.

carefully, faster [Sutherland et al., 2017] to instead use a permutation test. This general method [Alba Fernández et al., 2008; Dwass, 1957] observes that under H_0, the samples from $\mathbb{P}$ and $\mathbb{Q}$ are interchangeable; we can therefore estimate the null distribution of our test statistic by repeatedly re-computing it with the samples randomly re-assigned to $S_\mathbb{P}$ or $S_\mathbb{Q}$.

Test power. The main measure of efficacy of a null hypothesis test is its *power*: the probability that, for a particular $\mathbb{P} \neq \mathbb{Q}$ and n, we correctly reject H_0. Proposition 4.1 implies, where Φ is the standard normal CDF, that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}_u^2 > r\right) \to \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{r}{\sqrt{n}\,\sigma_{H_1}}\right);$$

we can find the approximate test power by using the rejection threshold, found via (e.g.) permutation testing, as r. We also know via Proposition 4.1 that this r will converge to a constant, and MMD, σ_{H_1} are also constants. For reasonably large n, the power is dominated by the first term, and the kernel yielding the most powerful test will approximately maximize [Sutherland et al., 2017]

$$J(\mathbb{P}, \mathbb{Q}; k) := \mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}; k)/\sigma_{H_1}(\mathbb{P}, \mathbb{Q}; k). \tag{4.3}$$

Selecting a kernel. The criterion $J(\mathbb{P}, \mathbb{Q}; k)$ depends on the particular $\mathbb{P}$ and $\mathbb{Q}$ at hand, and thus we typically will neither be able to choose a kernel *a priori*, nor exactly evaluate J given samples. We can, however, estimate it with

$$\hat{J}_\lambda(S_\mathbb{P}, S_\mathbb{Q}; k) := \frac{\widehat{\mathrm{MMD}}_u^2(S_\mathbb{P}, S_\mathbb{Q}; k)}{\hat{\sigma}_{H_1,\lambda}(S_\mathbb{P}, S_\mathbb{Q}; k)}, \tag{4.4}$$

where $\hat{\sigma}^2_{H_1,\lambda}$ is a regularized estimator of $\sigma^2_{H_1}$ given by[2]

$$\frac{4}{n^3}\sum_{i=1}^{n}\left(\sum_{j=1}^{n}H_{ij}\right)^2 - \frac{4}{n^4}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}H_{ij}\right)^2 + \lambda. \tag{4.5}$$

Given $S_{\mathbb{P}}$ and $S_{\mathbb{Q}}$, we could construct a test by choosing k to maximize $\hat{J}_\lambda(S_{\mathbb{P}}, S_{\mathbb{Q}}; k)$, then using a test statistic based on $\widehat{\mathrm{MMD}}(S_{\mathbb{P}}, S_{\mathbb{Q}}; k)$. This sample re-use, however, violates the conditions of Proposition 4.1, and permutation testing would require repeatedly re-training k with permuted labels.

Thus we split the data, get $k^{tr} \approx \arg\max_k \hat{J}_\lambda(S^{tr}_{\mathbb{P}}, S^{tr}_{\mathbb{Q}}; k)$, then compute the test statistic and permutation threshold on $S^{te}_{\mathbb{P}}$, $S^{te}_{\mathbb{Q}}$ using k^{tr}. This procedure was proposed for $\widehat{\mathrm{MMD}}^2_u$ by Sutherland et al. [2017], but the same technique works for a variety of tests [Gretton et al., 2012; Jitkrittum et al., 2016, 2017; Lopez-Paz and Oquab, 2017]. Our paper adopts this framework (Section 4.5) and studies it further.

MMD-GANs [Binkowski et al., 2018] seek a model $\mathbb{Q}_\theta$ to matches a target $\mathbb{P}$ according to a kernel optimized to distinguish the two. For instance, if $\mathbb{Q}_\theta$ is quite far from $\mathbb{P}$, an MMD-GAN requires a "weak" kernel for $\mathbb{Q}_\theta$ to find a path for improvement [Arbel et al., 2018], while our ideal kernel is one which perfectly distinguishes $\mathbb{P}$ and $\mathbb{Q}_\theta$ and would likely give no signal for improvement. Our algorithm, theoretical guarantees, and empirical evaluations thus all differ significantly from those for MMD-GANs.

[2]This estimator, as a V-statistic, is biased even when $\lambda = 0$ (although this bias is only $O(1/N)$; see Lemma A.3). Although Sutherland [2019]; Sutherland et al. [2017] give a quadratic-time estimator unbiased for $\sigma^2_{H_1}$, it is much more complicated to implement and analyze, likely has higher variance, and (being unbiased) can be negative, especially e.g. when the kernel is poor.

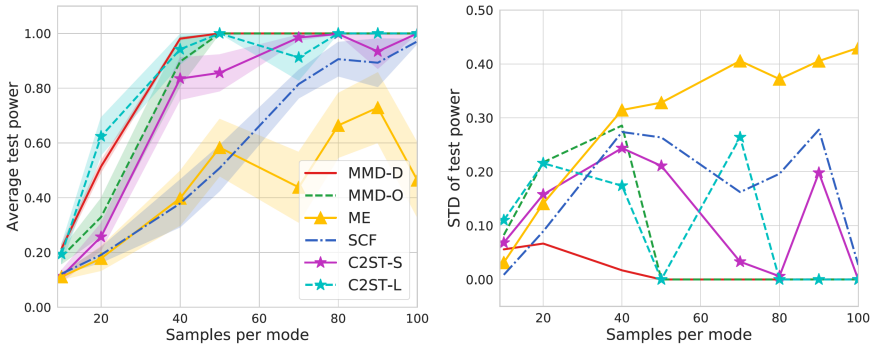
4.3 Limits of Simple Kernels

We can use the criterion $\hat{J}_\lambda$ of (4.4) even to select parameters among a simple family, such as the lengthscale of a Gaussian kernel. Doing so on the *Blob* problem of Figure 4.1 illustrates the limitations of using MMD with these kernels.

In Figure 4.2c, we show how the maximal value of $\hat{J}$ changes as we see more samples from $\mathbb{P}$ and $\mathbb{Q}$, for both a family of Gaussian kernels (green dashed line) and a family (4.1) of deep kernels (red line). The optimal $\hat{J}$ is always higher for the deep kernels; as expected, the empirical test power (Figure 4.2a) is also higher for deep kernels.

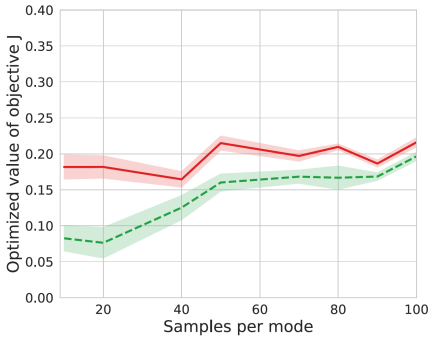
Most simple kernels used for MMD tests, whether the Gaussian we use here or Laplace, inverse multiquadric, even automatic relevance determination kernels, are all translation invariant: $k(x, y) = k(x - t, y - t)$ for any $t \in \mathbb{R}^d$. (All kernels used by Sutherland et al. [2017], for instance, were of this type.) Hence the kernel behaves the same way across space, as in Figure 4.1b. This means that for distributions whose behavior varies through space, whether because principal directions change (as in Figure 4.1) so the shape should be different, or because some regions are much denser than others and so need a smaller lengthscale [e.g. Wenliang et al., 2019, Figures 1 and 2], any single global choice is suboptimal.

Kernels which are not translation invariant, such as the deep kernels (4.1) shown in Figure 4.1c, can adapt to the different shapes necessary in different areas.

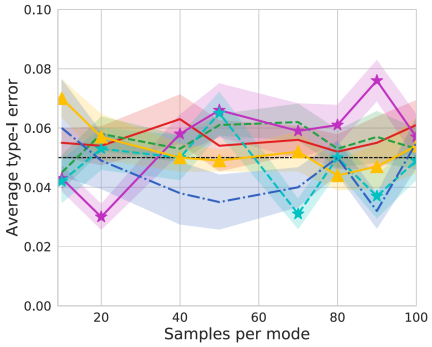


(a) Average test power

(b) STD of test power



(c) $\hat{J}_\lambda$ value (4.4)



(d) Type-I error

Figure 4.2: Results on *Blob-S* and *Blob-D* given $\alpha = 0.05$; see Section 4.7 for details. n_b is the number of samples at each mode, so $n_b = 100$ means drawing 900 samples from each of $\mathbb{P}$ and $\mathbb{Q}$. We report, when increasing n_b, (a) average test power, (b) standard deviation of test power, (c) the value of $\hat{J}_\lambda$, and (d) average type-I error. (a), (b) and (c) are on *Blob-D*, and (d) is on *Blob-S*. Shaded regions show standard errors for the mean, and the black line shows α.

4.4 Relationship to Classifier-Based Tests

Another popular method for conducting two-sample tests is to train a classifier between $S^{tr}_{\mathbb{P}}$ and $S^{tr}_{\mathbb{Q}}$, then assess its performance on $S^{te}_{\mathbb{P}}$, $S^{te}_{\mathbb{Q}}$. If $\mathbb{P} = \mathbb{Q}$, the classification problem is impossible and performance will be at chance.

The most common performance metric is the accuracy [Lopez-Paz and Oquab, 2017]; this scheme is fairly common among practitioners, and Ramdas et al.

[2016] showed it to be rate-optimal in one extremely limited setting (linear discriminant analysis between high-dimensional Gaussians with identical covariances). We will call this approach a Classifier Two-Sample Test based on Sign, C2ST-S. Letting $f : \mathcal{X} \to \mathbb{R}$ output classification scores, the C2ST-S statistic is $\widehat{\text{acc}}(S_{\mathbb{P}}, S_{\mathbb{Q}}; f)$ given by

$$\frac{1}{2n} \sum_{X_i \in S_{\mathbb{P}}} \mathbb{1}(f(X_i) > 0) + \frac{1}{2n} \sum_{Y_i \in S_{\mathbb{Q}}} \mathbb{1}(f(Y_i) \leq 0).$$

Let $\text{acc}(\mathbb{P}, \mathbb{Q}; f) := \frac{1}{2}\Pr(f(X) > 0) + \frac{1}{2}\Pr(f(Y) \leq 0)$; $\widehat{\text{acc}}$ is unbiased for acc and has a simple asymptotically normal null distribution.

Although it is perhaps not immediately obvious this is the case, C2ST-S is almost a special case of the MMD. Let

$$k_f^{(S)}(x, y) = \frac{1}{4} \mathbb{1}(f(x) > 0) \mathbb{1}(f(y) > 0). \tag{4.6}$$

A C2ST-S with f is equivalent to an MMD test with $k_f^{(S)}$:

Proposition 4.2. *It holds that*

$$\text{MMD}(\mathbb{P}, \mathbb{Q}; k_f^{(S)}) = \left| \text{acc}(\mathbb{P}, \mathbb{Q}; f) - \frac{1}{2} \right|$$
$$\widehat{\text{MMD}}_b(S_{\mathbb{P}}, S_{\mathbb{Q}}; k_f^{(S)}) = \left| \widehat{\text{acc}}(S_{\mathbb{P}}, S_{\mathbb{Q}}; f) - \frac{1}{2} \right|.$$

Proof. The mean embedding $\mu_{\mathbb{P}}$ under $k_f^{(S)}$ is simply $\frac{1}{2}\mathbb{E}\,\mathbb{1}(f(X) > 0) = \frac{1}{2}\Pr(f(X) > 0)$, so the MMD is

$$\frac{1}{2} \left| \Pr(f(X) > 0) - \Pr(f(Y) > 0) \right| = \left| \text{acc}(\mathbb{P}, \mathbb{Q}; f) - \frac{1}{2} \right|.$$

Moreover, $\widehat{\text{acc}}$ is acc on empirical distributions. ■

The C2ST-S, however, selects f to maximize cross-entropy (approximately maximizing $\widehat{\text{acc}}$), while we maximize $\hat{J}_\lambda$ (4.4). Although $k_f^{(S)}$ is not differentiable,

maximizing (4.3) would exactly maximize acc and hence maximize test power [Lopez-Paz and Oquab, 2017, Theorem 1].

Accessing f only through its sign allows for a simple null distribution, but it ignores f's measure of confidence: a highly confident output extremely far from the decision boundary is treated the same as a very uncertain one lying in an area of high overlap between $\mathbb{P}$ and $\mathbb{Q}$, dramatically increasing the variance of the statistic. A scheme we call C2ST-L instead tests difference in means of f on $\mathbb{P}$ and $\mathbb{Q}$ [Cheng and Cloninger, 2019]. Let

$$k_f^{(L)}(x,y) = f(x)f(y). \tag{4.7}$$

A C2ST-L is equivalent to an MMD test with $k_f^{(L)}$:

Proposition 4.3. *It holds that*

$$\mathrm{MMD}(\mathbb{P},\mathbb{Q};k_f^{(L)}) = |\mathbb{E}f(X) - \mathbb{E}f(Y)|$$
$$\widehat{\mathrm{MMD}}_b(S_\mathbb{P},S_\mathbb{Q};k_f^{(L)}) = \left|\frac{1}{n}\sum_{X_i\in S_\mathbb{P}} f(X_i) - \frac{1}{n}\sum_{Y_i\in S_\mathbb{Q}} f(Y_i)\right|.$$

Proof. This kernel's feature map is $k_f^{(L)}(x,\cdot) = f(x)$. ■

Now maximizing accuracy (or a cross-entropy proxy) no longer directly maximizes power. This kernel is differentiable, so we can directly compare the merits of maximizing (4.4) to maximizing cross-entropy; we will see in Section 4.7.2 that our more direct approach is empirically superior.

Compared to using $k_f^{(L)}$, however, Section 4.7.2 shows that learned MMD tests also obtain better performance using kernels like (4.1). This is analogous to a similar phenomenon observed in other problems by Binkowski et al. [2018] and Wenliang et al. [2019]: C2STs learn a full discriminator function on the training

set, and then apply only that function to the test set. Learning a deep kernel like (4.1) corresponds to learning only a powerful *representation* on the training set, and then *still learning* f itself from the test set – in a closed form that makes permutation testing simple.

One advantage of classifier-based methods is that they have computational cost linear in the sample size, rather than quadratic as for MMD. This problem, though, is less severe than it might appear: on a mini-batch we would use on a GPU anyway, the (quadratic) cost of $\hat{J}_\lambda$ given ϕ is typically trivial compared to the (linear) cost of computing featurizations. Estimating the MMD with mini-batches corresponds to the block estimator approach of Zaremba et al. [2013]. In our experiments (Section 4.7), there is very little difference in the runtime of our methods from C2STs.

4.5 Learning Deep Kernels

Choice of kernel architecture Most previous work on deep kernels has used a kernel κ directly on the output of a featurization network ϕ_ω, $k_\omega(x, y) = \kappa(\phi_\omega(x), \phi_\omega(y))$. This is certainly also an option for us. Any such k_ω, however, is characteristic if and only if ϕ_ω is injective. If we select our kernel well, this is not really a concern.[3] Even so, it would be reassuring to know that, even if the optimization goes awry, the resulting test will still be at least consistent. More importantly, it can be helpful in optimization to add a "safeguard" preventing the learned kernel from considering extremely far-away inputs as too similar. We

[3] A characteristic kernel on top of even $\phi_\omega(x) = \omega^\mathsf{T} x$ with a *random* ω will be almost surely consistent [Heller and Heller, 2016], and in general the existence of even one good ϕ_ω for a particular $\mathbb{P}$, $\mathbb{Q}$ pair is enough that a perfect optimizer would be able to distinguish the distributions [Arbel et al., 2018, Proposition 1].

can achieve these goals with the form (4.1), repeated here for reference:

$$k_\omega(x,y) = [(1-\epsilon)\kappa(\phi_\omega(x),\phi_\omega(y)) + \epsilon]q(x,y).$$

Here ϕ_ω is a deep network (with parameters ω) that extracts features, and κ is a kernel on those features; we use a Gaussian with lengthscale σ_ϕ, $\kappa(a,b) = \exp\left(-\frac{1}{2\sigma_\phi^2}\|a-b\|^2\right)$. We choose $0 < \epsilon < 1$ and q a Gaussian with lengthscale σ_q.

Proposition 4.4. *Let k_ω be of the form* (4.1) *with $\epsilon > 0$ and q characteristic. Then k_ω is characteristic.*

Learning the deep kernel The kernel optimization and testing procedure is summarized in Algorithm 4.1. For larger datasets, or when $n \neq m$, we use minibatches in the training procedure; for smaller datasets, we use full batches. We use the Adam optimizer [Kingma and Ba, 2015]. Note that the parameters ϵ, σ_ϕ, and σ_q are included in ω, all parameterized in log-space (i.e. we optimize ϵ' where $\epsilon = \exp(\epsilon')$).

4.6 Theoretical Analysis

We now show that optimizing the regularized test power criterion based on a finite number of samples works: as n increases, our estimates converge uniformly over a ball in parameter space, and therefore if there is a unique best kernel, we converge to it. Sutherland et al. [2017] gave no such guarantees; this result allows us to trust that, at least for reasonably large n and if our optimization process succeeds, we will find a kernel that generalizes nearly optimally rather than just overfitting to S^{tr}.

Algorithm 4.1 Testing with a learned deep kernel

Input: $S_{\mathbb{P}}$, $S_{\mathbb{Q}}$, various hyperparameters used below;
Initialize $\omega \leftarrow \omega_0$; $\lambda \leftarrow 10^{-8}$;
Split the data as $S_{\mathbb{P}} = S_{\mathbb{P}}^{tr} \cup S_{\mathbb{P}}^{te}$ and $S_{\mathbb{Q}} = S_{\mathbb{Q}}^{tr} \cup S_{\mathbb{Q}}^{te}$;
 # Phase 1: train the kernel parameters ω on $S_{\mathbb{P}}^{tr}$ and $S_{\mathbb{Q}}^{tr}$
for $T = 1, 2, \ldots, T_{max}$ **do**
  Fetch $X \leftarrow$ mini-batch from $S_{\mathbb{P}}^{tr}$;
  Fetch $Y \leftarrow$ mini-batch from $S_{\mathbb{Q}}^{te}$;
  Update $k_\omega \leftarrow$ kernel function with parameters ω; // *As in* (4.1)
  Compute $M(\omega) \leftarrow \widehat{\mathrm{MMD}}_u^2(X, Y; k_\omega)$; // *Using* (4.2)
  Compute $V_\lambda(\omega) \leftarrow \hat{\sigma}^2_{H_1,\lambda}(X, Y; k_\omega)$; // *Using* (4.5)
  Compute $\hat{J}_\lambda(\omega) \leftarrow M(\omega)/\sqrt{V_\lambda(\omega)}$; // *As in* (4.4)
  Update $\omega \leftarrow \omega + \eta \nabla_{\mathrm{Adam}} \hat{J}_\lambda(\omega)$; // *Maximize* $\hat{J}_\lambda(\omega)$
end
 # Phase 2: permutation test with k_ω on $S_{\mathbb{P}}^{te}$ and $S_{\mathbb{Q}}^{te}$
Compute $est \leftarrow \widehat{\mathrm{MMD}}_u^2(S_{\mathbb{P}}^{te}, S_{\mathbb{Q}}^{te}; k_\omega)$
for $i = 1, 2, \ldots, n_{perm}$ **do**
  Shuffle $S_{\mathbb{P}}^{te} \cup S_{\mathbb{Q}}^{te}$ into X and Y
  Compute $perm_i \leftarrow \widehat{\mathrm{MMD}}_u^2(X, Y; k_\omega)$
end
Output: k_ω, est, p-value $\frac{1}{n_{perm}} \sum_{i=1}^{n_{perm}} \mathbb{1}(perm_i \geq est)$.

Theorem 4.1 ([Liu et al., 2020a]). *Take k_ω as in Section 4.5, with ϕ_ω a fully-connected ReLU network with Λ layers and D total parameters. Let Ω be a set of kernel parameters for which $\sigma^2_{H_1}(\mathbb{P}, \mathbb{Q}; k_\omega) \geq s^2 > 0$, and the operator norms of each weight matrix and L_2 norms of each bias vector are at most R_Ω. Suppose each $x \in \mathcal{X}$ has $\|x\| \leq R_X$. Take $\lambda = n^{-1/3}$. Then, using $\tilde{O}_P$ to suppress logarithmic factors,*

$$\sup_{\omega \in \Omega} \left| \hat{J}_\lambda(S_{\mathbb{P}}, S_{\mathbb{Q}}; k_\omega) - J(\mathbb{P}, \mathbb{Q}; k_\omega) \right| = \tilde{O}_P\left(\frac{1}{s^2 n^{1/3}} \left[\frac{1}{s} + \sqrt{D} + \Lambda R_\Omega^{\Lambda - 1} \frac{R_X + 1}{\sigma_\phi} \right] \right).$$

If there is a unique best kernel ω^*, the maximizer of $\hat{J}_\lambda$ converges in probability to ω^* as $n \to \infty$.

A more general version of the result, including explicit constants and detailed

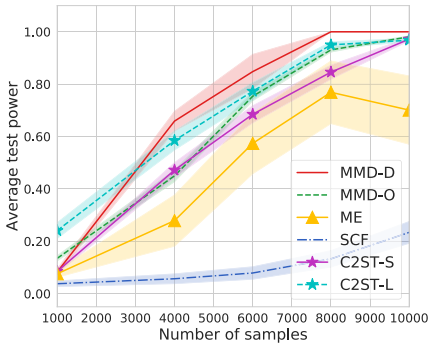
assumptions, is in Appendix A.2. Our main results (Theorem A.2 and Corollary A.2) allow for any kernel which changes smoothly with its parameterization in a Banach space, based on uniform convergence of the MMD and variance estimators using an ϵ-net argument.

Proposition A.4 establishes the result above for deep kernels.[4] Our results also apply to other kernel learning settings: Proposition A.3 gives the rate $\tilde{\mathcal{O}}_P\left(s^{-2}n^{-1/3}(s^{-1}+R_X)\right)$ for choosing the lengthscale of a single Gaussian, while Proposition A.5 gives the rate $\tilde{\mathcal{O}}_P\left(s^{-2}n^{-1/3}(s^{-1}+\sqrt{D})\right)$ for learning a linear combination of D fixed base kernels.

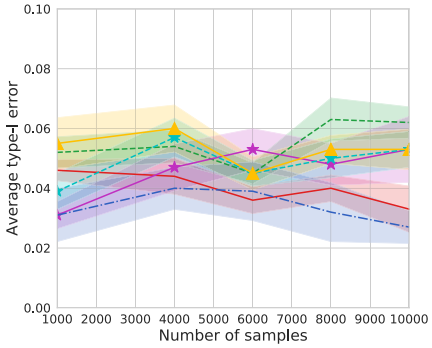
The dependence on s is somewhat unfortunate, but the ratio structure of J means that otherwise, errors in very small variances can hurt us arbitrarily. Despite this, "near-perfect" kernels (with reasonably large MMD and very small variance) will likely still be chosen as the maximizer of the regularized criterion, even if we do not estimate the (extremely large) ratio accurately. Likewise, near-constant kernels (with very small variance but still small J) will generally have their J *under*estimated, and so are unlikely to be selected when a better kernel is available. The ϵq component in (4.1) may also help avoid extremely small variances.

Given N data points, this result also gives insight into how many we should use to train the kernel and how many to test. With perfect optimization, Corollary A.4 shows a bound on the asymptotic power of the test is maximized by training on $\Theta\left(\left(N\sqrt{\log N}\right)^{\frac{3}{4}}\right)$ points, and testing on the remainder.

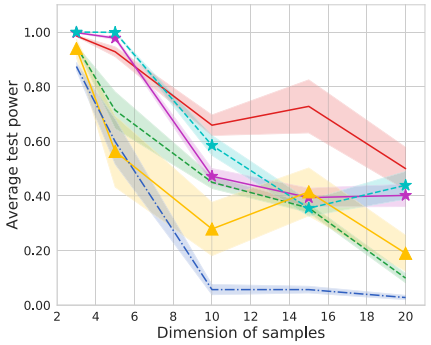
[4] Although Theorem 4.1 is for fully-connected networks, Remark A.2 establishes the same result for convolutional networks. Arbitrary Lipschitz activations also work as long as $\sigma(0)=0$.



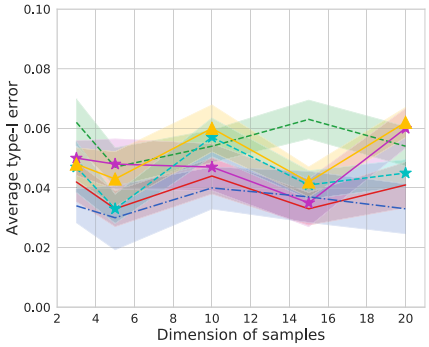
(a) Power vs. N; $d = 10$



(b) Level vs. N; $d = 10$



(c) Power vs. d; $N = 4000$



(d) Level vs. d; $N = 4000$

Figure 4.3: Results on *HDGM-S* and *HDGM-D* for $\alpha = 0.05$ (black line). Left: average test power (a) and Type I error (b) when increasing the number of samples N, keeping $d = 10$. Right: average test power (c) and Type I error (d) when increasing the dimension d, keeping $N = 4000$. Shaded regions show standard errors for the mean.

4.7 Experimental Results

In this section, we compare our method with several baselines on two simulated and three real-world datasets, including complex synthetic distributions, high-energy physics data, and challenging image data.

4.7.1 Comparison on Benchmark Datasets

Baselines. We compare the following tests in this chapter:

- MMD-D: MMD with a deep kernel; our method described in Section 4.5.
- MMD-O: MMD with a Gaussian kernel whose lengthscale is optimized as in Section 4.5. This gives better results than standard heuristics.
- Mean embedding (ME): a state-of-the-art test [Chwialkowski et al., 2015; Jitkrittum et al., 2016] based on differences in Gaussian kernel mean embeddings at a set of optimized points.
- Smooth characteristic functions (SCF): a state-of-the-art test [Chwialkowski et al., 2015; Jitkrittum et al., 2016] based on differences in Gaussian mean embeddings at a set of optimized frequencies.
- Classifier two-sample tests, including C2STS-S [Lopez-Paz and Oquab, 2017] and C2ST-L [Cheng and Cloninger, 2019] as described in Section 4.4. We set the test thresholds via permutation for both.

All of these tests are evaluated on five datasets: *Blob*, *HDGM*, *Higgs*, *MNIST* and *CIFAR-10*, where *HDGM*, *Higgs*, *MNIST* and *CIFAR-10* are high-dimension datasets.

Evaluation Procedures. For synthetic datasets, we take a single sample set for $S_{\mathbb{P}}^{tr}$ and $S_{\mathbb{Q}}^{tr}$ and learn a kernel/test locations/etc once for each method on that training set. We then evaluate its rejection rate on 100 new sample sets $S_{\mathbb{P}}^{te}$, $S_{\mathbb{Q}}^{te}$ from the same distribution. For real datasets, we select a subset of the available data for $S_{\mathbb{P}}^{tr}$ and $S_{\mathbb{Q}}^{tr}$ and train on that; we then evaluate on 100 random subsets, disjoint from the training set, of the remaining data. We repeat this full process 10 times, and report the mean rejection rate of each test. Table 4.8 shows significance tests.

Configurations. We implement all methods on Python 3.7 (Pytorch 1.1) with a NIVIDIA Titan V GPU. We run ME and SCF using the official code [Jitkrittum et al., 2016], and implement C2ST-S, C2ST-L, MMD-D and MMD-O by ourselves. We use permutation test to compute p-values of C2ST-S and C2ST-L, MMD-D, MMD-O and tests in Table 4.7. Following Sutherland et al. [2017], we set $\alpha = 0.05$ for all experiments. Following Lopez-Paz and Oquab [2017], we use a deep neural network F as the classifier in C2ST-S and C2ST-L, and train the F by minimizing cross entropy. To fairly compare MMD-D with C2ST-S and C2ST-L, the network ϕ_ω in MMD-D has the same architecture with feature extractor in F. Namely, $F = g \circ \phi_\omega$, where g is a two-layer fully-connected network. The network g is a simple binary classifier that takes extracted features (through ϕ_ω) as input. For test methods shown in Table 4.7, the network ϕ_ω in them also has the same architecture with that in MMD-D.

For *Blob*, *HDGM* and *Higgs*, ϕ_ω is a five-layer fully-connected neural network. The number of neurons in hidden and output layers of ϕ_ω are set to 50 for *Blob*, $3 \times d$ for *HDGM* and 20 for *Higgs*, where d is the dimension of samples. These neurons are with softplus activation function, i.e., $\log(1 + \exp(x))$. For *MNIST* and *CIFAR*, ϕ_ω is a *convolutional neural network* (CNN) that contains four convolutional layers and one fully-connected layer. The structure of the CNN follows the structure of the feature extractor in the discriminator of DCGAN [Radford et al., 2016] (see Figures 4.4 and 4.6 for the structure of ϕ_ω in MMD-D, and Figures 4.5 and 4.7 for the structure of classifier F in C2ST-S and C2ST-L). The link of DCGAN code is https://github.com/eriklindernoren/PyTorch-GAN/blob/master/implementations/dcgan/dcgan.py.

We use Adam optimizer [Kingma and Ba, 2015] to optimize 1) parameters of F

Figure 4.4: The structure of ϕ_ω in MMD-D on *MNIST*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout. Best viewed zoomed in.

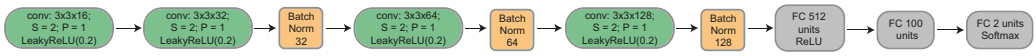


Figure 4.5: The structure of classifier F in C2ST-S and C2ST-L on *MNIST*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout. In the first layer, we will convert the *CIFAR* images from $32 \times 32 \times 3$ to $64 \times 64 \times 3$. Best viewed zoomed in.

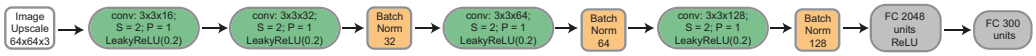


Figure 4.6: The structure of ϕ_ω in MMD-D on *CIFAR*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout in all layers. In the first layer, we will convert the *CIFAR* images from $32 \times 32 \times 3$ to $64 \times 64 \times 3$. Best viewed zoomed in.

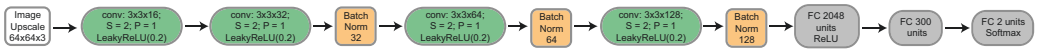


Figure 4.7: The structure of classifier F in C2ST-S and C2ST-L on *CIFAR*. The kernel size of each convolutional layer is 3; stride (S) is set to 2; padding (P) is set to 1. We do not use dropout. Best viewed zoomed in.

in C2ST-S and C2ST-L, 2) parameters of ϕ_ω in MMD-D and 3) kernel lengthscale in MMD-O. We set drop-out rate to zero when training C2ST-S, C2ST-L and MMD-D on all datasets.

Parameters Settings. Except for learning rate of Adam optimizer, we use default parameters of Adam optimizer provided by Pytorch. We use one validation set (with the same size of training set) to roughly search these parameters. Using these parameters, we compute test power of each test method on 100 test sets (with the same size of training set).

For ME and SCF, we follow Chwialkowski et al. [2015] and set $J = 10$ for *Higgs*. For other datasets, we set $J = 5$.

For C2ST-S and C2ST-L, we set batchsize to $\min\{2 \times n_b, 128\}$ for *Blob*, 128 for *HDGM* and *Higgs*, and 100 for *MNIST* and *CIFAR*. We set the number of epochs to $500 \times 18 \times n_b$/batchsize for *Blob*, 1,000 for *HDGM*, *Higgs* and *CIFAR*, and 2,000 for *MNIST*. We set learning rate to 0.001 for *Blob*, *HDGM* and *Higgs*, and 0.0002 for *MNIST* and *CIFAR* (following Radford et al. [2016]).

For MMD-O, we use full batch (i.e., all samples) to train MMD-O. we set the number of epochs to 1,000 for *Blob*, *HDGM*, *Higgs* and *CIFAR*, and 2,000 for *MNIST*. We set learning rate to 0.0005 for *Blob*, *MNIST* and *CIFAR*, and 0.001 for *HDGM*.

For MMD-D, we use full batch (i.e., all samples) to train MMD-D with samples from *Blob*, *HDGM* and *Higgs*. We use mini-batch (batchsize is 100) to train MMD-D with samples from *MNIST* and *CIFAR*. We set the number of epochs to 1,000 for *Blob*, *HDGM*, *Higgs* and *CIFAR*, and 2,000 for *MNIST*. We set learning rate to 0.0005 for *Blob* and *Higgs*, 10^{-5} for *HDGM*, 0.001 for *MNIST* and 0.0002 for and *CIFAR* (following Radford et al. [2016]).

***Blob* dataset.** *Blob-D* is the dataset shown in Figure 4.1; *Blob-S* has $\mathbb{Q}$ also equal to the distribution shown in Figure 4.1a, so that the null hypothesis holds. Details are given in Table 4.1. Results are shown in Figure 4.2. MMD-D and C2ST-L are the clear winners in power, with MMD-D better in the higher-sample regime, and MMD-D is more reliable than C2STs. Figure 4.2c shows that J is higher for MMD-D than MMD-O, in addition to the actual test power being better, as discussed in Section 4.3. All methods have expected Type I error rates.

Table 4.1: Specifications of $\mathbb{P}$ and $\mathbb{Q}$ of synthetic datasets. $\mu_1^b = [0,0], \mu_2^b = [0,1], \mu_3^b = [0,2], \ldots, \mu_8^b = [2,1], \mu_9^b = [2,2]$ (same with Figure 4.1a). $\mu_1^h = \mathbf{0}_d$, $\mu_2^h = 0.5 \times \mathbf{1}_d$, I_d is an identity matrix with size d. $\Delta_i^b = -0.02 - 0.002 \times (i-1)$ if $i < 5$ and $\Delta_i^b = 0.02 + 0.002 \times (i-6)$ if $i > 5$. if $i = 5$, $\Delta_i^b = 0$ (same with Figure 4.1a). Δ_1^h and Δ_2^h are set to 0.5 and -0.5, respectively.

Datasets	$\mathbb{P}$	$\mathbb{Q}$
Blob-S	$\sum_{i=1}^{9} \frac{1}{9} \mathcal{N}(\mu_i^b, 0.03 \times I_2)$	$\sum_{i=1}^{9} \frac{1}{9} \mathcal{N}(\mu_i^b, 0.03 \times I_2)$
Blob-D	$\sum_{i=1}^{9} \frac{1}{9} \mathcal{N}(\mu_i^b, 0.03 \times I_2)$	$\sum_{i=1}^{9} \frac{1}{9} \mathcal{N}\left(\mu_i^b, \begin{bmatrix} 0.03 & \Delta_i^b \\ \Delta_i^b & 0.03 \end{bmatrix}\right)$
HDGM-S	$\sum_{i=1}^{2} \frac{1}{2} \mathcal{N}(\mu_i^h, I_d)$	$\sum_{i=1}^{2} \frac{1}{2} \mathcal{N}(\mu_i^h, I_d)$
HDGM-D	$\sum_{i=1}^{2} \frac{1}{2} \mathcal{N}(\mu_i^h, I_d)$	$\sum_{i=1}^{2} \frac{1}{2} \mathcal{N}\left(\mu_i^h, \begin{bmatrix} 1 & \Delta_i^h & \mathbf{0}_{d-2} \\ \Delta_i^h & 1 & \mathbf{0}_{d-2} \\ \mathbf{0}_{d-2}^T & \mathbf{0}_{d-2}^T & I_{d-2} \end{bmatrix}\right)$

Table 4.2: *Higgs* ($\alpha = 0.05$): average test power±standard error for N samples. Bold represents the highest mean per row.

N	ME	SCF	C2ST-S	C2ST-L	MMD-O	MMD-D
1000	0.120±0.007	0.095±0.022	0.082±0.015	0.097±0.014	**0.132±0.005**	0.113±0.013
2000	0.165±0.019	0.130±0.026	0.183±0.032	0.232±0.017	0.291±0.012	**0.304±0.035**
3000	0.197±0.012	0.142±0.025	0.257±0.049	0.399±0.058	0.376±0.022	**0.403±0.050**
5000	0.410±0.041	0.261±0.044	0.592±0.037	0.447±0.045	0.659±0.018	**0.699±0.047**
8000	0.691±0.067	0.467±0.038	0.892±0.029	0.878±0.020	0.923±0.013	**0.952±0.024**
10000	0.786±0.041	0.603±0.066	0.974±0.007	0.985±0.005	**1.000±0.000**	**1.000±0.000**
Avg.	0.395	0.283	0.497	0.506	0.564	**0.579**

High-dimensional Gaussian mixtures. Here we study bimodal Gaussian mixtures in increasing dimension. Each distribution has two Gaussian components; in *HDGM-S*, $\mathbb{P}$ and $\mathbb{Q}$ are the same, while in *HDGM-D*, $\mathbb{P}$ and $\mathbb{Q}$ differ in the covariance of a single dimension pair but are otherwise the same. We consider both increasing N while keeping $d = 10$ and increasing d while keeping $N = 4000$, with results shown in Figure 4.3. Again, MMD-D has generally the best test power across a range of problem settings, with reasonable type I error.

Table 4.3: Results on *Higgs* ($\alpha = 0.05$). We report average Type I error on *Higgs* dataset when increasing number of samples (N). Note that, in *Higgs*, we have two types of Type I errors: 1) Type I error when two samples drawn from $\mathbb{P}$ (no Higgs bosons) and 2) Type I error when two samples drawn from $\mathbb{Q}$ (having Higgs bosons). Type I reported here is the average value of 1) and 2). Since Type I error reported here is the average value of two average Type I errors, we do not report standard errors of the average Type I error in this table.

N	ME	SCF	C2ST-S	C2ST-L	MMD-O	MMD-D
1000	0.048	0.040	0.043	0.048	0.059	0.037
2000	0.043	0.032	0.060	0.056	0.055	0.053
3000	0.049	0.043	0.046	0.053	0.051	0.069
5000	0.056	0.035	0.052	0.065	0.049	0.062
8000	0.050	0.034	0.065	0.067	0.056	0.037
10000	0.059	0.032	0.057	0.058	0.045	0.048
Avg.	0.051	0.036	0.054	0.058	0.050	0.051

Table 4.4: *MNIST* ($\alpha = 0.05$): average test power±standard error for comparing N real images to N DCGAN samples.

N	ME	SCF	C2ST-S	C2ST-L	MMD-O	MMD-D
200	0.414±0.050	0.107±0.018	0.193±0.037	0.234±0.031	0.188±0.010	**0.555±0.044**
400	0.921±0.032	0.152±0.021	0.646±0.039	0.706±0.047	0.363±0.017	**0.996±0.004**
600	**1.000±0.000**	0.294±0.008	**1.000±0.000**	0.977±0.012	0.619±0.021	**1.000±0.000**
800	**1.000±0.000**	0.317±0.017	**1.000±0.000**	**1.000±0.000**	0.797±0.015	**1.000±0.000**
1000	**1.000±0.000**	0.346±0.019	**1.000±0.000**	**1.000±0.000**	0.894±0.016	**1.000±0.000**
Avg.	0.867	0.243	0.768	0.783	0.572	**0.910**

Table 4.5: Results on *MNIST* given $\alpha = 0.05$. We report average Type I error±standard errors on real-*MNIST* vs. real-*MNIST* when increasing number of samples (N).

N	ME	SCF	C2ST-S	C2ST-L	MMD-O	MMD-D
200	0.076±0.011	0.075±0.010	0.035±0.006	0.045±0.005	0.068±0.004	0.056±0.003
400	0.062±0.010	0.056±0.007	0.044±0.006	0.040±0.004	0.053±0.005	0.056±0.005
600	0.051±0.003	0.049±0.009	0.039±0.005	0.054±0.007	0.066±0.008	0.056±0.008
800	0.054±0.006	0.046±0.006	0.043±0.005	0.042±0.007	0.051±0.005	0.054±0.007
1000	0.047±0.006	0.045±0.010	0.038±0.006	0.046±0.005	0.041±0.007	0.062±0.006
Avg.	0.058	0.054	0.040	0.045	0.056	0.057

***Higgs* dataset [Baldi et al., 2014].** We compare the jet ϕ-momenta distribution ($d = 4$) of the background process, $\mathbb{P}$, which lacks Higgs bosons, to the corresponding distribution $\mathbb{Q}$ for the process that produces Higgs bosons, following Chwialkowski et al. [2015]. *Higgs* dataset can be downloaded from UCI Machine Learning Repository. The link is https://archive.ics.uci.edu/ml/datasets/HIGGS. As discussed in these previous works, ϕ-momenta carry very little discriminating information for recognizing whether Higgs bosons were produced. We consider a series of tests with increased number of samples N.

We report average test power (comparing $\mathbb{P}$ to $\mathbb{Q}$) in Table 4.2, and average type-I error (comparing $\mathbb{P}$ to $\mathbb{P}$ or $\mathbb{Q}$ to $\mathbb{Q}$) in Table 4.3. As before, MMD-D generally performs the best; although the improvement over MMD-O here is not dramatic, MMD-D does notably outperform C2ST. All methods maintain reasonable Type I errors.

***MNIST* generative model.** The *MNIST* dataset contains 70 000 handwritten digit images [LeCun et al., 1998]. We compare true *MNIST* data samples $\mathbb{P}$ to samples $\mathbb{Q}$ from a pretrained deep convolutional generative adversarial network (DCGAN) [Radford et al., 2016]. *MNIST* dataset can be downloaded via Pytorch. See the code in https://github.com/eriklindernoren/PyTorch-GAN/blob/master/implementations/dcgan/dcgan.py.

We consider tests for increasing numbers of samples N, and report average test power (for $\mathbb{P}$ to $\mathbb{Q}$) in Table 4.4 and average Type I error ($\mathbb{P}$ to $\mathbb{P}$) in Table 4.5. MMD-D substantially outperforms its competitors in test power, with the desired Type I error. ME also does well in this case: it is perhaps particularly suited to this problem, since it is capable of identifying either modes dropped by the

generative model or spurious modes it inserts.

***CIFAR-10* vs *CIFAR-10.1*.** *CIFAR-10.1* [Recht et al., 2019] is an attempt to collect a new test set for the very popular *CIFAR-10* image classification dataset [Krizhevsky, 2009]. *CIFAR-10.1* is available from https://github.com/modestyachts/CIFAR-10.1/tree/master/datasets[5]. This new testing set contains 2,031 images from TinyImages [Torralba et al., 2008]. Normally, when evaluating a supervised model, we consider the test set an independent sample from the training distribution, ideally never-before-seen by the training algorithm. But modern computer vision model architectures and training procedures have been developed based on repeatedly evaluating on the *CIFAR-10* test set ($\mathbb{P}$), so it is possible that current models themselves are dependent on $\mathbb{P}$. *CIFAR-10.1* ($\mathbb{Q}$) is an attempt at an independent sample from this distribution, collected after the models were trained, so that they are truly independent of $\mathbb{Q}$. These models do obtain substantially lower accuracies on $\mathbb{Q}$ than on $\mathbb{P}$ – but this drop is surprisingly consistent across models, which seems unlikely to be due to the expected overfitting. The main potential explanation proposed by Recht et al. [2019] is dataset shift, but their attempt (in their Appendix C.2.8) at what amounts to a C2ST-S did not reject H_0.[6]

We train on 1000 images from each dataset and test on 1031, so that we use the entirety of *CIFAR-10.1* each time, and average over ten repetitions. These tests provide strong evidence (Table 4.6) that images in the *CIFAR-10.1* test set *are* statistically different from the *CIFAR-10* test set, with MMD-D again

[5] we use `cifar10.1_v4_data.npy`.

[6] Assuming pretrained classifiers are independent of $\mathbb{P}$, Figure 1 of Recht et al. [2019] indicates that the joint (images, labels) distribution certainly differs between *CIFAR-10* and *CIFAR-10.1*. We test here whether the marginal image distribution differs.

Table 4.6: *CIFAR-10.1* ($\alpha = 0.05$): mean rejection rates.

ME	SCF	C2ST-S	C2ST-L	MMD-O	MMD-D
0.588	0.171	0.452	0.529	0.316	**0.744**

strongest and ME still performing well. In [Recht et al., 2019], they also provide a new ImageNetV2 test set for the ImageNet dataset, with similar properties; we defer this more challenging problem to future work.

4.7.2 Ablation Study

We now study in more detail the difference between MMD-D and closely related methods. Recall from Section 4.4 that there are two main differences between MMD-D and C2STs: first, using a "full" kernel (4.1) rather than the sign-based kernel (4.6) or the intermediate linear kernel (4.7). Second, training to maximize $\hat{J}_\lambda$ (4.4) rather than a cross-entry surrogate. MMD-D uses a full kernel (4.1) trained for test power; C2ST-S effectively uses the sign kernel (4.6) trained for cross entropy.

In this section, we consider the performance of several intermediate models empirically, demonstrating that both factors help in testing. All are based on the same feature extraction architecture ϕ_ω; some models add a classification layer with new parameters w and b,

$$f_\omega(x) = w^\top \phi_\omega(x) + b,$$

which is treated as outputting classification logits. The model variants we consider are

S A kernel $\mathbb{1}(f_\omega(x) > 0)\mathbb{1}(f_\omega(y) > 0)$; corresponds to a test statistic of the accuracy of f (Proposition 4.2).

L A kernel $f_\omega(x)f_\omega(y)$; corresponds to a test statistic comparing the mean value of f (Proposition 4.3).

G A Gaussian kernel $\kappa(\phi_\omega(x), \phi_\omega(y))$.

D The deep kernel (4.1) based on ϕ_ω.

We combine these model variants with a suffix describing the optimization objective:

J Choose ω, including possibly w and b, to optimize the approximate test power (4.4).

M Choose ω, including possibly w and b, to maximize the value of the empirical MMD between two samples.[7]

C Choose ω, including w and b, to optimize cross-entropy using the classifier that specifies the probability of x belonging to $\mathbb{P}$ as $1/(1+\exp(-f_\omega(x)))$.[8]

Table 4.7 presents results for all of these methods (except for S+J, which is non-differentiable and hence difficult to optimize). Performance generally improves as we move from S to L to G to D, and from C to J.

4.7.3 Interpretability on *CIFAR-10* vs *CIFAR-10.1*

In Section 4.7.1, we have shown that images in *CIFAR-10* and *CIFAR-10.1* are not from the same distribution. Thus, it is interesting to try to understand the major difference between the datasets. Mean Embedding tests [Chwialkowski et al., 2015] compare the mean embeddings $\mu_\mathbb{P}$ and $\mu_\mathbb{Q}$ at test locations $v_1, \ldots, v_L$,

[7] If a deep kernel is unbounded, directly maximizing MMD will make optimized parameters of ϕ_ω be infinite. Thus, for L+M, we consider a normalized linear deep kernel: $\tanh(f_\omega(x)/\|S\|_\mathrm{F})\tanh(f_\omega(y)/\|S\|_\mathrm{F})$, where $S = [S_\mathbb{P}; S_\mathbb{Q}]$ and $\|\cdot\|_F$ is the Frobenius norm.

[8] G+C and D+C take the fixed ϕ_ω embeddings, then find the optimal lengthscale/etc by optimizing $\hat{J}_\lambda$.

Table 4.7: Mean test power on *Blob* ($n_b = 40$), *HDGM* ($N = 4000, d = 10$), *Higgs* ($N = 3000$) and *MNIST* ($N = 400$) for $\alpha = 0.05$. See Section 4.7.2 for the naming scheme; S+C corresponds to C2ST-S, L+C to C2ST-L, and D+J to MMD-D. L+M is the method proposed by Kirchler et al. [2019].

	S+C	L+C	G+C	D+C	L+M	G+M	D+M	L+J	G+J	D+J
Blob	0.835	0.942	0.901	0.900	0.851	0.960	0.906	0.952	0.966	**0.985**
HDGM	0.472	0.585	0.287	0.302	0.494	0.223	0.539	0.635	0.604	**0.659**
Higgs	0.257	0.399	0.353	0.384	0.321	0.254	0.379	0.295	0.364	**0.403**
MNIST	0.646	0.706	0.784	0.803	0.845	0.680	0.760	0.935	0.976	**0.996**
Avg.	0.553	0.658	0.581	0.597	0.628	0.529	0.646	0.704	0.727	**0.761**

Table 4.8: Paired t-test results ($\alpha = 0.05$) for the results of Section 4.7.1. For *HDGM*, we fix $d = 10$ (corresponding to Figure 4.3a). ✓ indicates MMD-D achieved statistically significantly higher mean test power than the other method, × that it did not.

Dataset	ME	SCF	C2ST-S	C2ST-L	MMD-O
Blob	✓	✓	✓	×	×
HDGM	✓	✓	✓	✓	✓
Higgs	✓	✓	✓	×	×
MNIST	✓	✓	✓	✓	✓

rather than through their overall norm. The test statistic is

$$S_n = \frac{1}{n-1}\sum_{i=1}^{n}(z_i - \bar{z}_n)(z_i - \bar{z}_n)^\mathsf{T},$$

where

$$\hat{\Lambda} = n\bar{z}_n^\mathsf{T} S^{-1}\bar{z}_n, \quad z_i = (k(x_i, v_j) - k(y_i, v_j))_{j=1}^{L} \in \mathbb{R}^L, \quad \bar{z}_n = \frac{1}{n}\sum_{i=1}^{n} z_i;$$

the asymptotic null distribution of $\hat{\Lambda}$ is χ^2_L, and the estimator is computable in linear time rather than $\widehat{\mathrm{MMD}}_U$'s quadratic time.

In Jitkrittum et al. [2017], they jointly learn the parameters v_j and kernel parameters to optimize test power. The best such test locations ($L = 1$) for a Gaussian kernel (with learned bandwidth) are shown in Figure 4.8. We could also try optimizing a deep kernel (4.1) and the test locations together; this

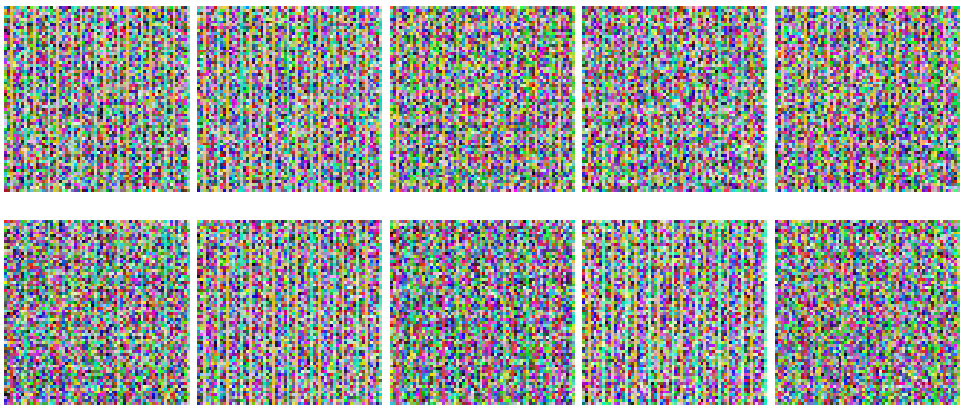


Figure 4.8: The best test locations (learned by an ME test with $L = 1$) from 10 experiments on *CIFAR-10* vs *CIFAR-10.1*. Average rejection rate is 0.415.

procedure, however, failed to find a useful test. We can find a better test, though, with a two-stage scheme: first, learn a deep kernel to maximize $\hat{J}_\lambda$, then choose v_i to maximize $\hat{\Lambda}$ with that kernel fixed. Results are shown in Figure 4.9.

Although these approaches give nontrivial test power, it is hard to interpret either set of images, as the test locations have moved far outside the set of natural images. We can instead constrain $v_1 \in S_{\mathbb{P}} \cup S_{\mathbb{Q}}$, simply picking the single point from the dataset which maximizes $\hat{\Lambda}$ (shown in Figure 4.10). This achieves similar test power, but lets us see that the difference might lie in images with smaller objects of interest than the mean for *CIFAR-10*.

4.8 Summary

The test power of MMD is limited by simple kernels (e.g., Gaussian kernel) when facing complex-structured distributions, but we can address this problem by proposing richer *deep kernels*. This chapter shows that optimizing the parameters of these kernels to maximize the test power, as proposed by Sutherland et al.

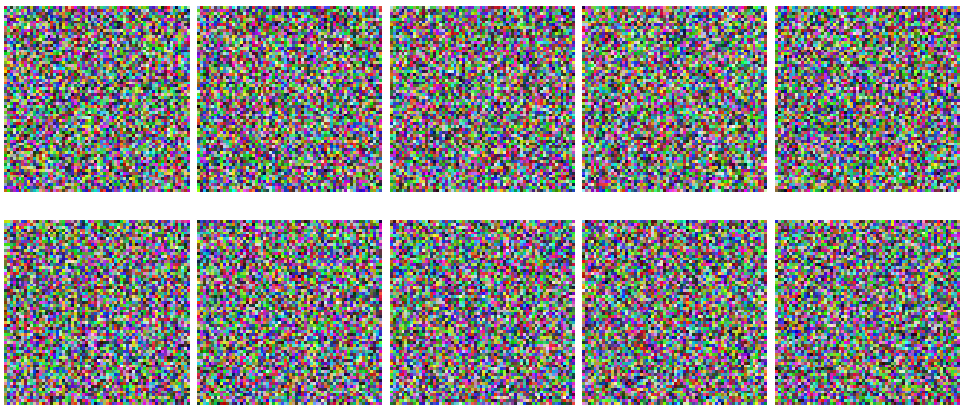


Figure 4.9: The best test locations (learned by an ME test, $L = 1$, with a deep kernel optimized for an MMD test) from 10 experiments on *CIFAR-10* vs *CIFAR-10.1*. Average rejection rate is 0.637.

Figure 4.10: The best test locations (selected among existing images with our learned deep kernel, $L = 1$) from 10 experiments on *CIFAR-10* vs *CIFAR-10.1*. Average rejection rate is 0.653.

[2017], outperforms state-of-the-art alternatives even when considering large, deep kernels with hundreds of thousands of parameters, rather than the simple shallow kernels they considered. This chapter provides theoretical guarantees that this process is reasonable to conduct on finite samples, and asymptotically selects the most powerful kernel. This chapter also gives deeper insight into the relationship between this approach and classifier two-sample tests [Lopez-Paz and Oquab, 2017], explaining why this approach outperforms that one.

CHAPTER 5

BUTTERFLY: A ONE-STEP APPROACH TOWARDS WILDLY UNSUPERVISED DOMAIN ADAPTATION

5.1 Introduction

The *homogeneous unsupervised domain adaptation* (HoUDA) methods train with clean labeled data in source domain (i.e., clean source data) and unlabeled data in target domain (i.e., unlabeled target data) to obtain classifiers for the *target domain* (TD), which mainly consist of three orthogonal techniques: *integral probability metrics* (IPM) [Yu et al., 2017], *adversarial training* [Gong et al., 2018; Saito et al., 2018] and *pseudo labeling* [Saito et al., 2017]. Compared to IPM- and adversarial-training-based methods, the pseudo-labeling-based method (i.e., *asymmetric tri-training domain adaptation* (ATDA) [Saito et al., 2017]) can construct a high-quality target-specific representation, providing a better classification performance.

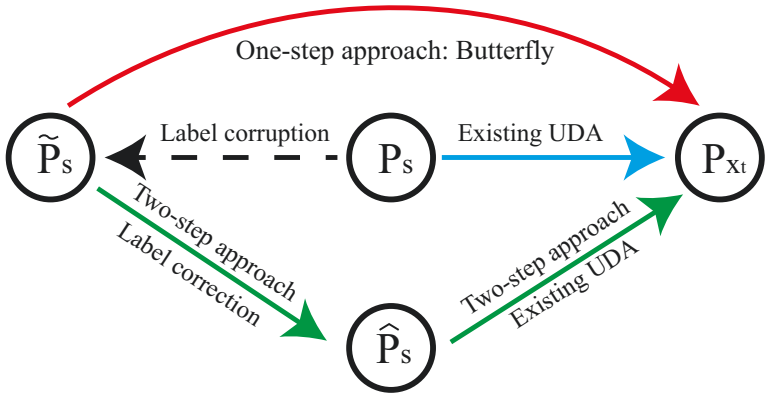
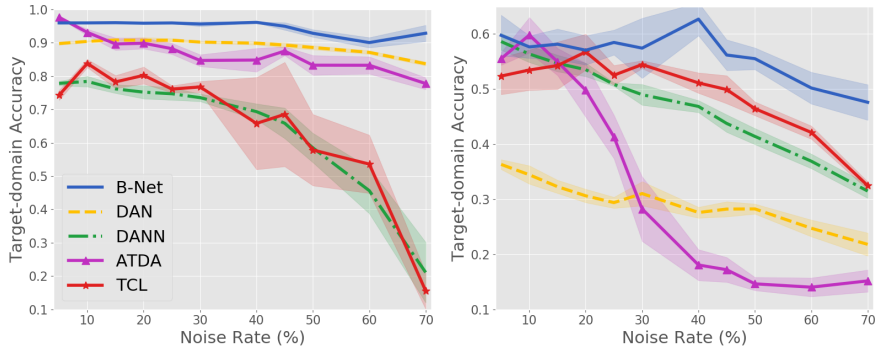
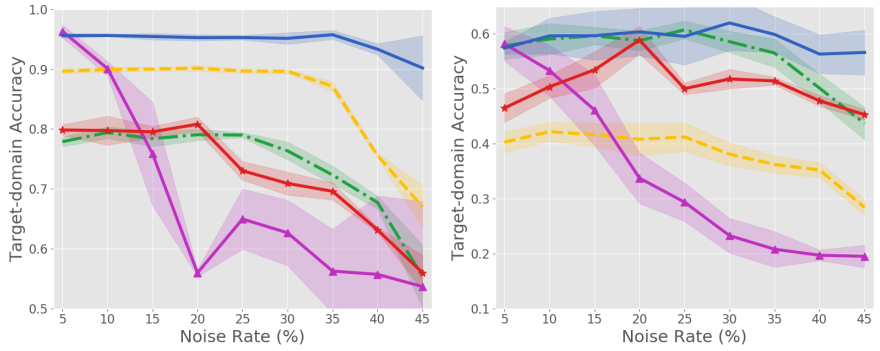


Figure 5.1: Wildly unsupervised domain adaptation (WUDA). The blue line denotes that HoUDA transfers knowledge from clean source data (P_s) to unlabeled target data (P_{x_t}). However, perfectly clean data is hard to acquire. This brings *wildly unsupervised domain adaptation* (WUDA), namely transferring knowledge from noisy source data ($\widetilde{P}_s$) to unlabeled target data (P_{x_t}). Note that label corruption process (black dash line) is unknown in practice. To handle WUDA, a compromise solution is a two-step approach (green line), which sequentially combines label-noise algorithms ($\widetilde{P}_s \to \hat{P}_s$, label correction) and existing HoUDA ($\hat{P}_s \to P_{x_t}$). This chapter proposes a robust one-step approach called Butterfly (red line, $\widetilde{P}_s \to P_{x_t}$ directly), which eliminates noise effects from $\widetilde{P}_s$.

However, in the wild, the data volume of source domain tends to be large [Tan et al., 2014]. To avoid the expensive labeling cost, labeled data in source domain normally come from amateur annotators or the Internet [Lee et al., 2018; Schroff et al., 2011; Tommasi and Tuytelaars, 2014]. This brings us a new, more realistic and more challenging problem, *wildly unsupervised domain adaptation* (abbreviated as WUDA, Figure 5.1). This adaptation aims to transfer knowledge from noisy labeled data in source domain ($\widetilde{P}_s$, i.e., noisy source data) to unlabeled target data (P_{x_t}). Unfortunately, existing HoUDA methods share an implicit assumption that *there are no noisy source data* [Shu et al., 2019; Yu et al., 2017]. Namely, these methods focus on transferring knowledge from clean source data (P_s) to unlabeled target data (P_{x_t}). Therefore, these methods cannot well handle



(a) Symmetry-flip noise: $S \rightarrow M$ (left), $M \rightarrow S$ (right)



(b) Pair-flip noise: $S \rightarrow M$ (left), $M \rightarrow S$ (right)

Figure 5.2: WUDA ruins representative HoUDA methods. Representative HoUDA methods includes *deep adaptation network* (DAN, a IPM based method [Long et al., 2015]), *domain-adversarial neural network* (DANN, a adversarial training based method [Ganin et al., 2016a]), *asymmetric tri-training domain adaptation* (ATDA, a pseudo-label based method [Saito et al., 2017]) and *transferable curriculum learning* (TCL, a robust HoUDA method [Shu et al., 2019]). B-Net is our proposed WUDA method. We report target-domain accuracy of all methods when the noise rate of source domain changes (a) from 5% to 70% (symmetry-flip noise) and (b) from 5% to 45% (pair-flip noise). Clearly, when the noise rate of source domain increases, target-domain accuracy of representative HoUDA methods drops quickly while that of B-Net keeps stable consistently.

WUDA.

To validate this fact, we empirically reveal the deficiency of existing HoUDA methods (Figure 5.2, e.g., *deep adaptation network* (DAN) [Long et al., 2015] and *domain-adversarial neural network* (DANN) [Ganin et al., 2016a]). To improve these methods, a straightforward solution is a two-step approach (shown in Figure 5.1). Namely, we can first use label-noise algorithms to train a classifier on noisy source data, then leverage this trained classifier to assign pseudo labels for noisy source data. Via HoUDA methods, we can transfer knowledge from pseudo-labeled source data ($\hat{P}_s$) to unlabeled target data (P_{x_t}). Nonetheless, pseudo-labeled source data are still noisy, and such two-step approach may not eliminate noise effects.

To circumvent the issue of the two-step approach (see green lines in Figure 5.1), we present a robust one-step approach called *Butterfly*. In high level, Butterfly directly transfers knowledge from $\widetilde{P}_s$ to P_{x_t}, and uses the transferred knowledge to construct target-specific representations. In low level, Butterfly maintains four networks dividing two branches (Figure 5.3): Two networks in Branch-I are jointly trained on noisy source data and pseudo-labeled target data (data in *mixture domain* (MD)); while two networks in Branch-II are trained on pseudo-labeled target data. Our ablation study (see Section 5.7.8) confirms the network design of Butterfly (see Section 5.3) is the optimal.

The reason why Butterfly can be robust takes root in the *dual-checking principle* (DCP): Butterfly checks high-correctness data out, from not only the data in MD but also the pseudo-labeled target data. After cross-propagating these high-correctness data, Butterfly can obtain high-quality *domain-invariant representations* (DIR) and *target-specific representations* (TSR) simultaneously in

an iterative manner. If we only check data in MD (i.e., B-Net-M in Section 5.7.8), the error existed in pseudo-labeled target data will accumulate, leading to low-quality DIR and TSR.

We conduct experiments on simulated WUDA tasks, including 4 *MNIST-to-SYND* tasks, 4 *SYND-to-MNIST* tasks and 24 human-sentiment tasks. Besides, we conduct experiments on 3 real-world WUDA tasks. Empirical results demonstrate that Butterfly can robustly transfer knowledge from noisy source data to unlabeled target data. Meanwhile, Butterfly performs much better than existing HoUDA methods when *source domain* (SD) suffers the extreme (e.g., 45%) noise.

5.2 Wildly Unsupervised Domain Adaptation

In this section, we first define a new, more realistic and more challenging problem setting called *wildly unsupervised domain adaptation (WUDA)*, and explain the nature of WUDA. Then, we empirically show that representative HoUDA methods cannot handle WUDA well, which motivates us to propose Butterfly (see Section 5.3).

Definition 5.1 (Wildly Unsupervised Domain Adaptation)**.** *Let X_t be a multivariate random variable defined on the space $\mathcal{X}$ with respective a probability density p_{x_t}, (X_s, Y_s) be a multivariate random variable defined on the space $\mathcal{X} \times \mathcal{Y}$ with respective a probability density $\tilde{p}_s$, where $\mathcal{Y} = \{1, \ldots, K\}$. Let p_{x_s} be the marginal density of $\tilde{p}_s$. Given i.i.d. data $\tilde{D}_s = \{(x_{si}, \tilde{y}_{si})\}_{i=1}^{n_s}$ and $D_t = \{x_{ti}\}_{i=1}^{n_t}$ drawn from $\tilde{P}_s$ and P_{x_t}, in wildly unsupervised domain adaptation, we aim to train with $\tilde{D}_s$ and D_t to accurately annotate data drawn from P_{x_t}, where $p_{x_s} \neq p_{x_t}$.*

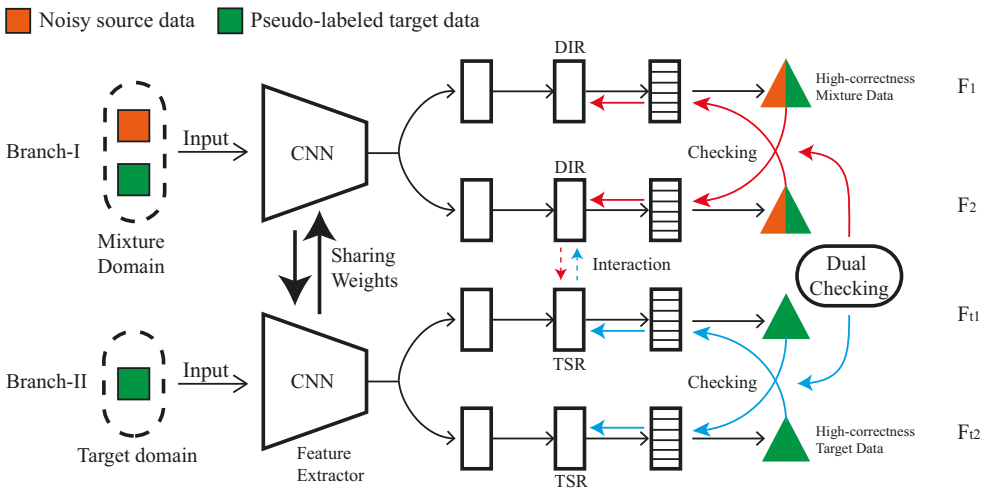


Figure 5.3: Butterfly Framework. Two networks (F_1 and F_2) in Branch-I are jointly trained on noisy source data and pseudo-labeled target data (mixture domain). Two networks in Branch-II (F_{t1} and F_{t2}) are trained on pseudo-labeled target data. By using dual-checking principle, Butterfly checks high-correctness data out from both mixture and pseudo-labeled target data. After cross-propagating checked data, Butterfly can obtain high-quality *domain-invariant representations* (DIR) and *target-specific representations* (TSR) simultaneously in an iterative manner. Note that the interaction between DIR and TSR happens via the shared CNN. Besides, in the first training epoch, since we do not have any pseudo-labeled target data, we need to use noisy source data as the pseudo-labeled target data, which follows [Saito et al., 2017].

Remark 5.1. In Definition 5.1, $\tilde{D}_s$ is noisy source data, D_t is unlabeled target data, and $\tilde{P}_s$ and P_{x_t} are two probability measures corresponding to densities $\tilde{p}_s(x_s, \tilde{y}_s)$ and $p_{x_t}(x_t)$.

5.2.1 Nature of WUDA

Specifically, there are five distributions involved in WUDA setting: 1) a marginal distribution on source data, i.e., p_{x_s} in Definition 5.1; 2) a marginal distribution on target data, i.e., p_{x_t} in Definition 5.1; 3) an incorrect conditional distribution

of label given x_s, $q(y_s|x_s)$; 4) a correct conditional distribution of label given x_s, $p(y_s|x_s)$ and 5) a correct conditional distribution of label given x_t, $p(y_t|x_t)$.

Based on Definition 5.1 and Appendix A.3.1.2, noisy source data $\tilde{D}_s$ are drawn from $\tilde{p}_s(x_s, y_s) = p_{x_s}(x_s)((1-\rho)p(y_s|x_s) + \rho q(y_s|x_s))$, where ρ is the noise rate in source data. Namely, source data $\tilde{D}_s$ are mixture of correct source data from $p_{x_s}(x_s)p(y_s|x_s)$ and incorrect data from $p_{x_s}(x_s)q(y_s|x_s)$. Target data D_t are drawn from p_{x_t}. In WUDA setting, we aim to train a classifier with $\tilde{D}_s$ and D_t. This classifier is expected to accurately annotate data from p_{x_t}, i.e., to accurately simulate distribution 5).

5.2.2 WUDA ruins HoUDA Methods

We take a simple example to illustrate the phenomenon that WUDA ruins representative HoUDA methods. In Section 5.5.1, we theoretically analyze the reason of this phenomenon.

We corrupt source data using symmetry flipping [Patrini et al., 2017] and pair flipping [Han et al., 2018] (see Eq. (5.18) and Eq. (5.19)). Namely, the corrupted source data ($\tilde{D}_s$ in Definition 5.1) are drawn from $\tilde{P}_s$ whose probability density is $\tilde{p}_s(x_s, y_s) = p_{x_s}(x_s)((1-\rho)p(y_s|x_s) + \rho q(y_s|x_s))$. We draw the target data D_t from P_{x_t} whose probability density is p_{x_t}. To instantiate source and target data, we leverage *MNIST* and *SYND* (see Figure 5.4), respectively.

Thus, we first construct two WUDA tasks with symmetry-flip noise: corrupted *SYND*→*MNIST* (*S*→*M*) and corrupted *MNIST*→*SYND* (*M*→*S*). In Figure 5.2 (a), we report accuracy of representative HoUDA methods on unlabeled target data, when the noise rate ρ of SD changes from 5% to 70%. It is clear that target-

domain accuracy of these representative HoUDA methods drops quickly when ρ increases. This means that WUDA ruins representative HoUDA methods. Then, we construct another two WUDA tasks with pair-flip noise. In Figure 5.2 (b), we report target-domain accuracy, when the noise rate ρ of SD changes from 5% to 45%. Again, WUDA still ruins representative HoUDA methods. Note that, in practice, pair-flip noise is much harder than symmetry-flip noise, the noise rate of pair-flip noise cannot be over 50% [Han et al., 2018].

However, the proposed Butterfly network (abbreviated as B-Net, Figure 5.3) performs robustly when ρ increases (blue lines in Figure 5.2). In Section 5.3, we will introduce Butterfly framework, and explain why Butterfly achieves better target-domain accuracy consistently in Section 5.4 and Section 5.5.

5.3 Butterfly: Towards Robust One-step Approach

To realize a robust WUDA approach, we propose a Butterfly framework (a one-step approach, Algorithm 5.1), which trains four networks dividing into two branches (Figure 5.3). By using DCP, Branch-I checks which data is correct in MD; while Branch-II checks which pseudo-labeled target data is correct. To ensure these checked data highly-correct, we apply the small-loss trick based on memorization effects of deep learning [Arpit et al., 2017]. After cross-propagating these checked data [Bengio, 2014], Butterfly can obtain high-quality DIR and TSR simultaneously in an iterative manner.

5.3.1 Loss Function in Butterfly

Four networks trained by Butterfly share the same loss function but with different inputs.

$$\mathcal{L}(\theta, \mathbf{u}; F, D) = \frac{1}{\sum_{i=1}^{n} u_i} \sum_{i=1}^{n} u_i \ell(F(x_i), \check{y}_i), \tag{5.1}$$

where n is the batch size (i.e., $n = |D|$), and F represents a network (e.g., F_1, F_2, F_{t1} and F_{t2}). $D = \{(x_i, \check{y}_i)\}_{i=1}^{n}$ is a mini-batch for training a network, where $\{x_i, \check{y}_i\}_{i=1}^{n}$ could be data in MD or TD (Figure 5.3), and θ represents parameters of F and $\mathbf{u} = [u_1, ..., u_n]^T$ is an n-by-1 vector whose elements equal 0 or 1. For two networks in Branch-I, following [Saito et al., 2017], we also add a regularizer $|\theta_{f11}^T \theta_{f21}|$ in their loss functions, where θ_{f11} and θ_{f21} are weights of the first fully-connect layer of F_1 and F_2. With this regularizer, F_1 and F_2 will learn from different features.

Nature of the loss $\mathcal{L}$. In loss function $\mathcal{L}$, we have n instances: $\{(x_i, \check{y}_i)\}_{i=1}^{n}$. For the i^{th} instance, we will compute its cross-entropy loss (i.e., $\ell(F(x_i), \check{y}_i)$), and we will denote this instance as "selected" if $u_i = 1$. Thus, the nature of $\mathcal{L}$ is actually the average value of cross-entropy loss of these "selected" instances. Note that, we need to set a constrain to prevent $\sum_{i=1}^{n} u_i = 0$ in $\mathcal{L}$, which means that we should select at least one instance to compute $\mathcal{L}$.

5.3.2 Training Procedure of Butterfly

For two networks in each branch, they will first check high-correctness data out and then cross update their parameters using these data.

Based on loss function defined in Eq. (5.1), entire training procedures of Butterfly are shown in Algorithm 5.1. First, we initialize training data for two

branches ($\tilde{D}$ for Branch-I and $\tilde{D}_t^l$ for Branch-II), four networks (F_1,F_2,F_{t1} and F_{t2}) and the number of pseudo labels (line 2). In the first epoch ($T = 1$), following [Saito et al., 2017], $\tilde{D}_t^l$ is the same with $\tilde{D}_s$ (i.e., we use noisy source data as pseudo-labeled target data), since we cannot annotate pseudo labels for target data when $T = 1$. After mini-batch $\check{D}$ is fetched from $\tilde{D}$ (line 4), F_1 and F_2 check high-correctness data out and update their parameters (lines 5) using Algorithm 5.2. Using similar procedures, F_{t1} and F_{t2} also update their parameters using Algorithm 5.2 (lines 6-7).

In each epoch, after N_{max} mini-batch updating, we randomly select n_t^l unlabeled target data and assign them pseudo labels using F_1 and F_2 (lines 8). Following [Saito et al., 2017], the Labeling function in Algorithm 5.1 (line 8) assigns pseudo labels for unlabeled target data, when predictions of F_1 and F_2 agree and at least one of them is confident about their predictions (probability above 0.9 or 0.95). Using this function, we can obtain the pseudo-labeled target data $\tilde{D}_t^l$ for training Branch-II in the next epoch. Then, we merge $\tilde{D}_t^l$ and $\tilde{D}_s$ to be $\tilde{D}$ for training Branch-I in the next epoch (line 9). Finally, we update n_t^l, $R(T)$ and $R_t(T)$ in lines 10-11 according to [Saito et al., 2017] and [Han et al., 2018]. Note that $R(T)$ is a piecewise-defined linear function. Namely, when $T \geq T_k$, $R(T) = 1 - \tau$; when $T \leq T_k$, $R(T) = 1 - T/T_k \times \tau$.

In Algorithm 5.1, we use τ to represent the noise rate (i.e., the ratio of data with incorrect labels) in MD and use τ_t to represent the noise rate in TD. However, in WUDA, we cannot obtain the ground-truth τ and τ_t. Thus, we regard τ and τ_t as hyper-parameters.

Checking process in Butterfly. In Algorithm 5.2, we first obtain four inputs: 1) networks F_1 and F_2, and 2) a mini-batch D, and 3) learning rate η

and 4) remember rate α (line 1). Then, we will obtain the best $\mathbf{u}_1$ by solving a minimization problem (line 2). $\mathcal{L}$ represents the loss function defined in Eq. (5.1). θ_1 represents the parameters of the network F_1. Similarly we will obtain the best $\mathbf{u}_2$ (line 3). θ_2 represents the parameters of the network F_2. Next, θ_1 and θ_2 are updated using gradient descent, where the gradients are computed using a given optimizer (lines 4-5). Finally, we substitute the updated θ_1 into F_1 and the updated θ_2 into F_2 and output F_1 and F_2 (line 6).

Solution to minimization problems in Algorithm 5.2. In line 2 or 3 in Algorithm 5.2, we need to solve a minimization problem: $\min_{\mathbf{u}':\mathbf{1u}'>\alpha|D|} \mathcal{L}(\theta, \mathbf{u}'; F, D)$ and return the best $\mathbf{u}'$ as $\mathbf{u}$ ($\mathbf{u}_1$ in line 2 and $\mathbf{u}_2$ in line 3). In this paragraph, we will show how to quickly solve this problem using a sorting algorithm. Recall the nature of the loss $\mathcal{L}$, we know $\mathcal{L}$ is the average value of cross-entropy losses of "selected" instances, and $\mathbf{1u}'$ is the number of these "selected" instances. Thus this minimization problem is equivalent to "given a fixed F (F_1 or F_2) and n instances in D, how to select at least k instances such that $\mathcal{L}$ is minimized", where $k = \lceil \alpha|D| \rceil$. To solve this problem, we first use a sorting algorithm (top_k function in TensorFlow) to sort these n instances according to their cross-entropy losses $\ell(F_1(x_i), \check{y}_i)$. Then, we select k instance with the smallest cross-entropy losses. Finally, let u_i of these k instances be 1 and u_i of the other instances be 0, and we can get the best $\mathbf{u} = [u_1, \ldots, u_n]$. The average value of cross-entropy losses of these k instances is the minimized value of $\mathcal{L}(\theta, \mathbf{u}'; F, D)$ under the constrain $\mathbf{1u}' > \alpha|D|$.

Algorithm 5.1 Butterfly Framework: quadruple training for WUDA problem

1: Input $\tilde{D}_s$, D_t, learning rate η, fixed τ, fixed τ_t, epoch T_k and T_{max}, iteration N_{max}, # of pseudo-labeled target data n_{init}, max of n_{init} $n^l_{t,max}$;
2: Initial F_1, F_2, F_{t1}, F_{t2}, $\tilde{D}^l_t = \tilde{D}_s$, $\tilde{D} = \tilde{D}_s$, $n^l_t = n_{init}$;
for $T = 1, 2, \ldots, T_{\max}$ **do**
  3: Shuffle training set $\tilde{D}$; // Noisy dataset
  for $N = 1, \ldots, N_{max}$ **do**
    4: Fetch mini-batch $\check{D}$ from $\tilde{D}$;
    5: Update Branch-I: F_1, F_2 = Checking($F_1, F_2, \check{D}, \eta, R(T)$); // Check data in MD using Algorithm 5.2
    6: Fetch mini-batch $\check{D}_t$ from $\tilde{D}^l_t$;
    7: Update Branch-II: F_{t1}, F_{t2} = Checking($F_{t1}, F_{t2}, \check{D}_t, \eta, R_t(T)$); // Check data in TD using Algorithm 5.2
  end
  8: Obtain $\tilde{D}^l_t$ = Labeling(F_1, F_2, D_t, n^l_t); // Label D_t, following [Saito et al., 2017]
  9: Obtain $\tilde{D} = \tilde{D}_s \cup \tilde{D}^l_t$; // Update MD
  10: Update $n^l_t = \min\{T/20 * n_t, n^l_{t,max}\}$;
  11: Update $R(T) = 1 - \min\{\frac{T}{T_k}\tau, \tau\}$, $R_t(T) = 1 - \min\{\frac{T}{T_k}\tau_t, \tau_t\}$;
end
12: Output F_{t1} and F_{t2}

Algorithm 5.2 Checking(F_1, F_2, D, η, α)

1: Input networks F_1, F_2, mini-batch D, learning rate η, remember rate α;
2: Obtain $\mathbf{u}_1 = \arg\min_{\mathbf{u}'_1 : \mathbf{1}\mathbf{u}'_1 > \alpha|D|} \mathcal{L}(\theta_1, \mathbf{u}'_1; F_1, D)$; // Check high-correctness data
3: Obtain $\mathbf{u}_2 = \arg\min_{\mathbf{u}'_2 : \mathbf{1}\mathbf{u}'_2 > \alpha|D|} \mathcal{L}(\theta_2, \mathbf{u}'_2; F_2, D)$; // Check high-correctness data
4: Update $\theta_1 = \theta_1 - \eta\nabla\mathcal{L}(\theta_1, \mathbf{u}_2; F_1, D)$; // Update θ_1
5: Update $\theta_2 = \theta_2 - \eta\nabla\mathcal{L}(\theta_2, \mathbf{u}_1; F_2, D)$; // Update θ_2
6: Output F_1 **and** F_2

5.4 Butterfly vs. Two-step Approach

In this section, we analyzes why Butterfly is better than two-step approach using theoretical results in Section 5.5. Following [Ben-David et al., 2010b], we derive an upper bound of target-domain risk for WUDA (see Theorem 5.2). Compared to existing HoUDA bounds, the WUDA bound has two extra terms (see Eq. (5.4)): Δ_s (noise effect from source data), and Δ_t (noise effects from pseudo-labeled target data). We will use Δ_s and Δ_t to show why Butterfly (a one-step approach) can

eliminate noise effects while two-step methods cannot.

5.4.1 Two-step Approach (A Compromise Solution)

To reduce noise effects, a straightforward solution is two-step approach. In the first step, we can train a classifier with noisy source data using Co-teaching [Han et al., 2018] and use this classifier to annotate pseudo labels for source data. In the second step, we use existing HoUDA methods (e.g., ATDA [Saito et al., 2017]) to train a target-domain classifier with pseudo-labeled-source data and pseudo-labeled target data.

However, two-step approach may not reduce noise effects Δ_s (i.e., not alleviating noise effects from source data). In two-step approach, after using Co-teaching, Δ_s will become pseudo-labeled-source effects Δ'_s (see Eq. (5.5)). The first part of Δ'_s may be less than that of Δ_s due to Co-teaching, but the second term of Δ'_s may be higher than that of Δ_s since Co-teaching does not consider to minimize it. Thus, it is hard to say whether $\Delta'_s < \Delta_s$. This means that, the two-step approach may not really reduce noise effects Δ_s. Besides, two-step approach does not take care of eliminating Δ_t explicitly. Thus, we can find that a two-step approach cannot eliminate Δ_s and Δ_t. Our empirical study also confirms the deficiency of two-step approach (see Section 5.7).

5.4.2 One-step Approach (Butterfly)

To eliminate noise effects $\Delta = \Delta_s + \Delta_t$, Butterfly aims to select correct data simultaneously from noisy source data and pseudo-labeled target data (see Section 5.3). Let ρ^s_{01} be the probability that incorrect data is selected from noisy

source data, and ρ_{01}^t be the probability that incorrect data is selected from pseudo-labeled target data. Theorem 5.3 shows that $\Delta \to 0$ if $\rho_{01}^s \to 0$ and $\rho_{01}^t \to 0$. Since Butterfly can select correct data with a high probability (i.e., $\rho_{01}^s \to 0$ and $\rho_{01}^t \to 0$), noise effects will be eliminated ($\Delta \to 0$).

5.5 Theoretical Analysis

This section presents theoretical findings related to WUDA problem and Butterfly[1]. Practitioner may safely skip it. We use the following notations in this section:

- a space $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} = \{1, 2, \ldots, K\}$ as a label set;
- $f_t(x_t)$ and $\tilde{f}_t(x_t)$ represent the ground-truth and pseudo labeling function of the target domain, where $f_t, \tilde{f}_t : \mathcal{X} \to \mathcal{Y}$;
- $A = \{x : \tilde{f}_t(x) \neq f_t(x)\}$ and $B = \mathcal{X}/A$ represent the area where $\tilde{f}_t(x) \neq f_t(x)$ (the set A) and the area where $\tilde{f}_t(x) = f_t(x)$ (the set B);
- $\tilde{p}_s(x_s, \tilde{y}_s)$, $p_s(x_s, y_s)$ and $q_s(x_s, y_s)$ represent probability densities of noisy, correct and incorrect *multivariate random variables* (m.r.v.) defined on $\mathcal{X} \times \mathcal{Y}$, respectively[2], and $\tilde{p}_{x_s}(x_s)$, $p_{x_s}(x_s)$ and $q_{x_s}(x_s)$ are their marginal densities;
- $p_{x_t}(x_t)$ represents the probability density of m.r.v. X_t defined on $\mathcal{X}$;

[1]Please note that, all proofs are demonstrated in Appendix A.3.2.

[2]There are two common ways to express the probability density of noisy m.r.v. (see Appendix A.3.1). One way is to use a mixture of densities of correct and incorrect m.r.v..

- $q_{x_t}(x) = p_{x_t}(x)1_A(x)/P_{x_t}(A)$ represents the probability density of X_t restricted in A;
- $p'_{x_t}(x_t) = p_{x_t}(x_t)1_B(x_t)/P_{x_t}(B)$ represents the probability density of X_t restricted in B;
- $\mathcal{H}$ is the class of arbitrary *decision functions* $h : \mathcal{X} \to \mathbb{R}^K$;
- $\ell : \mathbb{R}^K \times \mathcal{Y} \to \mathbb{R}_+$ is the loss function. $\ell(t, y)$ means the loss incurred by predicting an output t (e.g., $h(x)$) when the ground truth is y;
- $\mathbb{L}_{\mathcal{H}} = \{\ell(h(x), y) | h \in \mathcal{H}, x \in \mathcal{X}, y \in \mathcal{Y}\}$ is the class of loss functions associated with $\mathcal{H}$;
- expected risks on the noisy m.r.v. and correct m.r.v.:

$$\tilde{R}_s(h) = \mathbb{E}_{\tilde{p}_s(x_s, \tilde{y}_s)}[\ell(h(x_s), \tilde{y}_s)],$$
$$R_s(h) = \mathbb{E}_{p_s(x_s, y_s)}[\ell(h(x_s), y_s)];$$

- expected discrepancy (associated with ℓ) between an arbitrary decision function h and a ground-truth or pseudo labeling function f (f could be f_t or $\tilde{f}_t$) under different marginal densities:

$$\tilde{R}_s(h, f) = \mathbb{E}_{\tilde{p}_{x_s}(x_s)}[\ell(h(x_s), f(x_s))],$$
$$R_s(h, f) = \mathbb{E}_{p_{x_s}(x_s)}[\ell(h(x_s), f(x_s))],$$
$$R_t(h, f) = \mathbb{E}_{p_{x_t}(x_t)}[\ell(h(x_t), f(x_t))].$$

5.5.1 WUDA ruins HoUDA Methods

Theoretically, we show that existing HoUDA methods cannot transfer useful knowledge from $\tilde{D}_s$ to D_t directly. We first present the relation between $R_s(h)$ and $\tilde{R}_s(h)$.

Theorem 5.1. *If $\tilde{p}_s(x_s, \tilde{y}_s)$ is generated by a transition matrix Q as demonstrated in Appendix A.3.1.1, we have*

$$\tilde{R}_s(h) = R_s(h) + \mathbb{E}_{p_{x_s}(x_s)}[\eta^T(x_s)(Q - I)\ell(h(x_s))], \tag{5.2}$$

where $\ell(h(x_s)) = [\ell(h(x_s), 1), ..., \ell(h(x_s), K)]^T$ and $\eta(x_s) = [p_{Y_s|X_s}(1|x_s), ..., p_{Y_s|X_s}(K|x_s)]^T$. If $\tilde{p}_s(x_s, \tilde{y}_s)$ is generated by sample selection as described in in Appendix A.3.1.2, we have

$$\tilde{R}_s(h) = (1 - \rho)R_s(h) + \rho\mathbb{E}_{q_{x_s}(x_s)}[{\eta_{\mathbf{q}}}^T(x_s)\ell(h(x_s))], \tag{5.3}$$

where $\eta_{\mathbf{q}}(x_s) = [q_{Y_s|X_s}(1|x_s), ..., q_{Y_s|X_s}(K|x_s)]^T$.

Remark 5.2. In Eq. (5.3), $\mathbb{E}_{q_{x_s}(x_s)}[{\eta_{\mathbf{q}}}^T(x_s)\ell(h(x_s))]$ represents the expected risk on the incorrect m.r.v.. To ensure to obtain useful knowledge from $\tilde{P}_s$, we need to avoid $\tilde{R}_s(h) \approx \mathbb{E}_{q_{x_s}(x_s)}[{\eta_{\mathbf{q}}}^T(x_s)\ell(h(x_s))]$. Specifically, we assume: there is a constant $0 < M_s < \infty$ such that $\mathbb{E}_{q_{x_s}(x_s)}[{\eta_{\mathbf{q}}}^T(x_s)\ell(h(x_s))] \leq R_s(h) + M_s$.

Theorem 5.1 shows that $\tilde{R}_s(h)$ only equals $R_s(h)$ when two cases happen: 1) $Q = I$ and $\rho = 0$ and 2) some special combinations (e.g., special p_{x_s}, q_{x_s}, Q, η and ℓ) to make the second term in Eq. (5.2) equal zero or to make the second term in Eq. (5.3) equal $\rho R_s(h)$. Case 1) means that source data are clean, which is not real in the wild. Case 2) almost never happens, since it is hard to find such special combinations when p_{x_s}, q_{x_s}, Q and η are unknown. Thus, $\tilde{R}_s(h)$ has an

essential difference with $R_s(h)$. Then, following proof skills in [Ben-David et al., 2010b], we derive the upper bound of $R_t(h)$ as follows.

Theorem 5.2. *For any $h \in \mathcal{H}$, we have*

$$\begin{aligned} R_t(h, f_t) \leq & \underbrace{\tilde{R}_s(h)}_{(i)\ \textbf{noisy-data risk}} + \underbrace{|R_t(h, \tilde{f}_t) - \tilde{R}_s(h, \tilde{f}_t)|}_{(ii)\ \textbf{discrepancy between distributions}} \\ & + \underbrace{|R_s(h, \tilde{f}_t) - R_s(h)|}_{(iii)\ \textbf{domain dissimilarity}} \\ & + \underbrace{|\tilde{R}_s(h) - R_s(h)| + |\tilde{R}_s(h, \tilde{f}_t) - R_s(h, \tilde{f}_t)|}_{(iv)\ \textbf{noise effects from source}\ \Delta_s} \\ & + \underbrace{|R_t(h, f_t) - R_t(h, \tilde{f}_t)|}_{(v)\ \textbf{noise effects from target}\ \Delta_t} . \end{aligned} \tag{5.4}$$

Remark 5.3. To ensure that we can gain useful knowledge from $\tilde{f}_t(x_t)$, we assume: there is a constant $0 < M_t < \infty$ such that $\mathbb{E}_{q_{x_s}(x)}[\ell(h(x), \tilde{f}_t(x))] \leq R_s(h, \tilde{f}_t) + M_t$ and $\mathbb{E}_{q_{x_t}(x)}[\ell(h(x), \tilde{f}_t(x))] \leq R_t(h, f_t) + M_t$.

It is clear that the upper bound of $R_t(h, f_t)$, shown in Eq. (5.4), has 5 terms. However, existing HoUDA methods only focus on minimizing (i) + (ii) [Ganin et al., 2016a; Ghifary et al., 2017; Long et al., 2015] or (i) + (ii) + (iii) [Saito et al., 2017], which ignores terms (iv) and (v) (i.e., $\Delta = \Delta_s + \Delta_t$). Thus, existing HoUDA methods cannot handle WUDA well.

5.5.2 Two-step Approach is a Compromise Solution

To reduce noise effects, a straightforward solution is two-step approach. For example, in the first step, we can train a classifier with noisy source data using Co-teaching [Han et al., 2018] and use this classifier to annotate pseudo labels for source data. In the second step, we use ATDA [Saito et al., 2017] to train

a target-domain classifier with pseudo-labeled-source data and pseudo-labeled target data.

Nonetheless, the pseudo-labeled source data are still noisy. Let labels of noisy source data $\tilde{y}_s$ be replaced with pseudo labels $\tilde{y}'_s$ after using Co-teaching. Noise effects Δ will become pseudo-label effects Δ_p as follows.

$$\Delta_p = \underbrace{|\tilde{R}'_s(h) - R_s(h)| + |\tilde{R}'_s(h, \tilde{f}_t) - R_s(h, \tilde{f}_t)|}_{\textbf{pseudo-labeled-source effects } \Delta'_s} + \Delta_t, \tag{5.5}$$

where $\tilde{R}'_s(h)$ and $\tilde{R}'_s(h, \tilde{f}_t)$ correspond to $\tilde{R}_s(h)$ and $\tilde{R}_s(h, \tilde{f}_t)$ in Δ_s. It is clear that the difference between Δ_p and Δ is $\Delta'_s - \Delta_s$. The first term in Δ'_s may be less than that in Δ_s due to a label-noise algorithm (e.g., Co-teaching [Han et al., 2018]), but the second term in Δ'_s may be higher than that in Δ_s since a label-noise algorithm (e.g., Co-teaching) does not consider to minimize it. Thus, it is hard to say whether $\Delta'_s < \Delta_s$ (i.e., $\Delta_p < \Delta$). This means that two-step approach may not really reduce noise effects.

5.5.3 Why does Butterfly Eliminate Noise Effect?

To eliminate noise effects Δ, we aim to select correct data simultaneously from noisy source data and pseudo-labeled target data. In theory, we prove that noise effects will be eliminated if we can select correct data with a high probability. Let ρ^s_{01} represent the probability that incorrect data is selected from noisy source data, and ρ^t_{01} represent the probability that incorrect data is selected from pseudo-labeled target data. Theorem 5.3 shows that $\Delta \to 0$ if $\rho^s_{01} \to 0$ and $\rho^t_{01} \to 0$ and presents a new upper bound of $R_t(h, f_t)$. Before stating Theorem 5.3, we first present two m.r.v.s below.

- (X_s, Y_s, V_s) defined on $\mathcal{X} \times \mathcal{Y} \times \mathcal{V}$ with a probability density $p_s^{\text{po}}(x_s, y_s, v_s)$, where $\mathcal{V} = \{0, 1\}$;
- (X_t, V_t) defined on $\mathcal{X} \times \mathcal{V}$ with a probability density $p_t^{\text{po}}(x_t, v_t)$, where $\mathcal{V} = \{0, 1\}$. $p_{V_t}^{\text{po}}(v_t)$ is the marginal density of $p_t^{\text{po}}(x_t, v_t)$.

V_s and V_t are *perfect-selection random variables*. Data drawn from the distribution of (X_s, Y_s, V_s) can be regarded as a pool that mixes the correct ($v_s = 1$) and incorrect ($v_s = 0$) source data. Data drawn from the distribution of (X_t, V_t) can be regarded as a pool that mixes the correct ($v_t = 1$) and incorrect ($v_t = 0$) pseudo-labeled target data. Namely, 1) $(X_s, Y_s | V_s = 1)$ is the correct m.r.v. defined at the beginning of Section 5.5, and 2) $V_t = 1$ means $f_t(x_t) = \tilde{f}_t(x_t)$ and $V_t = 0$ means $f_t(x_t) \neq \tilde{f}_t(x_t)$. It is clear that, higher value of $p_{V_t}^{\text{po}}(V_t = 1)$ means that $\tilde{f}_t$ is more like f_t. In following, we use ρ_{v_t} to represent $p_{V_t}^{\text{po}}(v_t = 0)$. Note that both perfect-selection random variables V_s and V_t cannot be observed and we can only observe following m.r.v.s.

- (X_s, Y_s, U_s) defined on $\mathcal{X} \times \mathcal{Y} \times \mathcal{V}$ with a probability density $\tilde{p}_s^{\text{po}}(x_s, y_s, u_s)$;
- (X_t, U_t) defined on $\mathcal{X} \times \mathcal{V}$ with a probability density $\tilde{p}_t^{\text{po}}(x_t, u_t)$. $\tilde{p}_{U_t}^{\text{po}}(u_t)$ is the marginal density of $\tilde{p}_t^{\text{po}}(x_t, u_t)$.

U_s and U_t are *algorithm-selection random variables*. Data drawn from the distribution of (X_s, Y_s, U_s) can be regarded as a pool that mixes the selected ($u_s = 1$) and unselected ($u_s = 0$) noisy source data. Data drawn from the distribution of (X_t, U_t) can be regarded as a pool that mixes the selected ($u_t = 1$) and unselected ($u_t = 0$) pseudo-labeled target data. We can obtain observations of (X_s, Y_s, U_s) and (X_t, U_t) using Algorithm 5.2. Namely, after executing Algorithm 5.2, we

can obtain observations $\{x_{si}, \tilde{y}_{si}, u_{si}\}_{i=1}^{n_s}$ and $\{x_{ti}, u_{ti}\}_{i=1}^{n_t}$. Based on (X_s, Y_s, U_s) and (X_t, U_t), we can define the following expected risks.

$$\tilde{R}_s^{\mathrm{po}}(h, u_s) = (1-\rho_{u_s})^{-1}\mathbb{E}_{\tilde{p}_s^{\mathrm{po}}(x_s,y_s,u_s)}[u_s \ell(h(x_s), y_s)],$$
$$\tilde{R}_t^{\mathrm{po}}(h, \tilde{f}_t, u_t) = (1-\rho_{u_t})^{-1}\mathbb{E}_{\tilde{p}_t^{\mathrm{po}}(x_t,u_t)}[u_t \ell(h(x_t), \tilde{f}_t(x_t))],$$
$$\tilde{R}_s^{\mathrm{po}}(h, \tilde{f}_t, u_s) = (1-\rho_{u_s})^{-1}\mathbb{E}_{\tilde{p}_s^{\mathrm{po}}(x_s,y_s,u_s)}[u_s \ell(h(x_s), \tilde{f}_t(x_s))].$$

where $\rho_{u_s} = \tilde{p}_{U_s}^{\mathrm{po}}(u_s = 0)$ and $\rho_{u_t} = \tilde{p}_{U_t}^{\mathrm{po}}(u_t = 0)$. Since we can observe (X_s, Y_s, U_s) and (X_t, U_t), the empirical estimators of these three risks can be easily computed using the loss function defined in Eq. (5.1). Then, we define following probabilities to describe the relation between perfect-selection random variables and algorithm-selection random variables, where $i, j = 0, 1$.

- $\rho_{ji}^s = \Pr(V_s = j | U_s = i)$ represents the probability of the event: $V_s = j$ given $U_s = i$,
- $\rho_{ji}^t = \Pr(V_t = j | U_t = i)$ represents the probability of the event: $V_t = j$ given $U_t = i$.

Remark 5.4. Based on above definitions, we know that 1) ρ_{01}^s is the probability that incorrect data is selected from noisy source data, and 2) ρ_{01}^t is the probability that incorrect data is selected from pseudo-labeled target data.

Using ρ_{ji}^s and ρ_{ji}^t, we can show 1) relation between probability densities of $(X_s, Y_s | V_s)$ and $(X_s, Y_s | U_s)$, 2) relation between probability densities of $(X_t | V_t)$ and $(X_t | U_t)$ using ρ_{ji}^s and ρ_{ji}^t as follows.

$$\tilde{p}_{X_s,Y_s|U_s}^{\mathrm{po}}(x_s, y_s | i) = \rho_{0i}^s p_{X_s,Y_s|V_s}^{\mathrm{po}}(x_s, y_s | 0) + \rho_{1i}^s p_{X_s,Y_s|V_s}^{\mathrm{po}}(x_s, y_s | 1),$$
$$\tilde{p}_{X_t|U_t}^{\mathrm{po}}(x_t | i) = \rho_{0i}^t p_{X_t|V_t}^{\mathrm{po}}(x_t | 0) + \rho_{1i}^t p_{X_t|V_t}^{\mathrm{po}}(x_t | 1).$$

Since

$$p^{\mathrm{po}}_{X_s,Y_s|V_s}(x_s,y_s|1) = p_s(x_s,y_s),$$
$$p^{\mathrm{po}}_{X_s,Y_s|V_s}(x_s,y_s|0) = q_s(x_s,y_s),$$
$$p^{\mathrm{po}}_{X_t|V_t}(x_t|0) = p_{x_t}(x_t)1_A(x_t)/P_{x_t}(A) = q_{x_t}(x_t),$$
$$p^{\mathrm{po}}_{X_t|V_t}(x_t|1) = p_{x_t}(x_t)1_B(x_t)/P_{x_t}(B) = p'_{x_t}(x_t),$$

we have

$$\tilde{p}^{\mathrm{po}}_{X_s,Y_s|U_s}(x_s,y_s|i) = \rho^s_{0i}q_s(x_s,y_s) + \rho^s_{1i}p_s(x_s,y_s), \tag{5.6}$$

$$\tilde{p}^{\mathrm{po}}_{X_t|U_t}(x_t|i) = \rho^t_{0i}q_{x_t}(x_t) + \rho^t_{1i}p'_{x_t}(x_t). \tag{5.7}$$

Remark 5.5. Eq. (5.6) and Eq. (5.7) show that, if $\rho^s_{01} \to 0$ and $\rho^t_{01} \to 0$, we have 1) $\tilde{p}^{\mathrm{po}}_{X_s,Y_s|U_s}(x_s,y_s|1) \to p_s(x_s,y_s)$ and 2) $\tilde{p}^{\mathrm{po}}_{X_t|U_t}(x_t|1) \to p'_{x_t}(x_t)$. Please note that we cannot prove Theorem 5.3 by directly using 1) and 2).

Next, we present a lemma to show relation between $\tilde{R}^{\mathrm{po}}_s(h,u_s)$ and $R_s(h)$.

Lemma 5.1. *Given the m.r.v.* (X_s,Y_s,U_s) *with the probability density* $\tilde{p}^{po}_s(x_s,y_s,u_s)$ *and Eq. (5.6), we have*

$$|\tilde{R}^{po}_s(h,u_s) - R_s(h)| \le \rho^s_{01}\max\{\mathbb{E}_{q_s(x_s,y_s)}[\ell(h(x_s),y_s)], R_s(h)\}. \tag{5.8}$$

Similar with Lemma 5.1, we can obtain

$$|\tilde{R}^{\mathrm{po}}_s(h,\tilde{f}_t,u_s) - R_s(h,\tilde{f}_t)| \le \rho^s_{01}\max\{\mathbb{E}_{q_{x_s}(x_s)}[\ell(h(x_s),\tilde{f}_t(x_s))], R_s(h,\tilde{f}_t)\}. \tag{5.9}$$

Then, we give another lemma to show relation between $\tilde{R}^{\mathrm{po}}_t(h,\tilde{f}_t,u_t)$ and $R_t(h,\tilde{f}_t)$.

Lemma 5.2. *Given the m.r.v.* (X_t, U_t) *with the probability density* $\tilde{p}_s^{po}(x_t, u_t)$ *and Eq. (5.7), if* $\mathbb{E}_{p'_{x_t}(x_t)}[\ell(h(x_t), f_t(x_t))] \leq R_t(h, f_t) + \rho_{01}^s M_t$*, then we have*

(5.10)

$$|\tilde{R}_t^{po}(h, \tilde{f}_t, u_t) - R_t(h, f_t)| \leq \rho_{01}^t \max\{\mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t), \tilde{f}_t(x_t))], R_t(h, f_t)\} + \rho_{11}^t \rho_{01}^s M_t.$$

Remark 5.6. In Lemma 5.2, $\mathbb{E}_{p'_{x_t}(x_t)}[\ell(h(x_t), f_t(x_t))] \leq R_t(h, f_t) + \rho_{01}^s M_t$ means that the expected risk restricted in B (i.e., $\mathbb{E}_{p'_{x_t}(x_t)}[\ell(h(x_t), f_t(x_t))]$) can represent the true risk $R_t(h, f_t)$ when ρ_{01}^s is small. In Butterfly, it is equivalent to that pseudo-labeled target data are more useful if we can select more correct data from noisy source data. If this assumption fails, we cannot gain useful knowledge from $\tilde{f}_t$ even when we can perfectly select correct data from pseudo-labeled target data ($\rho_{01}^t = 0$).

Inequalities (5.8), (5.9) and (5.10) show that if we can perfectly avoid annotating incorrect data as "correct" (i.e., $\rho_{01}^s = 0$ and $\rho_{01}^t = 0$), we have $\tilde{R}_s^{\text{po}}(h, u_s) = R_s(h)$, $\tilde{R}_s^{\text{po}}(h, \tilde{f}_t, u_t) = R_s(h, \tilde{f}_t)$ and $\tilde{R}_t^{\text{po}}(h, \tilde{f}_t, u_t) = R_t(h, f_t)$. Nonetheless, ρ_{01}^s and ρ_{01}^t never equal zero, and values of $\mathbb{E}_{q_s(x_s, y_s)}[\ell(h(x), y)]$, $\mathbb{E}_{q_{x_s}(x_s)}[\ell(h(x_s), \tilde{f}_t(x_s))]$ and $\mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t), \tilde{f}_t(x_t))]$ may equal $+\infty$ for some $h \in \mathcal{H}$. Namely, even when ρ_{01}^s and ρ_{01}^t are very small, $\tilde{R}_s^{\text{po}}(h, u_s)$ is probably far away from $R_s(h)$. Thus, without proper assumptions,it is useless to use (X_s, Y_s, U_s) to represent $(X_s, Y_s | V_s = 1)$.

In Theorem 5.3, we prove that, under assumptions in Remarks 5.2, 5.3 and Lemma 5.2, $\tilde{R}_s^{\text{po}}(h, u_s) \to R_s(h)$, $\tilde{R}_s^{\text{po}}(h, \tilde{f}_t, u_t) \to R_s(h, \tilde{f}_t)$ and $\tilde{R}_t^{\text{po}}(h, \tilde{f}_t, u_t) \to R_t(h, f_t)$ if $\rho_{01}^s \to 0$ and $\rho_{01}^t \to 0$. Moreover, we give a new upper bound of $R_t(h, f_t)$. In the new upper bound, we show that: $\Delta \to 0$ if $\rho_{01}^s \to 0$ and $\rho_{01}^t \to 0$.

Theorem 5.3. *Given two m.r.v.s* (X_s, Y_s, U_s) *defined on* $\mathcal{X} \times \mathcal{Y} \times \mathcal{V}$ *and* (X_t, U_t) *defined on* $\mathcal{X} \times \mathcal{V}$*, under the assumptions in Remark 5.2, Remark 5.3 and Lemma 5.2,*

$\forall \epsilon \in (0,1)$, there are δ_s and δ_t, if $\rho^s_{01} < \delta_s$ and $\rho^t_{01} < \delta_t$, for any $h \in \mathcal{H}$, we will have

$$|\tilde{R}^{po}_s(h, \tilde{f}_t, u_s) - R_s(h, \tilde{f}_t)| + |\tilde{R}^{po}_s(h, u_s) - R_s(h)| < 2\epsilon. \tag{5.11}$$

Moreover, we will have

$$\begin{aligned} R_t(h, f_t) \leq & \underbrace{\tilde{R}^{po}_s(h, u_s)}_{(i)\ \textbf{noisy-data risk}} + \underbrace{|\tilde{R}^{po}_t(h, \tilde{f}_t, u_t) - \tilde{R}^{po}_s(h, \tilde{f}_t, u_s)|}_{(ii)\ \textbf{discrepancy between distributions}} \\ & + \underbrace{|R_s(h, \tilde{f}_t) - R_s(h)|}_{(iii)\ \textbf{domain dissimilarity}} + \underbrace{2\epsilon}_{(iv)\ \textbf{noise effects}\ \Delta_s} + \underbrace{2\epsilon}_{(iv)\ \textbf{noise effects}\ \Delta_t}. \end{aligned} \tag{5.12}$$

Theorem 5.3 shows that if selected data have a high probability to be correct ones ($\rho^s_{01} \to 0$ and $\rho^t_{01} \to 0$), then Δ_s and Δ_t approach zero, meaning that noise effects are eliminated. This motivates us to find a reliable way to select correct data from noisy source data and pseudo-labeled target data and propose the butterfly to WUDA problem.

5.5.4 Principle-guided Butterfly

Guided by Theorem 5.3, a robust approach should check high-correctness data out (meaning $\rho^s_{01} \to 0$ and $\rho^t_{01} \to 0$). This checking process will make (iv) and (v), $2\epsilon + 2\epsilon$, become 0. Then, we can obtain gradients of $\tilde{R}^{\text{po}}_s(h, u_s)$, $\tilde{R}_s(h, \tilde{f}_t, u_s)$ and $\tilde{R}^{\text{po}}_t(h, \tilde{f}_t, u_t)$ w.r.t. parameters of h and use these gradients to minimize them, which minimizes (i) and (ii) as $(i)+(ii) \leq \tilde{R}^{\text{po}}_s(h, u_s) + \tilde{R}_s(h, \tilde{f}_t, u_s) + \tilde{R}^{\text{po}}_t(h, \tilde{f}_t, u_t)$. Note that (iii) cannot be directly minimized since we cannot pinpoint clean source data. However, following [Saito et al., 2017], we can indirectly minimize (iii) via minimizing $\tilde{R}^{\text{po}}_s(h, u_s) + \tilde{R}^{\text{po}}_s(h, \tilde{f}_t, u_s)$, as $(iii) \leq R_s(h, \tilde{f}_t) + R_s(h) \leq \tilde{R}^{\text{po}}_s(h, u_s) + \tilde{R}^{\text{po}}_s(h, \tilde{f}_t, u_s) + 2\epsilon$, where the last inequality follows Eq. (5.11). This means that a robust approach guided by Theorem 5.3 can minimize all terms in the right side of inequality in Eq. (5.12).

To realize this robust approach, we propose a Butterfly framework (Algorithm 5.1), which trains four networks dividing into two branches (Figure 5.3). By using dual-checking principle, Branch-I checks which data is correct in the mixture domain; while Branch-II checks which pseudo-labeled target data is correct. To ensure these checked data highly-correct, we apply the small-loss trick based on memorization effects of deep learning [Arpit et al., 2017]. After cross-propagating these checked data [Bengio, 2014], Butterfly can obtain high-quality DIR and TSR simultaneously in an iterative manner. Theoretically, Branch-I minimizes (i)+(ii)+(iii)+(iv); while Branch-II minimizes (ii)+(v). This means that Butterfly can minimize all terms in the right side of inequality in Eq. (5.12).

5.5.5 A Generalization Bound for WUDA

In this subsection, we prove a generalization bound for WUDA problem using the loss function Eq. (5.1) and Theorem 5.3[3]. First, we introduce the Rademacher complexity of a class of vector-valued functions [Bartlett and Mendelson, 2002; Li et al., 2019a, 2018b; Mansour et al., 2009; Maurer, 2016; Zhang et al., 2019], which measures the degree to which a class can fit random noise. Rademacher Complexity of $\mathcal{H}$ is defined as follows.

Definition 5.2 (Rademacher Complexity of $\mathcal{H}$)**.** *Given a sample $S = \{(x_i)\}_{i=1}^n$, the empirical Rademacher complexity of the set $\mathcal{H}$ is defined as follows.*

$$\hat{\Re}_S(\mathcal{H}) = \frac{2}{m}\mathbb{E}_{\sigma}\Big(\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sum_{k=1}^{K}\sigma_{ik}h_k(x_i)\Big),$$

[3]Please note that this is a generalization bound for WUDA problem rather than Butterfly. In Butterfly, we essentially have four classifiers (F_1, F_2, F_{t1}, F_{t2}), which is very difficult to analyze it. We will develop a generalization and estimation error bound for Butterfly in the future.

where $h_k(\cdot)$ is the k^{th} component of function $h \in \mathcal{H}$ and the σ_{ik} are $n \times K$ matrix of independent Rademacher variables [Maurer, 2016]. The Rademacher complexity of the set $\mathcal{H}$ is defined as the expectation of $\hat{\mathfrak{R}}_H(\mathcal{H})$ over all samples of size n:

$$\mathfrak{R}_n(\mathcal{H}) = \mathop{\mathbb{E}}_{S}\Big(\hat{\mathfrak{R}}_S(\mathcal{H})\Big||S| = n\Big).$$

Then, using the Rademacher complexity, we can prove an upper bound of $\tilde{R}_s^{\text{po}}(h,u_s)$ to show the relation between $\tilde{R}_s^{\text{po}}(h,u_s)$ and the loss function Eq. (5.1). As a common practice [Kiryo et al., 2017; Mohri et al., 2018], we assume that, 1) there are $C_h > 0$ and $C_L > 0$ such that $\sup_{h\in\mathcal{H}} \|h\|_\infty \le C_h$ and $\sup_{\|t\|_\infty \le C_h} \max_y \ell(t,y) \le C_L$, and 2) $\ell(t,y)$ is Lipschitz continuous in $\|t\|_\infty \le C_h$ with a Lipschitz constant L_ℓ.

Lemma 5.3. *Given a sample $S_s = \{(x_{si}, y_{si}, u_{si})\}_{i=1}^n$ drawn from the probability density $\tilde{p}_s^{po}(x_s, y_s, u_s)$, with the probability of at least $1-\delta$ over samples S_s of size n drawn from $\tilde{p}_s^{po}(x_s, y_s, u_s)$, the following inequality holds.*

$$\tilde{R}_s^{po}(h,\mathbf{u}_s) \le \mathcal{L}(\theta, h; \mathbf{u}_s, D_s^{xy}) + \frac{\sqrt{2}L_\ell \hat{\mathfrak{R}}_{D_s^x}(\mathcal{H})}{1-\tau_s} + \frac{3C_L}{1-\tau_s}\sqrt{\frac{\ln\frac{\delta}{2}}{2n}}, \tag{5.13}$$

where $\mathcal{L}$ is defined in Eq. (5.1), $D_s^{xy} = \{x_{si}, y_{si}\}_{i=1}^n$, $D_s^x = \{x_{si}\}_{i=1}^n$, $\mathbf{u}_s = [u_{s1}, \ldots, u_{sn}]^T$ and $\tau_s = \rho_{u_s} = 1 - \sum_{i=1}^n u_{si}/n$.

Finally, we prove the generalization bound for WUDA problem as follows.

Theorem 5.4 (Generalization Bound for WUDA)**.** *Given $S_s = \{(x_{si}, y_{si}, u_{si})\}_{i=1}^{n_s}$ drawn from the probability density $\tilde{p}_s^{po}(x_s, y_s, u_s)$ and $S_t = \{(x_{ti}, u_{ti})\}_{i=1}^{n_t}$ drawn from the probability density $\tilde{p}_t^{po}(x_t, u_t)$, under the assumptions in Remark 5.2, Remark 5.3 and Lemma 5.2, if $\rho_{01}^s < 1/\sqrt{n_s T}$ and $\rho_{01}^t < 1/\sqrt{n_t T}$, then, with the*

probability of at least $1-3\delta$*, for any* $h \in \mathcal{H}$*, the following inequality holds.*

$$R_t(h,f_t) \le 2\Big(\mathcal{L}(\theta,h;\mathbf{u}_s,D_s^{xy}) + \mathcal{L}(\theta,h;\mathbf{u}_s,D_{\tilde{s}}^{xy})\Big) + \mathcal{L}(\theta,h;\mathbf{u}_t,D_{\tilde{t}}^{xy}) + \frac{4\sqrt{2}L_\ell \hat{\Re}_{D_s^x}(\mathcal{H})}{1-\tau_s}$$
$$+ \frac{\sqrt{2}L_\ell \hat{\Re}_{D_t^x}(\mathcal{H})}{1-\tau_t} + \frac{12C_L}{1-\tau_s}\sqrt{\frac{\ln\frac{\delta}{2}}{2n_s}} + \frac{3C_L}{1-\tau_t}\sqrt{\frac{\ln\frac{\delta}{2}}{2n_t}} + \frac{2M_s+M_t}{\sqrt{n_sT}} + \frac{3M_t}{\sqrt{n_tT}}, \tag{5.14}$$

where $\mathcal{L}$ *is defined in Eq.* (5.1)*,* T *is the number of training epochs,* $D_s^{xy} = \{x_{si}, y_{si}\}_{i=1}^{n_s}$*,* $D_{\tilde{s}}^{xy} = \{x_{si}, \tilde{f}_t(x_{si})\}_{i=1}^{n_s}$*,* $D_{\tilde{t}}^{xy} = \{x_{ti}, \tilde{f}_t(x_{ti})\}_{i=1}^{n_t}$ $D_s^x = \{x_{si}\}_{i=1}^{n_s}$*,* $D_t^x = \{x_{ti}\}_{i=1}^{n_t}$*,* $\mathbf{u}_s = [u_{s1},\ldots,u_{sn_s}]^T$*,* $\tau_s = \rho_{u_s} = 1 - \sum_{i=1}^{n_s} u_{si}/n_s$*,* $\mathbf{u}_t = [u_{t1},\ldots,u_{tn_t}]^T$ *and* $\tau_t = \rho_{u_t} = 1 - \sum_{i=1}^{n_t} u_{ti}/n_t$*.*

Theorem 5.4 shows the empirical upper bound of the target risk (i.e., $R_t(h,f_t)$). Based on this bound, we can obtain the estimation error bound of $R_t(h,f_t)$ as follows. First, let

$$\hat{R}_t^{\mathcal{L}}(h,S_s,S_t) = 2\Big(\mathcal{L}(\theta,h;\mathbf{u}_s,D_s^{xy}) + \mathcal{L}(\theta,h;\mathbf{u}_s,D_{\tilde{s}}^{xy})\Big) + \mathcal{L}(\theta,h;\mathbf{u}_t,D_{\tilde{t}}^{xy}), \tag{5.15}$$

where D_s^{xy}, $D_{\tilde{s}}^{xy}$ and $D_{\tilde{t}}^{xy}$ are defined in Theorem 5.4, and $\widetilde{h} = \arg\min_{h\in\mathcal{H}} \hat{R}_t^{\mathcal{L}}(h,S_s,S_t)$ means the empirical minimizer of $\hat{R}_t^{\mathcal{L}}(h,S_s,S_t)$, and $h^* = \arg\min_{h\in\mathcal{H}} R_t(h,f_t)$ means the true risk minimizer of $R_t(h,f_t)$, and $\mathcal{H}' = \{h | \hat{R}_t^{\mathcal{L}}(h,S_s,S_t) \le \epsilon\}$. Then, we have

$$\begin{aligned}
R_t(\widetilde{h},f_t) - R_t(h^*,f_t) = {} & R_t(\widetilde{h},f_t) - \hat{R}_t^{\mathcal{L}}(\widetilde{h},S_s,S_t) + \hat{R}_t^{\mathcal{L}}(\widetilde{h},S_s,S_t) - R_t(h^*,f_t) \\
& + \hat{R}_t^{\mathcal{L}}(h^*,S_s,S_t) - \hat{R}_t^{\mathcal{L}}(h^*,S_s,S_t) \\
= {} & R_t(\widetilde{h},f_t) - \hat{R}_t^{\mathcal{L}}(\widetilde{h},S_s,S_t) + \hat{R}_t^{\mathcal{L}}(h^*,S_s,S_t) - R_t(h^*,f_t) \\
& + \hat{R}_t^{\mathcal{L}}(\widetilde{h},S_s,S_t) - \hat{R}_t^{\mathcal{L}}(h^*,S_s,S_t) \\
\le {} & \sup_{h\in\mathcal{H}'}(R_t(h,f_t) - \hat{R}_t^{\mathcal{L}}(h,S_s,S_t)) + \epsilon + 0,
\end{aligned} \tag{5.16}$$

where $\hat{R}_t^{\mathcal{L}}(\widetilde{h}, S_s, S_t) \leq \hat{R}_t^{\mathcal{L}}(h^*, S_s, S_t)$ due to the definition of $\widetilde{h}$. If all conditions in Theorem 5.4 are satisfied, with the probability of at least $1-3\delta$, for any $h \in \mathcal{H}$, we have

$$R_t(\widetilde{h}, f_t) - R_t(h^*, f_t) \leq \frac{4\sqrt{2}L_\ell \hat{\mathfrak{R}}_{D_s^x}(\mathcal{H}')}{1-\tau_s} + \frac{\sqrt{2}L_\ell \hat{\mathfrak{R}}_{D_t^x}(\mathcal{H}')}{1-\tau_t} + \frac{12C_L}{1-\tau_s}\sqrt{\frac{\ln\frac{\delta}{2}}{2n_s}} + \frac{3C_L}{1-\tau_t}\sqrt{\frac{\ln\frac{\delta}{2}}{2n_t}} + \frac{2M_s + 1M_t}{\sqrt{n_s T}} + \frac{3M_t}{\sqrt{n_t T}} + \epsilon. \tag{5.17}$$

Eq. (5.17) ensures that learning with $\hat{R}_t^{\mathcal{L}}(\widetilde{h}, S_s, S_t)$ is consistent: as $n_s, n_t \to \infty$ and $\epsilon \to 0$, $R_t(\widetilde{h}, f_t) \to R_t(h^*, f_t)$. For linear-in-parameter model with a bounded norm, $\hat{\mathfrak{R}}_{D_s^x}(\mathcal{H}) = \mathcal{O}(1/\sqrt{n_s})$ and $\hat{\mathfrak{R}}_{D_t^x}(\mathcal{H}) = \mathcal{O}(1/\sqrt{n_t})$ and thus $R_t(\widetilde{h}, f_t) \to R_t(h^*, f_t)$ in $\mathcal{O}(1/\sqrt{n_s} + 1/\sqrt{n_t})$.

5.6 Comparison to Related Works

In this section, we compare Butterfly with related works and show why related works cannot handle WUDA problem.

Relations to Co-teaching. As Butterfly is related to Co-teaching, we discuss their major differences here. Although Co-teaching [Han et al., 2018] applies the small-loss trick and the cross-update technique to train deep networks against noisy data, it can only deal with one-domain problem instead cross-domain problem. Besides, we argue that Butterfly is not a simple mixtrue of Co-teaching and ATDA for two reasons.

First, network structure of Butterfly is different with that of ATDA and Co-teaching: Butterfly maintains four networks; while ATDA maintains three and Co-teaching maintains two. We cannot simply combine ADTA and Co-teaching to derive Butterfly. Second, we have justified that the sequential mixture of

Co-teaching and ATDA (i.e., two-step method) cannot eliminate noise effects caused by noisy source data (see Section 5.4). Specifically, two-step methods only take care of part of noise effects but Butterfly takes care of the whole noise effects. Thus, Butterfly is the first method to eliminate noise effects rather than alleviate it.

Relations to TCL. Recently, *transferable curriculum learning* (TCL) is a robust HoUDA method to handle noise [Shu et al., 2019]. TCL uses small-loss trick to train the *domain-adversarial neural network* (DANN) [Ganin et al., 2016a]. However, TCL can only minimize $(i)+(ii)+(iv)$, while Butterfly can minimize all terms in the right side of Eq. (5.12).

5.7 Experiments

We conduct experiments on 32 simulated WUDA tasks and 3 real-world WUDA tasks to verify the efficacy of Butterfly in this section.

5.7.1 Simulated WUDA Tasks

We verify the effectiveness of our approach on three benchmark datasets (vision and text), including *MNIST*, *SYN-DIGITS (SYND)*[4] and *human-sentiment* analysis (i.e., *Amazon products reviews* on *book*, *dvd*, *electronics* and *kitchen*) [5]. They are used to construct 14 basic tasks: *MNIST*→*SYND* (*M*→*S*), *SYND*→*MNIST* (*S*→*M*), *book*→*dvd* (*B*→*D*), *book*→*electronics* (*B*→*E*), ... , and *kitchen* → *electron-*

[4] *Digit* datasets (*MNIST* and *SYN Digit*) can be downloaded from official code of ATDA. The link is https://github.com/ksaito-ut/atda.

[5] *Sentiment* datasets (*Amazon products reviews*) can be downloaded from the official code of marginalized Stacked Denoising Autoencoder. The link is https://www.cse.wustl.edu/~mchen/code/mSDA.tar.

ics ($K \rightarrow E$). These tasks are often used for evaluation of HoUDA methods [Ganin et al., 2016a; Saito et al., 2017, 2018]. Figure 5.4 shows datasets *MNIST* and *SYND*.

Since all source datasets are clean, we corrupt source data using symmetry flipping [Patrini et al., 2017] and pair flipping [Han et al., 2018] with noise rate ρ chosen from $\{0.2, 0.45\}$. Precise definitions of Symmetry flipping (Q_S) and Pair flipping (Q_P) are presented below, where ρ is the noise rate and K is the number of labels.

$$Q_S = \begin{bmatrix} 1-\rho & \frac{\rho}{K-1} & \cdots & \frac{\rho}{K-1} & \frac{\rho}{K-1} \\ \frac{\rho}{K-1} & 1-\rho & \frac{\rho}{K-1} & \cdots & \frac{\rho}{K-1} \\ \vdots & & \ddots & & \vdots \\ \frac{\rho}{K-1} & \cdots & \frac{\rho}{K-1} & 1-\rho & \frac{\rho}{K-1} \\ \frac{\rho}{K-1} & \frac{\rho}{K-1} & \cdots & \frac{\rho}{K-1} & 1-\rho \end{bmatrix}, \tag{5.18}$$

$$Q_P = \begin{bmatrix} 1-\rho & \rho & 0 & \dots & 0 \\ 0 & 1-\rho & \rho & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & & & 1-\rho & \rho \\ \rho & 0 & \dots & 0 & 1-\rho \end{bmatrix}. \tag{5.19}$$

So, for each basic task, we have four kinds of noisy source data: *Pair*-45% (P45), *Pair*-20% (P20), *Symmetry*-45% (S45), *Symmetry*-20% (S20). Following [Han et al., 2018; Jiang et al., 2018], we can corrupt clean-label datasets manually using the noise transition matrix Q_S and Q_P. Namely, we evaluate the performance of each method using 32 simulated WUDA tasks: 8 digit tasks and 24 human-sentiment tasks. Note that the human-sentiment task is a binary classification

(a) *MNIST* (b) *SYND*

Figure 5.4: Visualization of *MNIST* and *SYND*.

problem, so pair flipping is equal to symmetry flipping. Thus, we only have 24 human-sentiment tasks.

5.7.2 Real-world WUDA Tasks

We also verify the efficacy of our approach on "cross-dataset benchmark" including *Bing*, *Caltech256*, *Imagenet* and *SUN* [Tommasi and Tuytelaars, 2014] [6]. In this benchmark, *Bing*, *Caltech256*, *Imagenet* and *SUN* contain common 40 classes. Since *Bing* dataset was formed by collecting images retrieved by Bing image search, it contains rich noisy data, with presence of multiple objects in the same image and caricaturization [Tommasi and Tuytelaars, 2014]. We use *Bing* as noisy source data, and *Caltech256*, *Imagenet* and *SUN* as unlabeled target data, which can form three real-world WUDA tasks. Figure 5.5 shows datasets *Bing*, *Caltech256*, *Imagenet* and *SUN* (taking "horse" as the common class).

[6] *Real-world* datasets (*BCIS*) can be downloaded from the website of the project "A Testbed for Cross-Dataset Analysis": https://sites.google.com/site/crossdataset/home/files ("setup DENSE decaf7", 1.3GB, decaf7 features).

(a) *Bing* provided by [Bergamo and Torresani, 2010]

(b) *Caltech256* provided by [Griffin et al., 2007]

(c) *ImageNet* provided by [Deng et al., 2009]

(d) *SUN* provided by [Xiao et al., 2010]

Figure 5.5: Visualization of *Bing*, *Caltech256*, *ImageNet* and *SUN* (taking "horse" as the common class).

5.7.3 Baselines

We realize Butterfly using four networks (abbreviated as B-Net) and compare B-Net with following baselines: 1) ATDA: representative pseudo-labeling-based HoUDA method [Saito et al., 2017]; 2) *deep adaptation networks* (DAN): representative IPM-based HoUDA method [Long et al., 2015]; 3) DANN: representative adversarial-training-based HoUDA method [Ganin et al., 2016a]; 4) TCL: an existing robust HoUDA method; 5) Co teaching+ATDA (Co+ATDA): a two-step method (see Section 5.4); 6) Co teaching+TCL (Co+TCL): a two-step method.

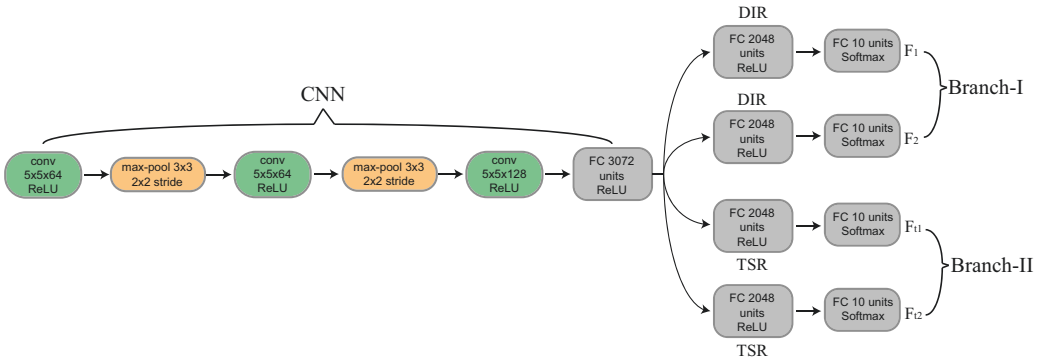
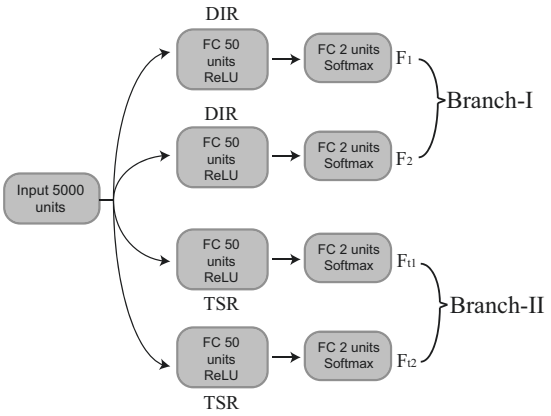


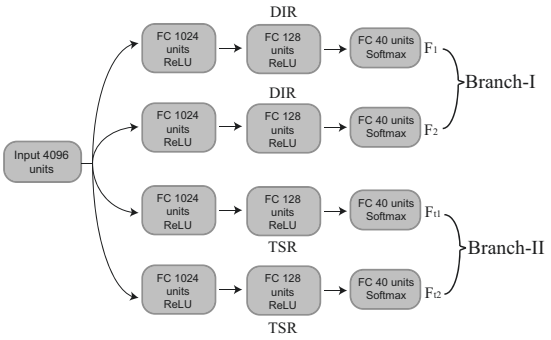
Figure 5.6: The architecture of B-Net for digit WUDA tasks *SYND* ↔ *MNIST*. We added BN layer in the last convolution layer in CNN and FC layers in F_1 and F_2. We also used dropout in the last convolution layer in CNN and FC layers in F_1, F_2, F_{t1} and F_{t2} (dropout probability is set to 0.5).

5.7.4 Network Structure and Optimizer

We implement all methods on Python 3.6 with a NIVIDIA P100 GPU. We use MomentumSGD for optimization in digit and real-world tasks, and set the momentum as 0.9. We use Adagrad for optimization in human-sentiment tasks because of sparsity of review data [Saito et al., 2017]. F_1, F_2, F_{t1} and F_{t2} are 6-layer CNN (3 convolutional and 3 fully-connected layers) for digit tasks; and are 3-layer neural networks (3 fully-connected layers) for human-sentiment tasks; and are 4-layer neural networks (4 fully-connected layers) for real-world tasks. The ReLU active function is used as avtivation function of these networks. Besides, dropout and batch normalization are also used. The network topology is shown in Figures 5.6 and 5.7. As deep networks are highly nonconvex, even with the same network and optimization method, different initializations can lead to different local optimal. Thus, following [Malach and Shalev-Shwartz, 2017], we take four networks with the same architecture but different initialization as four classifiers.



(a) Human-sentiment



(b) Real-world

Figure 5.7: The architecture of B-Net for (a) human-sentiment WUDA tasks and (b) real-world WUDA tasks. We added BN layer in the first FC layers in F_1 and F_2. We also used dropout in the first FC layers in F_1, F_2, F_{t1} and F_{t2} (dropout probability is set to 0.5).

5.7.5 Experimental Setup

For all 35 WUDA tasks, T_k is set to 5, and T_{max} is set to 30, and $\ell(\cdot,\cdot)$ is the cross-entropy loss function. Learning rate is set to 0.01 for simulated tasks and 0.05 for real-world WUDA tasks, τ_t is set to 0.05 for simulated tasks and 0.02 for real-world WUDA tasks. Confidence level of labeling function in line 8 of Algorithm 5.1 is set to 0.95 for 8 digit tasks, and 0.9 for 24 human-sentiment tasks and 0.8 for real-world WUDA tasks. τ is set to 0.4 for digit tasks, 0.1 for

Table 5.1: Target-domain accuracy on 8 digit WUDA tasks (*SYND↔MNIST*). Bold value represents the highest accuracy in each row.

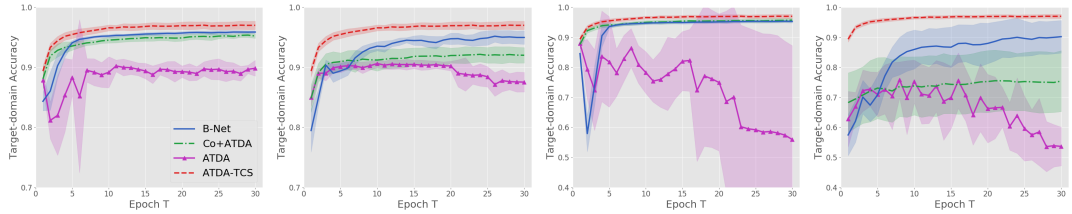
Tasks	Type	DAN	DANN	ATDA	TCL	Co+TCL	Co+ATDA	B-Net
$S \rightarrow M$	P20	90.17%	79.06%	55.95%	80.81%	88.56%	**95.37%**	95.29%
	P45	67.00%	55.34%	53.66%	55.97%	73.27%	75.43%	**90.21%**
	S20	90.74%	75.19%	89.87%	80.23%	85.88%	95.22%	**95.88%**
	S45	89.31%	65.87%	87.53%	68.54%	75.69%	92.03%	**94.97%**
$M \rightarrow S$	P20	40.82%	58.78%	33.74%	58.88%	59.08%	58.02%	**60.36%**
	P45	28.41%	43.70%	19.50%	45.31%	47.15%	46.80%	**56.62%**
	S20	30.62%	53.52%	49.80%	56.74%	56.91%	56.64%	**57.05%**
	S45	28.21%	43.76%	17.20%	49.91%	51.22%	54.29%	**56.18%**
Average		58.16%	58.01%	50.91%	62.05%	67.22%	71.73%	**75.82%**

human-sentiment tasks, 0.2 for real-world WUDA tasks. $n^l_{t,max}$ is set to $15,000$ for digit tasks, 500 for human-sentiment tasks and 4000 for real-world WUDA tasks. N_{max} is set to 1000 for digit tasks, and 200 for human-sentiment and real-world tasks. Batch size is set to 128 for digit, real-world WUDA tasks, and 24 for human-sentiment tasks. Penalty parameter is set to 0.01 for digit, real-world WUDA tasks, and 0.001 for human-sentiment tasks.

To fairly compare all methods, they have the same network structure. Namely, ATDA, DAN, DANN, TCL and B-Net adopt the same network structure for each dataset. Note that DANN and TCL use the same structure for their discriminate networks. All experiments are repeated 10 times and we report the average accuracy values and *standard deviation* (STD) of accuracy values of 10 experiments.

5.7.6 Results on Simulated WUDA Tasks

This subsection presents accuracy on unlabled target data (i.e., target-domain accuracy) in 32 simulated WUDA tasks.



(a) S20 (b) S45 (c) P20 (d) P45

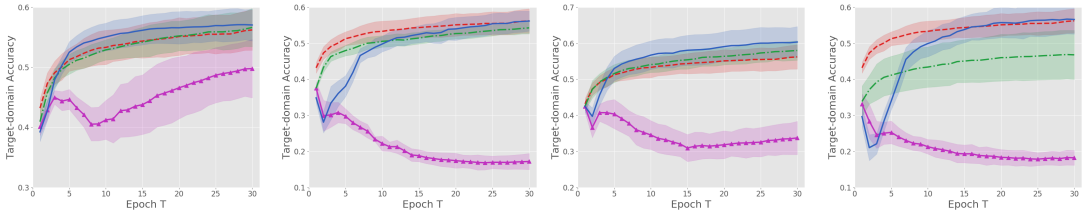
Figure 5.8: Target-domain accuracy vs. number of epochs on four *SYND→MNIST* WUDA tasks.

5.7.6.1 Results on digits WUDA tasks

Table 5.1 reports the target-domain accuracy in 8 digit tasks. As can be seen, average target-domain accuracy of B-Net is higher than those of all baselines. On S20 case (the easiest case), most methods work well. ATDA has a satisfactory performance although it does not consider the noise effects explicitly. Then, when facing harder cases (i.e., P20 and P45), ATDA fails to transfer useful knowledge from noisy source data to unlabeled target data. When facing the hardest cases (i.e., $M \rightarrow S$ with P45 and S45), DANN has higher accuracy than DAN and ATDA have. However, when facing the easiest cases (i.e., $S \rightarrow M$ with P20 and S20), the performance of DANN is worse than that of DAN and ATDA.

Although two-step method Co+ATDA (or Co+TCL) outperforms ATDA (or TCL) in all 8 tasks, it cannot beat one-step method: B-Net in terms of average target-domain accuracy. This result is an evidence for the claim in Section 5.4. In the task $S \rightarrow M$ with P20, Co+ATDA outperforms all methods (slightly higher than B-Net), since pseudo-labeled source data are almost correct.

Figures 5.8 and 5.9 show the target-domain accuracy vs. number of epochs among ATDA, Co+ATDA and B-Net. Besides, we show the accuracy of ATDA trained with clean source data (ATDA-TCS) as a reference point. When accuracy



(a) S20 (b) S45 (c) P20 (d) P45

Figure 5.9: Target-domain accuracy vs. number of epochs on four *MNIST→SYND* WUDA tasks.

of one method is close to that of ATDA-TCS (red dash line), this method successfully eliminates noise effects. From our observations, it is clear that B-Net is very close to ATDA-TCS in 7 out of 8 tasks (except for $S \rightarrow M$ task with P45, Figure 5.8-(d)), which is an evidence that Butterfly can eliminate noise effects. Since P45 case is the hardest one, it is reasonable that B-Net cannot perfectly eliminate noise effects. An interesting phenomenon is that, B-Net outperforms ATDA-TCS in 2 $M \rightarrow S$ tasks (Figure 5.9-(a), (c)). This means that B-Net transfers more useful knowledge (from noisy source data to unlabeled target data) even than ATDA-TCS (from clean source data to unlabeled target data).

5.7.6.2 Results on human sentiment WUDA tasks

Tables 5.2 and 5.3 report the target-domain accuracy of each method in 24 human-sentiment WUDA tasks. For these tasks, B-Net has the highest average target-domain accuracy. It should be noted that two-step method does not always perform better than existing HoUDA methods, such as for 20%-noise situation. The reason is that Co-teaching performs poorly when pinpointing clean source data from noisy source data. Another observation is that noise effects is not eliminated like target-domain accuracy in 8 digit WUDA tasks. The reason is

Table 5.2: Target-domain accuracy on 12 human-sentiment WUDA tasks with the 20% noise rate. Bold values mean the highest values in each row.

Tasks	DAN	DANN	ATDA	TCL	Co+TCL	Co+ATDA	B-Net
B→*D*	68.28%	68.08%	70.31%	71.40%	67.81%	66.70%	**71.84%**
B→*E*	63.78%	63.53%	72.79%	65.08%	60.54%	68.89%	**75.92%**
B→*K*	65.48%	64.63%	71.79%	66.80%	61.23%	66.51%	**76.32%**
D→*B*	64.63%	64.52%	70.25%	67.33%	65.22%	68.04%	**70.56%**
D→*E*	65.33%	65.16%	69.99%	66.74%	64.55%	67.32%	**73.73%**
D→*K*	65.68%	66.28%	74.53%	68.82%	67.98%	72.20%	**77.97%**
E→*B*	60.41%	60.15%	**63.89%**	63.13%	61.18%	61.08%	62.22%
E→*D*	62.35%	61.67%	62.30%	62.93%	60.81%	59.77%	**63.53%**
E→*K*	72.05%	71.51%	74.00%	75.36%	72.65%	70.85%	**78.96%**
K→*B*	59.94%	59.40%	**63.53%**	62.77%	60.71%	61.22%	63.36%
K→*D*	61.46%	61.51%	64.66%	64.16%	64.15%	64.94%	**66.98%**
K→*E*	70.60%	72.23%	74.75%	74.14%	68.95%	69.69%	**76.96%**
Average	65.00%	64.89%	69.40%	67.39%	64.65%	66.43%	**71.53%**

Table 5.3: Target-domain accuracy on 12 human-sentiment WUDA tasks with the 45% noise rate. Bold values mean the highest values in each row.

Tasks	DAN	DANN	ATDA	TCL	Co+TCL	Co+ATDA	B-Net
B→*D*	52.43%	52.98%	53.56%	54.44%	53.21%	54.32%	**56.59%**
B→*E*	52.17%	53.50%	55.14%	54.14%	53.98%	**57.34%**	55.74%
B→*K*	52.89%	51.84%	51.14%	53.32%	51.77%	53.28%	**57.00%**
D→*B*	53.11%	53.04%	54.48%	53.27%	54.85%	**55.95%**	55.15%
D→*E*	51.30%	53.04%	54.21%	53.77%	55.63%	56.08%	**58.91%**
D→*K*	52.15%	53.17%	57.99%	52.45%	58.10%	59.94%	**66.20%**
E→*B*	51.38%	51.08%	52.54%	52.14%	54.88%	53.30%	**54.93%**
E→*D*	52.83%	51.24%	49.02%	52.57%	50.03%	49.62%	**52.88%**
E→*K*	54.21%	53.58%	51.66%	55.04%	56.15%	52.10%	**56.12%**
K→*B*	50.44%	51.77%	**51.96%**	51.50%	53.81%	52.59%	51.39%
K→*D*	52.20%	51.45%	52.86%	53.19%	55.69%	54.52%	**53.53%**
K→*E*	**54.72%**	53.33%	52.11%	53.46%	51.26%	52.62%	53.71%
Average	52.49%	52.50%	53.65%	53.27%	54.11%	54.31%	**56.01%**

that these datasets provide fixed features and we cannot extract better features in the training process. However, in digit WUDA tasks, we gradually obtains better features of each domain and finally eliminate noise effects.

Table 5.4: Target-domain accuracy on 3 real-world WUDA tasks. The source domain is the *Bing* dataset that contains noisy information from the Internet. Bold value represents the highest accuracy in each row.

Target	DAN	DANN	ATDA	TCL	Co+TCL	Co+ATDA	B-Net
Caltech256	77.83%	78.00%	80.84%	79.35%	79.27%	79.89%	**81.71%**
Imagenet	70.29%	72.16%	74.89%	72.53%	72.33%	74.73%	**75.00%**
SUN	24.56%	26.80%	26.26%	28.80%	29.15%	26.31%	**30.54%**
Average	57.56%	58.99%	60.66%	60.23%	60.25%	60.31%	**62.42%**

5.7.7 Results on Real-world WUDA Tasks

Table 5.4 reports the target-domain accuracy in 3 tasks. B-Net enjoys the best performance on all tasks. It should be noted that, in *Bing→Caltech256* and *Bing→ImageNet* tasks, ATDA is slightly worse than B-Net. However, in *Bing→SUN* task, ATDA is much worse than B-Net. The reason is that the DIR between *Bing* and *SUN* are more affected by noisy source data. This is also observed when comparing DANN and TCL. Compared to Co+ATDA, ATDA is slightly better than Co+ATDA. This abnormal phenomenon can be explained using Δ (see Section 5.4), after using Co-teaching to assign pseudo labels for noisy source data, the second term in Δ_s may increase, which results in that Δ increases, i.e., noise effects actually increase. This phenomenon is an evidence that a two-step method may not really reduce noise effects.

5.7.8 Ablation Study

Finally, we conduct thorough experiments to show the contribution of individual components in B-Net. We report average target-domain accuracy on 32 simulated WUDA tasks (8 digit and 24 human-sentiment WUDA tasks) and 3 real-world WUDA tasks. We consider following baselines:

- Tri-C-Net: triply check data in SD, MD and TD. Compared to B-Net, Tri-C-Net has another branch (denoted by Branch-III) to check data in SD. Namely, Tri-C-Net has three branches (i.e., six networks). Parameters of CNN of the Branch-III are the same with that of Branch-I and Branch-II.
- B w/o C: train B-Net by Algorithm 5.1, without adding $|\theta_{f11}^T \theta_{f21}|$ into the loss function of B-Net.
- DCP-D: realize DCP via Decoupling [Malach and Shalev-Shwartz, 2017] to check data in MD and TD.
- DCP-M: realize DCP via MentorNet [Jiang et al., 2018] to check data in MD and TD.
- B-Net-S: train B-Net where the check is turned on for Source data in MD.
- B-Net-T: train B-Net where the check is turned on for Target data in TD.
- B-Net-ST: train B-Net where the checks are turned on for Source data in MD and Target data in TD.
- B-Net-M: train B-Net where the check is turned on for all data in MD.

Note that in the full B-Net, the checks are turned on for all data in MD and TD. Comparing B-Net with Tri-C-Net shows whether two branches (i.e., four networks) are the optimal design. Comparing B-Net with B w/o C reveals if the constraint $|\theta_{f11}^T \theta_{f21}|$ takes effects. Comparing B-Net with DCP-D and DCP-M shows whether realizing DCP via Co-teaching is the optimal way. Comparing B-Net with B-Net-S, B-Net-T, B-Net-ST and B-Net-M reveals if DCP is necessary.

Table 5.5 reports average target-domain accuracy of above baselines and B-Net. As can be seen, 1) maintaining 4 networks (like B-Net) is better than maintaining 6 networks (like Tri-C-Net) since B-Net outperforms Tri-C-Net in terms of average target-domain accuracy; 2) B-Net benefits from adding the constraint to the loss function $\mathcal{L}$; 3) realizing DCP by Co-teaching is better than using Decoupling or MentorNet; and 4) DCP is necessary since accuracy of B-Net is higher than those of B-Net-S, B-Net-T, B-Net-ST and B-Net-M.

Table 5.5: Results of ablation study. Average target-domain accuracy on 8 simulated digit WUDA tasks (*Digit*), 24 simulated human-sentiment WUDA tasks (*Sentiment*) and 3 real-world WUDA tasks (*Real-world*). Bold value represents the highest accuracy in each row.

Datasets	Tri-C-Net	B w/o C	DCP-D	DCP-M	B-Net-S	B-Net-T	B-Net-ST	B-Net-M	B-Net
Digit	59.80%	74.52%	59.19%	70.85%	71.93%	52.00%	72.27%	73.89%	**75.82%**
Sentiment	61.25%	63.57%	61.37%	63.39%	61.49%	61.12%	61.73%	62.21%	**63.77%**
Real-world	61.50%	62.27%	59.82%	62.34%	61.91%	60.87%	62.24%	62.17%	**62.42%**

5.8 Summary

This chapter opens a new problem called *wildly unsupervised domain adaptation* (WUDA), and theoretically and empirically shows that existing methods cannot handle WUDA. To address this problem, a robust one-step approach called *Butterfly* is proposed. Butterfly maintains four deep networks simultaneously: Two take care of all adaptations; while the other two can focus on classification in target domain. We compare Butterfly with existing HoUDA methods on 32 simulated and 3 real-world WUDA tasks. Empirical results demonstrate that Butterfly can robustly transfer knowledge from noisy source data to unlabeled target data.

CHAPTER 6

HETEROGENEOUS DOMAIN ADAPTATION: AN UNSUPERVISED APPROACH AND ITS THEORETICAL GUARANTEES

6.1 Introduction

Given the time and cost associated with human labeling, many domains (datasets) are unlabeled in current era, which means the existing (semi-)supervised heterogeneous domain adaptation methods do not perform well on the majority of target domains. Namely, developing a valid *heterogeneous unsupervised domain adaptation* (HeUDA) method is very important in the field of domain adaptation. The aim of this chapter is to establish a theoretical foundation for HeUDA methods that predict labels for a heterogeneous and unlabeled target domain without parallel sets. We are motivated by the observation that two

heterogeneous domains may come from one domain. Namely, features of two heterogeneous domains could be outputs of heterogeneous projections of features of the one domain (see Figure 6.1). In the following two paragraphs, we present two examples to describe this observation.

Example 1. Sentences written in Latin can be translated into sentences written in French and Spanish. The French and Spanish sentences have different representations but share a similar meaning. If the Latin sentences are labeled as “positive", then the French and Spanish sentences are probably labeled as “positive”. In this example, we can construct a Latin domain using Latin sentences and the task (labeling sentences as “positive" or “negative"). Then, French domain and Spanish domain come from one domain: Latin domain, where French and Spanish domains consist of French and Spanish sentences (translated from Latin sentences) and the task (labeling sentences as “positive" or “negative").

Example 2. Taking another example in real-world scenarios: human sentiment, as an underlying domain (to analyze whether a person is happy), is difficult to record accurately. We can only obtain its projection or representation on real events, such as Amazon product reviews and Rotten Tomatoes movie reviews. The Amazon product reviews and the Rotten Tomatoes movie reviews are two heterogeneous domains but come from an underlying domain: human sentiment.

Based on this observation, we propose two key factors, V and D, to reveal the similarity between two heterogeneous domains:

- the variation (V) between the conditional probability density functions of both domains;

- the distance (D) between the feature spaces of the two heterogeneous domains.

In general, small V means that two domains have similar ground-truth labeling functions and small D means that two feature spaces are close.

This chapter aims to construct *homogeneous representations* to preserve the original similarity (evaluated by V and D) between two heterogeneous domains, while allowing knowledge to be transferred. We denote V_{He}, V_{Ho}, D_{He} and D_{Ho} by values of V and D of the original *heterogeneous* (He) domains and the *homogeneous* (Ho) representations. The basic assumption of unsupervised domain adaptation methods is that two domains have similar ground-truth labeling functions. Hence, the constructed homogeneous representations must make $V_{Ho} \leq V_{He}$. Similarly, $D_{Ho} \leq D_{He}$ is expected, indicating that the distance between two feature spaces of the homogeneous representations is small. We mainly focus on how to construct the homogeneous representations where $V_{Ho} = V_{He}$ and $D_{Ho} = D_{He}$ (the exact homogeneous representations of two heterogeneous domains).

To ensure the efficacy of the homogeneous representations, this chapter presents: 1) an unsupervised knowledge transfer theorem that guarantees the correctness of transferring knowledge (to make $V_{Ho} = V_{He}$); and 2) a principal angle-based metric to measure the distance between two pairs of domains: one pair comprises the original source and target domains and the other pair comprises two homogeneous representations of two domains (to help make $D_{Ho} = D_{He}$). Based on the constructed exact homogeneous representations of two heterogeneous domains, HoUDA methods can be applied to transfer knowledge across

the representations.

The unsupervised knowledge transfer theorem sets out the transfer conditions necessary to prevent negative transfer (to make $V_{Ho} = V_{He}$). *Linear monotonic maps* (LMMs) meet the transfer conditions of the theorem and are therefore used to construct the homogeneous representations. Rather than directly measuring the distance between two heterogeneous feature spaces, the distance between two feature subspaces of different dimensions is measured using the principal angles of Grassmann manifold. This new distance metric reflects the extent to which the homogeneous representations have preserved the geometric relationship between the original heterogeneous domains (to make $D_{Ho} = D_{He}$). It is defined on two pairs of subspace sets; one pair of subspace sets reflects the original domains, the other reflects the homogeneous representations.

Homogeneous representations of the heterogeneous domains are constructed by minimizing the distance metric based on the constraints associated with LMMs, i.e., minimize $\|D_{Ho} - D_{He}\|_{\ell_1}$ under the constraints $V_{Ho} = V_{He}$. Knowledge is transferred between the domains through the homogeneous representations via a *geodesic flow kernel* (GFK) [Gong et al., 2014]. The complete proposed HeUDA method incorporates all these elements and is called the Grassmann-LMM-GFK method - GLG for short. Figure 6.1 illustrates the process of GLG.

To validate the efficacy of GLG, five public datasets were reorganized into ten tasks across three applications: cancer detection, credit assessment, and text classification. The experimental results reveal that the proposed method can reliably transfer knowledge across two heterogeneous domains when the target domain is unlabeled and there are no parallel sets. The main contributions of this chapter are:

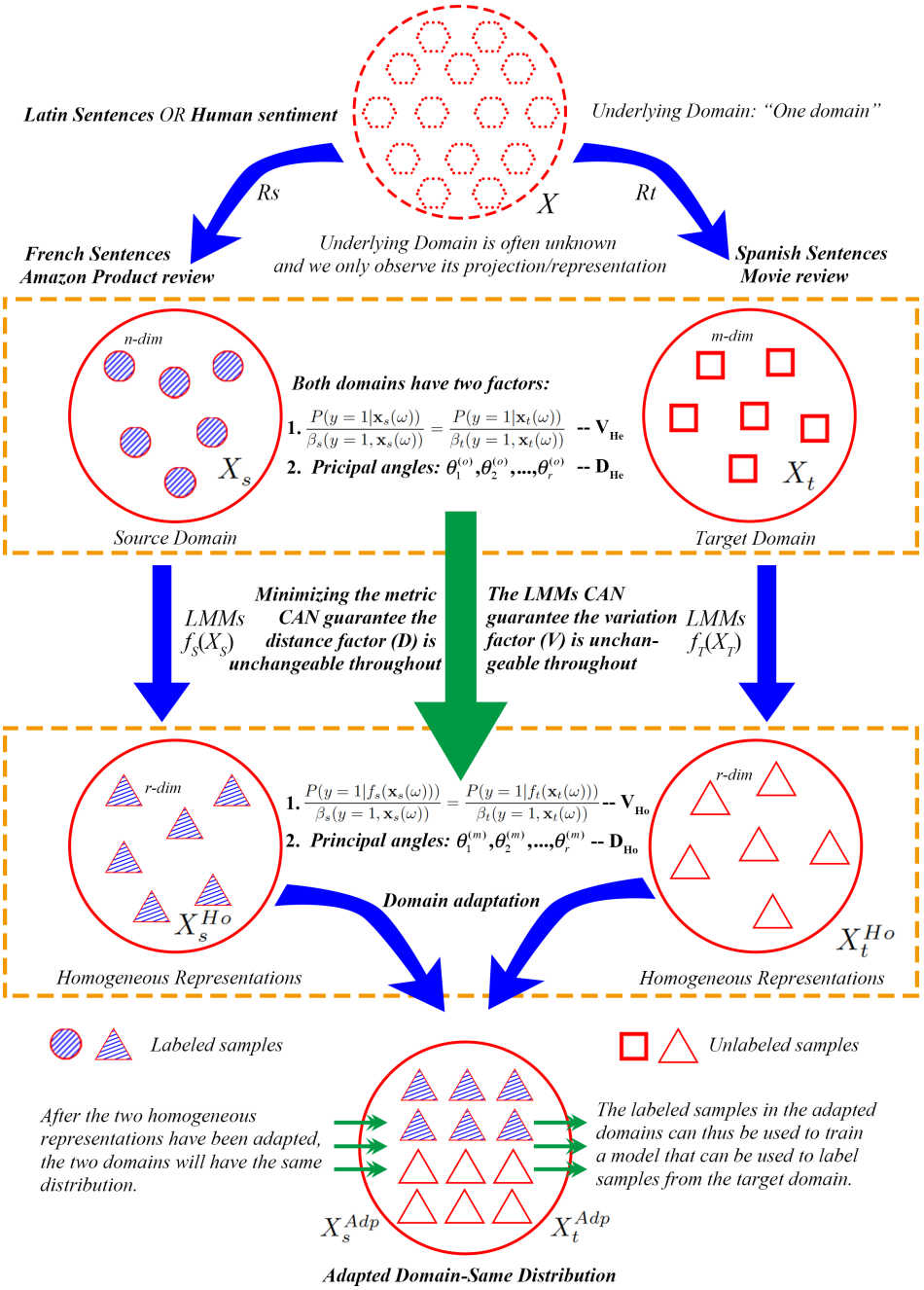


Figure 6.1: The progress of the GLG method. The original source and target domains come from the same underlying domain (e.g., classifying Latin sentences or analyzing human sentiment). However, the underlying domain is hard to observe and we can only observe its projection/representation on two (or more) domains, e.g., two heterogeneous domains in this figure.

1) An effective heterogeneous unsupervised domain adaptation method, called GLG, is proposed to transfer knowledge from a source domain to an unlabeled target domain in settings where both domains have heterogeneous feature spaces and are free of parallel sets.

2) An unsupervised knowledge transfer theorem that prevents negative transfer for HeUDA methods is proved.

3) A new principal angle based metric is proposed to show the extent to which homogeneous representations have preserved the geometric distance between the original domains. This new metric also reveals the relationship between two heterogeneous (different-dimensionality) feature spaces.

Proofs of lemmas and theorems can be found in the Appendix A.4.

6.2 Problem Setting and Notations

Following our motivation (two heterogeneous domains may come from one domain), we first give a distribution $\mathcal{P}$ over a multivariate random variable $\mathbf{X}$ defined on an instance set $\mathcal{X}$, $\mathbf{X}: \mathcal{X} \to \mathbb{R}^k$ and a labeling function $f: \mathbb{R}^k \to [0,1]$. The value of $f(\mathbf{X})$ corresponds to the probability that the label of $\mathbf{X}$ is 1. In this chapter, we use ω to represent a subset of $\mathcal{X}$, i.e. $\omega \subset \mathcal{X}$, and use $P(\mathbf{Y}=1|\mathbf{X})$ to represent $f(\mathbf{X})$, where $\mathbf{Y}$ is the label of $\mathbf{X}$ and the value of $\mathbf{Y}$ is -1 or 1. The multivariate random variables corresponding to features of two heterogeneous domains are images of $\mathbf{X}$:

$$\mathbf{X_s} = R_s(\mathbf{X}), \ \ \mathbf{X_t} = R_t(\mathbf{X}), \tag{6.1}$$

where $R_s: \mathbb{R}^k \to \mathbb{R}^m$, $R_t: \mathbb{R}^k \to \mathbb{R}^n$, $\mathbf{X_s} \sim \mathcal{P}_s$ and $\mathbf{X_t} \sim \mathcal{P}_t$. In the heterogeneous unsupervised domain adaptation setting, $m \neq n$ and we can observe a source

domain $\mathbf{D_s} = \{(x_{si}, y_{si})\}_{i=1}^{N}$ and a target domain $\mathbf{D_t} = \{(x_{ti})\}_{i=1}^{N}$, where $x_{si} \in \mathbb{R}^m$, $x_{ti} \in \mathbb{R}^n$ are observations of the multivariate random variables $\mathbf{X_s}$ and $\mathbf{X_t}$, respectively, and y_{si}, taking value from $\{-1, 1\}$, is the label of x_{si}. $X_s = \{(x_{si})\}_{i=1}^{N}$ builds up a feature space of $\mathbf{D_s}$ and $X_t = \{(x_{ti})\}_{i=1}^{N}$ builds up a feature space of $\mathbf{D_t}$ and $Y_s = \{(y_{si})\}_{i=1}^{N}$ builds up of a label space of $\mathbf{D_s}$. In the following section, $\mathbf{D_s} = (X_s, Y_s)$ and $\mathbf{D_t} = (X_t)$ for short. The HeUDA problem is how to use $\mathbf{D_s}$ and $\mathbf{D_t}$ to label each x_{ti} in $\mathbf{D_t}$.

In the Example 1 (see Section 6.1), $\mathcal{X}$ represents sentences written in Latin and ω is a subset to collect some Latin sentences from $\mathcal{X}$. $\mathbf{X}$ is a multivariate random variable and represents the Latin representations of sentences in $\mathcal{X}$. Since we consider that French and Spanish sentences are translated from Latin sentences, $\mathbf{X_s}$ is the French representations of sentences in $\mathcal{X}$ and $\mathbf{X_t}$ be the Spanish representations of sentences in $\mathcal{X}$. It should be noted that, in general, Latin sentences and French (or Spanish) sentences are disjoint. However, in this example, French (or Spanish) sentences are translated from Latin sentences, which means that French (or Spanish) sentences and Latin sentences are associated.

6.3 Heterogeneous Unsupervised domain adaptation

Our HeUDA method, called GLG, is built around an unsupervised knowledge transfer theorem that avoids negative transfer through a variation factor V that measures the difference between the conditional probability density functions in both domains. The unsupervised knowledge transfer theorem guarantees

linear monotonic maps (LMMs) against negative transfer once used to construct homogeneous representations of the heterogeneous domains (because $V_{Ho} = V_{He}$). A metric, which reflects the distance between the original domains and the homogeneous representations, ensures that the distance factor D_{He} between the original domains is preserved (i.e., $D_{Ho} = D_{He}$). Thus, the central premise of the GLG method is to find the best LMM such that the distance between the original domains is preserved.

6.3.1 Unsupervised Knowledge Transfer Theorem for HeUDA

This subsection first presents the relationships between $P(\mathbf{Y} = 1|\mathbf{X})$ and $P(\mathbf{Y} = 1|\mathbf{X_s})$ (or $P(\mathbf{Y} = 1|\mathbf{X_t})$) and then gives the definition of the variation factor (V) between $P(\mathbf{Y} = 1|\mathbf{X_s})$ and $P(\mathbf{Y} = 1|\mathbf{X_t})$. Based on V, we propose the unsupervised knowledge transfer theorem for HeUDA.

Given a measurable subset $\omega \subset \mathcal{X}$, we can obtain the probability $c(\omega) = P(\mathbf{Y} = 1|\mathbf{X}(\omega))$. We expect that the probability $P(\mathbf{Y} = 1|R_s(\mathbf{X}(\omega)))$ and $P(\mathbf{Y} = 1|R_t(\mathbf{X}(\omega)))$ will be around $c(\omega)$. If ω is regarded as the Latin sentences mentioned in Section 6.1, $\mathbf{X}_s = R_s(\mathbf{X}(\omega))$ and $\mathbf{X}_t = R_t(\mathbf{X}(\omega))$ are French and Spanish representations of the Latin sentences. If the Latin sentences are labeled as “positive" ($\mathbf{Y} = 1$), we of course expect that the French and Spanish sentences will have a high probability of being labeled as “positive". To ensure this, $\forall \omega \subset \mathcal{X}$, we assume the following equality holds.

$$\frac{P(\mathbf{Y} = 1|\mathbf{X}_s(\omega))}{\beta_s(\mathbf{Y} = 1, \mathbf{X}_s(\omega))} = \frac{P(\mathbf{Y} = 1|\mathbf{X}_t(\omega))}{\beta_t(\mathbf{Y} = 1, \mathbf{X}_t(\omega))} = c(\omega), \tag{6.2}$$

where $\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega))$ and $\beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega))$ are two real-value functions. Since two heterogeneous domains have a similar task (i.e., labeling sentences as “positive" or “negative"), we know $\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega))$ and $\beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega))$ should be around 1 and have following properties for any ω.

$$\begin{aligned} &\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega)) \neq \frac{1-c(\omega)}{c(\omega)} \\ &\text{or } \ \beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega)) \neq \frac{1-c(\omega)}{c(\omega)}. \end{aligned} \tag{6.3}$$

The properties described in (6.3) ensure that it is beneficial to transfer knowledge from the source domain to the target domain. If we do not have both properties described in (6.3), i.e., $\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega)) = (1-c(\omega))/c(\omega)$, we will have $P(\mathbf{Y}=1|\mathbf{X}_s(\omega)) = 1-c(\omega) = P(\mathbf{Y}=-1|\mathbf{X}(\omega))$, indicating that positive Latin sentences are represented by negative French sentences. Based on (6.2), we define the variation factor $V_{He}(P(\mathbf{Y}=1|\mathbf{X}_s(\omega)),P(\mathbf{Y}=1|\mathbf{X}_t(\omega)))$ as follows.

$$\begin{aligned} V_{He}(P(\mathbf{Y}=1|\mathbf{X}_s(\omega)),P(\mathbf{Y}=1|\mathbf{X}_t(\omega))) &= \left|P(\mathbf{Y}=1|\mathbf{X}_s(\omega)) - P(\mathbf{Y}=1|\mathbf{X}_t(\omega))\right| \\ &= c(\omega)\left|\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega)) - \beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega))\right|. \end{aligned} \tag{6.4}$$

To study how to correctly transfer knowledge across two heterogeneous domains, we first give a definition of extreme negative transfer to show the worst case.

Definition 6.1 (Extreme negative transfer)**.** *Given* $\mathbf{X_s} \sim \mathcal{P}_s$, $\mathbf{X_t} \sim \mathcal{P}_t$ *and Eq.* (6.2)*, if,* $\forall\omega \subset \mathcal{X}$, $f_s(\mathbf{X_s}) : \mathbb{R}^m \to \mathbb{R}^r$ *and* $f_t(\mathbf{X_t}) : \mathbb{R}^n \to \mathbb{R}^r$ *satisfy*

$$P(\mathbf{Y}=1|f_s(\mathbf{X}_s(\omega)) = P(\mathbf{Y}=-1|f_t(\mathbf{X}_t(\omega)),$$

then we call that $f_s(\mathbf{X_s})$ *and* $f_t(\mathbf{X_t})$ *cause extreme negative transfer.*

Based on Definition 6.1, if extreme negative transfer happens, we will transfer incorrect knowledge across two domains. In experiments, we can use target-domain classification accuracy to quantify extreme negative transfer: lower accuracy means that extremer negative transfer happens. Section 6.5.3 shows the consequence caused by extreme negative transfer.

However, we cannot quantify extreme negative transfer without presence of labeled data in target domain. Thus, to avoid the extreme negative transfer in advance, we present the heterogeneous unsupervised domain adaptation condition as follows. Satisfying this condition means that the knowledge will be transferred in expected way.

Definition 6.2 (HeUDA condition)**.** *Given* $\mathbf{X_s} \sim \mathcal{P}_s$, $\mathbf{X_t} \sim \mathcal{P}_t$ *and the Eq.* (6.2)*, if there are two maps* $f_s(\mathbf{X_s})$ *and* $f_t(\mathbf{X_t})$*, then,* $\forall \omega \subset \mathcal{X}$*, the heterogeneous unsupervised domain adaptation condition can be expressed by the following equation.*

$$\frac{P(\mathbf{Y}=1|f_s(\mathbf{X}_s(\omega)))}{\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega))} = \frac{P(\mathbf{Y}=1|f_t(\mathbf{X}_t(\omega)))}{\beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega))} = c(\omega), \tag{6.5}$$

where ω *is a measurable set.*

If this condition is satisfied, it is clear that

$$P(\mathbf{Y}=1|f_s(\mathbf{X}_s(\omega)) \neq P(\mathbf{Y}=-1|f_t(\mathbf{X}_t(\omega)),$$

and

$$V_{Ho}(P(\mathbf{Y}=1|f_s(\mathbf{X}_s(\omega)),P(\mathbf{Y}=1|f_t(\mathbf{X}_t(\omega))) = c(\omega)\left|\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega)) - \beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega))\right|,$$

indicating that f_s and f_t will not cause extreme negative transfer and $V_{He} = V_{Ho}$.

Remark 6.1. *The HeUDA condition defined in Definition 6.2 is a sufficient condition to correctly transfer knowledge across two heterogeneous domains, but it is not a necessary condition. Although we could define more HeUDA conditions (sufficient conditions) to correctly transfer knowledge across two heterogeneous domains, we cannot find maps to satisfy every condition. In this chapter, the HeUDA condition described in Definition 6.2 can be satisfied by the proposed mapping function: linear monotonic map (defined in Section 6.3.3), which means that we find a practical way to correctly transfer knowledge across two heterogeneous domains.*

Although Definition 6.2 provides the basic transfer condition in HeUDA scenario, we still need to determine which kinds of map (i.e., f_s and f_t) satisfy this condition. To explore one such map, we propose monotonic maps as follows:

Definition 6.3 (monotonic map)**.** *If a map $f : \mathbb{R}^m \to \mathbb{R}^r$ satisfies the following condition*

$$x_i < x_j \Rightarrow f(x_i) < f(x_j),$$

where $(x_i, <)$ and $(f(x_i), <)$ are binary relations and "$<$" is a strict partial order over $\mathbb{R}^m$ and $f(\mathbb{R}^m)$, then the map f is a monotonic map.

The proposed unsupervised knowledge transfer theorem follows, based on Definition 6.3.

Theorem 6.1 (unsupervised knowledge transfer theorem)**.** *Given $\mathbf{X_s} \sim \mathcal{P}_s$, $\mathbf{X_t} \sim \mathcal{P}_t$ and the Eq.* (6.2)*, if there are two maps $f_s(\mathbf{X_s}) : \mathbb{R}^m \to \mathbb{R}^r$ and $f_t(\mathbf{X_t}) : \mathbb{R}^n \to \mathbb{R}^r$ satisfy that*

1) $f_s(\mathbf{X_s})$ and $f_t(\mathbf{X_t})$ are monotonic maps;

2) $f_s^{-1}(f_s(\mathbf{X_s})) = \mathbf{X_s}$ *and* $f_t^{-1}(f_t(\mathbf{X_t})) = \mathbf{X_t}$*;*

then $f_s(\mathbf{X_s})$ *and* $f_t(\mathbf{X_t})$ *satisfy the heterogeneous unsupervised domain adaptation conditions.*

Based on Theorem 6.1, we demonstrate a choice $f_s(\mathbf{X_s})$ and $f_t(\mathbf{X_t})$ to satisfy the heterogeneous unsupervised domain adaptation condition, and highlight the sufficient conditions for reliable unsupervised knowledge transfer. If a mapping function from heterogeneous domains to homogeneous representations satisfies two conditions in Theorem 6.1, it can transfer knowledge across domains with theoretical reliability.

6.3.2 Principal Angle-based Measurement between Heterogeneous Feature Spaces

In this subsection, the method for measuring the distance between two subspaces is introduced. On a Grassmann manifold $G_{N,m}$ (or $G_{N,n}$), subspaces with m (or n) dimensions of $\mathbb{R}^N$ are regarded as points in $G_{N,m}$ (or $G_{N,n}$). This means that measuring the distance between two subspaces can be calculated by the distance between those two points on the Grassmann manifold. First, the subspaces spanned by X_s and X_t are confirmed using singular value decomposition (SVD). The distance between the spanned subspaces $A = span(X_s)$ and $B = span(X_t)$ can then be calculated in terms of the corresponding points on the Grassmann manifold.

There are two HoUDA methods that use a Grassmann manifold in this way: DAGM and GFK. DAGM was proposed by Gopalan et al. [Gopalan et al., 2014]. GFK was proposed by Gong and Grauman [Gong et al., 2014]. Both have one

shortcoming: the source domain and the target domain must have feature spaces of the same dimension, mainly due to the lack of geodesic flow on $G_{N,m}$ and $G_{N,n}$ ($m \neq n$). In [Ye and Lim, 2016], Ye and Lim successfully proposed the principal angles between two different dimensional subspaces, which helps measure the distance between two heterogeneous feature spaces consisting of X_s and X_t. Principal angles for heterogeneous subspaces are defined as follows.

Definition 6.4 (principal angles for heterogeneous subspaces [Ye and Lim, 2016])**.** *Given two subspaces $A \in G_{N,m}$ and $B \in G_{N,n}$ ($m \neq n$), which form the matrixes $A \in \mathbb{R}^{N\times m}$ and $B \in \mathbb{R}^{N\times n}$, the i^{th} principal vectors (p_i, q_i), $i = 1,\ldots,r$, are defined as solutions for the optimization problem ($r = \min(n,m)$):*

$$\begin{aligned} &\max \ \ p^T q \\ &s.\,t. \ \ p \in A,\ p^T p_1 = \ldots = p^T p_{i-1},\ \|p\| = 1, \\ &\qquad q \in B,\ q^T q_1 = \ldots = q^T q_{i-1},\ \|q\| = 1, \end{aligned} \tag{6.6}$$

Then, the principal angles for heterogeneous subspaces are defined as

$$cos\theta_i = p_i^T q_i,\ i = 1,\ldots,r.$$

Ye and Lim [Ye and Lim, 2016] proved that the optimization solution to (6.6) can be computed using SVD. Thus, we can calculate the principal angles between two different-dimensionality subspaces, and this idea forms the distance factor D mentioned in Section 6.1. To perfectly define distances between subspaces of different dimensions, Ye and Lim used two Schubert varieties to prove that all the defined distances in subspaces of the same dimensions are also correct when the dimensionalities differ. This means we can calculate a distance between two subspaces of different dimensions using the principal angles defined in Definition

6.4. Given $A = span(X_s)$ and $B = span(X_t)$, the distance vector between X_s and X_t is defined as a vector containing principal angles between A and B, which has the following expression.

$$D_{He}(X_s, X_t) = arccos([\sigma_1(A^T B), \sigma_2(A^T B), ..., \sigma_r(A^T B)]),$$

where $r = \min(n, m)$, $\sigma_i(A^T B)$ is the i^{th} singular value of $A^T B$ computed by SVD (the i^{th} principal angles $\theta_i = arccos(\sigma_i(A^T B))$).

If we can find two maps f_s and f_t that satisfy the conditions of Theorem 6.1, we can obtain the D_{Ho} as follows.

$$\begin{aligned} & D_{Ho}(f_s(X_s), f_t(X_t)) \\ = & arccos([\sigma_1(C^T D), \sigma_2(C^T D), ..., \sigma_r(C^T D)]), \end{aligned}$$

where $C = span(f_s(X_s))$ and $D = span(f_t(X_t))$. Hence, we can measure the distance between D_{He} and D_{Ho} via these singular values of matrix $A^T B$ and $C^T D$.

Remark 6.2. *The distance $D_{He}(X_s, X_t)$ defined in this subsection aims to describe a geometric relationship between X_s and X_t. Compared to KL divergence, which estimates the distance between probability distributions, $D_{He}(X_s, X_t)$ has the following differences.*

a) $D_{He}(X_s, X_t)$ is a vector that contains principal angles between a subspace spanned by X_s and a subspace spanned by X_t, which means that it describes a geometric relationship between X_s and X_t. However, KL divergence is a real number to describe a relationship between X_s and X_t from a probability perspective, so, $D_{He}(X_s, X_t)$ and KL divergence have different aims.

b) $D_{He}(X_s, X_t)$ is able to describe a geometric relationship between X_s and X_t when X_s and X_t have different dimensionalities (e.g., the dimensionality

of X_s is 24 and the dimensionality of X_t is 14). However, KL divergence can only be computed when X_s and X_t have the same dimensionalities (e.g., the dimensionality of X_s is 14 and the dimensionality of X_t is 14). This is why it is necessary to define a new distance to describe the relationship between two heterogeneous feature spaces from two heterogeneous domains. To the best of our knowledge, there is little discussion about the relationship between two different-dimensionality distributions.

6.3.3 GLG: The Proposed HeUDA Method

With the unsupervised knowledge transfer theorem that ensures the reliability of heterogeneous unsupervised domain adaptation, and with the principal angles of Grassmann manifolds explained, we now turn to the proposed method, GLG. The optimization solution for GLG is outlined in Section 6.4.

A common idea for finding the homogeneous representations of heterogeneous domains is to find maps that can project feature spaces of different dimensions (heterogeneous domains) onto feature spaces with same dimensions. However, most heterogeneous domain adaptation methods require at least some labeled instances or paired instances in the target domain to maintain the relationship between the source and target domains. Thus, the key to a HeUDA method is to find a few properties that can be maintained between the original domains and the homogeneous representations.

Here, these two factors are the variation factor (V_{He} and V_{Ho} defined in Section 6.3.1) and the distance factor (D_{He} and D_{Ho} defined in Section 6.3.2). Theorem 6.1 determines the properties the maps should satisfy to make $V_{He} = V_{Ho}$

and principal angles shows the distance between two heterogeneous (or homogeneous) feature spaces (D_{He} and D_{Ho}). However, there are still two concerns: 1) which type of mapping function is suitable for Theorem 6.1; and 2) which properties should the map maintain between the original domains and the homogeneous representations. The first concern with the unsupervised knowledge transfer theorem is addressed by selecting LMMs as the map of choice.

Lemma 6.1 (linear monotonic map)**.** *Given a map $f : \mathbb{R}^m \to \mathbb{R}^r$ with form $f(x) = xU^T$, $f(x)$ is a monotonic map if and only if $U > 0$ or $U < 0$, where $x \in \mathbb{R}^m$ and $U \in \mathbb{R}^{r \times m}$.*

Since the defined map in Lemma 6.1 only uses U and according to the generalized inverse of a matrix, the matrix $f(X_s)$ satisfies $f^{-1}(f(X_s)) = X_s$. Therefore, we can prove that LMMs satisfy the conditions in Theorem 6.2.

Theorem 6.2 (LMM for HeUDA)**.** *Given $\mathbf{X_s} \sim \mathcal{P}_s$, $\mathbf{X_t} \sim \mathcal{P}_t$ and Eq.* (6.2)*, if there are two maps $f_s(\mathbf{X_s}) : \mathbb{R}^m \to \mathbb{R}^r$ and $f_t(\mathbf{X_t}) : \mathbb{R}^n \to \mathbb{R}^r$ are LMMs, then $f_s(\mathbf{X_s})$ and $f_t(\mathbf{X_t})$ satisfy the HeUDA condition.*

Remark 6.3. *From this theorem and the nature of LMMs, we know this positive map can better handle datasets that have many monotonic samples because the probabilities in these monotonic samples can be preserved without any loss. The existence of these samples offers the greatest probability of preventing negative transfers.*

Theorem 6.2 addresses the first concern and provides a suitable map, such as the map in Lemma 6.1, to project two heterogeneous feature spaces onto the same dimensional feature space. It is worthwhile showing that an LMM is just

one among many suitable maps for Theorem 3.1. A nonlinear map can also be used to construct the map, as long as the map is monotonic. In future work, we intend to explore additional maps suitable for other HeUDA methods.

This brings us to the second concern: which properties can be maintained during the mapping process between the original domains and the homogeneous representations? As mentioned above, the principal angles play a significant role in defining the distance between two subspaces on a Grassmann manifold, and in explaining the projection between them [Wong, 1967]. Ensuring the principal angles remain unchanged is thus one option for maintaining some useful properties.

Specifically, for any two pairs of subspaces (A,B) and (C,D), if the principal angles of (A,B) and (C,D) are the same (implying that $\min\{\dim(A), \dim(B)\} = \min\{\dim(C), \dim(D)\}$, $\dim(A)$ represents the dimension of A), then the relationship between A and B can be regarded as similar to the relationship between C and D. Based on this idea, the definition of measurement $\mathcal{D}$, which describes the relationships between two pairs of subspaces, follows.

Definition 6.5 (measurement between subspace pairs)**.** *Given two pairs of subspaces (A,B) and (C,D), the measurement $\mathcal{D}((A,B),(C,D))$ between (A,B) and (C,D) is defined as*

$$\mathcal{D}((A,B),(C,D)) = \sum_{i=1}^{r} \left| \sigma_i(A^T B) - \sigma_i(C^T D) \right|, \tag{6.7}$$

where A,B,C and D are subspaces in $\mathbb{R}^N$, r=min{dim(A), dim(B), dim(C), dim(D)} and $\sigma_i(A^T B)$ is the i^{th} singular value of matrix $A^T B$ and represents the cosine value of the i^{th} principal angle between A and B.

Remark 6.4. *The measurement $\mathcal{D}$ is defined on two pairs of two subspaces (e.g., pair 1: (A,B) and pair 2: (C,D), where A,B,C and D are subspaces) rather than two distributions. This distance describes the distance between two pairs of subspaces (e.g., relationships between (A,B) and (C,D)), which is different with distance between probability distributions, such as KL divergence.*

Measurement $\mathcal{D}$ defined on $G_{N,*}^T \times G_{N,*}$ is actually a metric, as proven in the following theorem.

Theorem 6.3. *$(\mathcal{D}, G_{N,*}^T \times G_{N,*})$ is a metric space, where $G_{N,*} = \{A | A \in G_{N,i},\ i = 1,...,N-1\}$.*

In contrast to the metric proposed in [Ye and Lim, 2016], our metric focuses on the distance between two pairs of subspaces, such as (A,B) and (C,D), rather than two subspaces, such as A and B. The proposed metric, especially designed for the HeUDA problem, shows the extent to which homogeneous representations have preserved the geometric distance between two heterogeneous feature spaces. However, the metric proposed in [Ye and Lim, 2016] only focuses on the distance between two subspaces, such as A and B. The definition of the consistency of the geometric relationship with respect to the feature spaces of two domains can be given in terms of the metric $\mathcal{D}$ as follows.

Definition 6.6 (consistency of the geometric relationship)**.** *Given the source domain $\mathbf{D_s} = (X_s, Y_s)$ and the heterogeneous and unlabeled target domain $\mathbf{D_s} = (X_t)$, let $f_s(X_s) = X_s U_s^T$ and $f_t(X_t) = X_t U_t^T$, if $\forall \delta \in (0, \delta_0]$, $\exists \epsilon < \mathcal{O}(\delta_0)$ such that*

$$\int_0^{\delta_0} \mathcal{D}\Big((S_{X_s^\delta}, S_{X_t^\delta}), (S_m(f_s, X_s^\delta), S_m(f_t, X_t^\delta))\Big) d\delta < \epsilon, \tag{6.8}$$

then we can say that (X_s, X_t) *and* $(f_s(X_s), f_t(X_t))$ *have consistent geometric relationship, where* $S_{X^\delta} = span(X + \delta \cdot \mathbf{1}_X)$, $S_m(f, X^\delta) = span(f(X + \delta \cdot \mathbf{1}_X))$, $U_s \in \mathbb{R}^{r \times m}$, $U_t \in \mathbb{R}^{r \times n}$, $r = min\{m, n\}$ *and* $\mathbf{1}_X$ *is a matrix of ones of the same size as* X.

This definition precisely demonstrates how f_s and f_t influence the geometric relationship between the original feature spaces and the feature spaces of homogeneous representations. If there are slight changes in the original feature spaces, we hope feature spaces of the homogeneous representations will also see slight changes. If they do, it means that the feature spaces of the homogeneous representations are consistent with the geometric relationships of the two original feature spaces. Based on definitions of D_{He} and D_{Ho}, (6.8) is expressed by

$$\begin{aligned} &\int_0^{\delta_0} \mathcal{D}\Big((S_{X_s^\delta}, S_{X_t^\delta}), (S_m(f_s, X_s^\delta), S_m(f_t, X_t^\delta)\Big) d\delta < \epsilon \\ \Leftrightarrow &\int_0^{\delta_0} \left\| D_{He}(X_s^\delta, X_t^\delta) - D_{Ho}(f_s(X_s^\delta), f_t(X_t^\delta)) \right\|_{\ell_1} d\delta < \epsilon. \end{aligned} \tag{6.9}$$

To ensure the consistency of the geometric relationship of the two original feature spaces, we minimize the following cost function to ensure that we are able to find an ϵ that is less than $\mathcal{O}(\delta_0)$, such that $\int_0^{\delta_0} \mathcal{D}\big((S_{X_s^\delta}, S_{X_t^\delta}), (S_m(f_s, X_s^\delta), S_m(f_t, X_t^\delta)\big) d\delta < \epsilon$ when there are slight changes $\delta \in (0, \delta_0]$ in the original feature spaces.

Definition 6.7 (cost function I)**.** *Given the source domain* $\mathbf{D_s} = (X_s, Y_s)$ *and the heterogeneous and unlabeled target domain* $\mathbf{D_s} = (X_t)$*, let* $f_s(X_s) = X_s U_s^T$ *and* $f_t(X_t) = X_t U_t^T$*, the cost function* J_1 *of GLG is defined as*

$$\begin{aligned} J_1(X_s, X_t; U_s, U_t) = &\int_0^{\delta_0} \left\| D_{He}(X_s^\delta, X_t^\delta) - D_{Ho}(X_s^\delta, X_t^\delta) \right\|_{\ell_1} d\delta \\ &+ \frac{1}{2}\lambda_s Tr(U_s U_s^T) + \frac{1}{2}\lambda_t Tr(U_t U_t^T), \end{aligned} \tag{6.10}$$

where $X^{\delta} = X + \delta \cdot \mathbf{1}_X$, $U_s \in \mathbb{R}^{r \times m}$, $U_t \in \mathbb{R}^{r \times n}$, $r = min\{m,n\}$ *and* $\mathbf{1}_X$ *is a matrix of ones of the same size as* X.

This definition shows the discrepancy between the original feature spaces and the feature spaces of the homogeneous representations via principal angles. If we use $\theta_i^{(o)}$ to represent the i^{th} principal angle of the original feature spaces and $\theta_i^{(m)}$ to represent the i^{th} principal angle of the feature spaces of the homogeneous representations, J_1 measures $|cos(\theta_i^{(o)}) - cos(\theta_i^{(m)})|$ when the original feature spaces have slight changes. $Trace(U_s U_s^T)$ and $trace(U_t U_t^T)$ are used to smooth f_s and f_t. λ_s is set to $0.01/mr$, and λ_t is set to $0.01/nr$. When $m = n$, λ_s and λ_t are set to 0. From Definition 6.7, it is clear that the maps $f_s(X_s)$ and $f_t(X_t)$ will ensure that all principal angles will change slightly as J_1 approaches 0, even when there is some disturbance of up to δ_0. Thus, based on Theorem 6.2 and Definition 6.7, the GLG method is presented as follows.

GLG method. The GLG method aims to find $U_s \in \mathbb{R}^{r \times m}$, $U_t \in \mathbb{R}^{r \times n}$ to minimize the cost function J_1, as defined in (6.10), while $f_s(X_s) = X_s U_s^T$ and $f_t(X_t) = X_t U_t^T$ are LMMs. GLG is expressed as

$$\min_{U_s, U_t} \; J_1(X_s, X_t; U_s, U_t)$$
$$s.\,t.\;\; U_s > 0 \;\text{ and }\; U_t > 0.$$

$f_s(X_s)$ and $f_t(X_t)$ are the new instances corresponding to X_s and X_t in the homogeneous representations with a dimension of r. Knowledge is then transferred between $f_s(X_s)$ and $f_t(X_t)$ using GFK.

Admittedly, LMMs are somewhat restrictive maps because all elements in the U

must be positive numbers. However, we use LMMs to prevent negative transfers, which can significantly prevent very low prediction accuracy in the target domain. From the perspective of the entire transfer process, an LMM, as a positive map, is the only map that can help construct the homogeneous representations ($V_{He} = V_{Ho}$ and $D_{He} = D_{Ho}$). The GFK method provides the second map, which does not have such rigid restrictions and brings two homogeneous representations closer. Hence, the composite map (LMM+GFK) does not carry rigid restrictions and can therefore handle more complex problems. LMMs ensure correctness, thus avoiding negative transfer, and the GFK method improves the ability to transfer knowledge. The following theorem demonstrates the relationship between GFK and GLG.

Theorem 6.4 (degeneracy of GLG)**.** *Given the source domain* $\mathbf{D_s} = (X_s, Y_s)$ *and the heterogeneous and unlabeled target domain* $\mathbf{D_s} = (X_t)$*, if two domains are homogeneous (m = n), then the GLG method degenerates into the GFK method.*

Since this optimization procedure is related to subspaces spanned by the original instances (X_s and X_t) and the subspaces spanned by the distributed instances (X_s^δ and X_t^δ), determining the best way to efficiently arrive at an optimized solution is a difficult and complex problem. Section 6.4 proposes the optimization algorithm, focusing on the solution to GLG.

6.3.4 Discussion of Definitions and Theorems

Since GLG is built around several definitions and theorems, this subsection explains why one definition leads to another and how one theorem leads to other, as well as discussing the importance of these theoretical demonstrations.

Definition 6.2 gives the heterogeneous unsupervised domain adaptation condition (HeUDA condition). If this condition can be satisfied, the knowledge from a source domain will be correctly transferred to a heterogeneous target domain. Theorem 6.1 shows the kind of map that can satisfy the HeUDA condition given in Definition 6.2. In Theorem 6.1, a new map - monotonic map defined in Definition 6.3 - is involved to prove Theorem 6.1. To find maps such as those presented in Theorem 6.1, an LMM is proposed in Lemma 6.1, and Theorem 6.2 proves that LMMs can map two heterogeneous feature spaces to two homogeneous representations with theoretical guarantee. This leads to our first theoretical contribution: how to theoretically prevent negative transfer in the heterogeneous unsupervised domain adaptation setting.

To find the best LMMs for two heterogeneous feature spaces, principal angles, explained in Definition 6.4, are used to describe the distance between two heterogeneous feature spaces. A new measurement $\mathcal{D}$ is proposed in Definition 6.5 to describe the relationships between the original heterogeneous feature spaces and the homogeneous representations that are mapped from the original heterogeneous feature spaces by LMMs. To maintain the principal angles between two original feature spaces, the cost function J_1 is proposed in Definition 6.7. Theorem 6.3 proves that $\mathcal{D}$ is also a metric, which ensures that minimizing J_1 is meaningful for maintaining the principal angles between two original feature spaces. Theorem 6.3 also indicates that $J_1 = 0$ if source and target domains are homogeneous domains, which leads to Theorem 6.4. Theorem 6.3 and Theorem 6.4 lead to our second theoretical contribution: how to describe and maintain the geometric distance between two heterogeneous feature spaces.

6.3.5 Limitation of GLG

Practically, GLG can be extended to address multi-class classification problem since the procedure for constructing homogeneous representations of two heterogeneous domains does not involve y_{si} (labels in a source domain).

However, using GLG to directly address multi-class classification problems does not provide sufficient theoretical guarantees. LMMs, key mapping functions in GLG, can only guarantee that the probability of label "+1" (denoted by $P1$) of an instance set, such as a subset x_t belonging to X_t, will not change to $1-P1$ after mapping this instance set to its homogeneous representation ($f_t(x_t)$). For example, if $P1(x_t) = 0.6$, then $P1(f_t(x_t))$ only lies in the interval $(0.4, 0.6]$, but, in the multi-class situation (considering 10 classes), if $P1(x_t) = 0.1$, then $P1(f_t(x_t))$ will lie in the interval $[0.1, 0.9)$. The interval $[0.1, 0.9)$ is not accepted because it is too long. If GLG is directly used to address the multi-class classification problem, the accuracy in the target domain will be low. To address this problem, a new mapping function (e.g., $f_t(x_t)$) is needed to ensure that $P1(f_t(x_t))$ is close to $P1(x_t)$, which is difficult to satisfy in unsupervised and heterogeneous situation.

In our future work, we aim to extend GLG to address multi-class classification problems by using label-noise learning methods because an unlabeled target domain with predicted labels can be regarded as a domain with noisy labels.

6.4 Optimization of GLG

According to (6.10), we need to calculate 1) $\partial\sigma_i(C^TD)/\partial U_s$, $\partial\sigma_i(C^TD)/\partial U_t$ and 2) the integration with respect to δ to minimize J_1 via a gradient descent algorithm, where $C = span(f_s(X_s^\delta))$, $D = span(f_t(X_t^\delta))$, $\delta \in (0, \delta_0]$ and $i = 1, ..., r$.

Calculating $\partial\sigma_i(C^TD)/\partial U_s$ and $\partial\sigma_i(C^TD)/\partial U_t$ contains the process of spanning a feature space to become a subspace. Thus, when there are disturbances in an original feature space, the microscopic changes of the eigenvectors in an *Eigen dynamic system* (EDS) need to be analyzed (Eigenvectors are used to construct the subspaces spanned by a feature space, i.e., C and D). The following subsection discusses the microscopic analysis of an EDS.

6.4.1 Microscopic Analysis of an Eigen Dynamic System

In this section, we explore the extent of the changes in subspace $A = span(X)$ when the feature space (X) has suffered a disturbance, expressed as $\partial A/\partial X$. Without loss of generality, assume $A \in G_{N,n}$ (formed as an $\mathbb{R}^{N\times n}$ matrix) and $X \in \mathbb{R}^{N\times n}$, where n is the number of features of X and N is the dimension of the whole space. In keeping with SVD, A is the first n columns of the eigenvectors of XX^T, which means we have the following equations:

$$XX^T y_i = y_i\lambda_i,\ i = 1,...,n$$
$$y_i^T y_i = 1,$$

where y_i is the i^{th} column of A, and λ_i is the eigenvalue corresponding to y_i.

It is clear that if X is disturbed, due to equality, y_i and λ_i will change correspondingly. This equation represents a basic EDS, which is widely used in many fields. To microscopically analyze this equation, we differentiate it into

$$\frac{\partial XX^T}{\partial X}y_i + XX^T\frac{\partial y_i}{\partial X} = y_i\frac{\partial\lambda_i}{\partial X} + \frac{\partial y_i}{\partial X}\lambda_i. \tag{6.11}$$

After a series of calculations, Lemma 6.2 is derived as follows.

Lemma 6.2 (first-order derivatives of EDS)**.** *Given a feature space $X \in \mathbb{R}^{N\times n}$, let $A = span(X) \in G_{N,n}$ (formed as an $\mathbb{R}^{N\times n}$ matrix), let y_i be the i^{th} column of A, and let λ_i be the eigenvalue corresponding to y_i. The first-order derivatives of the EDS are*

$$\frac{\partial y_i}{\partial X} = -(XX^T - \lambda_i I)^+ \frac{\partial XX^T}{\partial X} y_i,$$
$$\frac{\partial \lambda_i}{\partial X} = y_i^T \frac{\partial XX^T}{\partial X} y_i,$$

where $(XX^T - \lambda_i I)^+$ is the Moore-Penrose pseudoinverse of $XX^T - \lambda_i I$.

Based on Lemma 6.2, we know the extent of the changes in subspace $A = span(X)$ when the feature space (X) has suffered a disturbance, expressed as $\partial A/\partial X$.

6.4.2 Gradients of Cost Function I

With the proposed lemma, we obtain the derivative of cost function J_1 using following chain rules. For simplicity, S_m^s is short for $S_m(f_s, X_s^\delta)$ and S_m^t is short for $S_m(f_t, X_t^\delta)$.

(6.12)
$$\frac{\partial J_1}{\partial (U_s)_{cd}} = \int_{\delta=0}^{\delta_0} \frac{\partial J_1}{\partial \mathcal{D}} \sum_{i=1}^{r} \frac{\partial \mathcal{D}}{(\partial \sigma_i((S_m^s)^T S_m^t)} \cdot Tr\left(\left(\frac{\partial \sigma_i((S_m^s)^T S_m^t)}{\partial S_m^s}\right)^T \frac{\partial S_m^s}{\partial (U_s)_{cd}}\right) d\delta + \lambda_s (U_s)_{cd}.$$

The first and second terms of the right side can be easily calculated according to the definition of the cost function J_1. Using chain rules, the third term can be calculated by the following equations:

(6.13)
$$\frac{\partial \sigma_i((S_m^s)^T S_m^t)}{\partial (S_m^S)_{kl}} = Tr\left(\left(\frac{\partial \sigma_i((S_m^s)^T S_m^t)}{\partial (S_m^s)^T S_m^s}\right)^T \frac{\partial (S_m^s)^T S_m^s}{\partial (S_m^S)_{kl}}\right),$$

$$\frac{\partial(S_m^S)_{kl})}{\partial(U_s)_{cd}} = Tr\left(\left(\frac{\partial(S_m^s)_{kl}}{\partial f_s(X_s^\delta)}\right)^T \frac{\partial f_s(X_s^\delta)}{\partial(U_s)_{cd}}\right). \tag{6.14}$$

In terms of the first-order derivatives of EDS, we have following equations:

(6.15)

$$\left(\frac{\partial\sigma_i((S_m^s)^T S_m^t)}{\partial(S_m^s)^T S_m^s}\right)_{pq} = \frac{1}{2\sigma_i((S_m^s)^T S_m^t)} y_i^T \Big(J_{pq}((S_m^s)^T S_m^t)^T + ((S_m^s)^T S_m^t) J_{pq}^T\Big) y_i,$$

$$\left(\frac{\partial(S_m^s)_{kl}}{\partial f_s(X_s^\delta)}\right) = -\Big((f_s(f_s)^T - \lambda_l I)^+ (J_{ab}(f_s)^T + f_s J_{ab}^T)\cdot(S_m^s)_{*l}\Big)_k, \tag{6.16}$$

where y_i is the eigenvector corresponding to $\sigma_i((S_m^s)^T S_m^t)$, λ_l is the l^{th} eigenvalue corresponding to l^{th} column of S_m^s, and J_{pq} is a single-entry matrix with 1 at $(p;q)$ and zero elsewhere. (6.15) will generate a matrix of the same size as $(S_m^s)^T S_m^t$, and (6.16) will generate a matrix of the same size as $f_s(X_s^\delta)$, i.e., f_s in (6.16). For other terms of (6.13) and (6.14), we have the following equations:

$$\frac{\partial(S_m^s)^T S_m^s}{\partial(S_m^S)_{kl}} = J_{kl}^T S_m^t + (S_m^s)^T J_{kl}, \tag{6.17}$$

$$\frac{\partial f_s(X_s^\delta)}{\partial(U_s)_{cd}} = X_s^\delta J_{cd}. \tag{6.18}$$

We adopt Simpson's rule to integrate δ. Simpson's rule is a method of numerical integration that can be used to calculate the value of cost function J1. We set $\Delta = \delta_0/10$, and the derivative of cost function J_1 is calculated with

$$\frac{\partial J_1}{\partial(U_s)_{cd}} = \frac{\Delta}{6}\sum_{I=0}^{9}\Big(g_s(I\Delta) + 4g_s\big(I\Delta + \frac{\Delta}{2}\big) + g_s(I\Delta+\Delta)\Big) + \lambda_s(U_s)_{cd}, \tag{6.19}$$

where $g_s(\Delta)$ is the integrated part in (6.12) with $S_m^s = S_m(f_s, X_s^\Delta)$. The gradient descent equations for minimizing the cost function J_1 with respect to $(U_s)_{cd}$ are

$$(U_s)_{cd} = (U_s)_{cd} - v_{bool}^s \times \eta \frac{\partial J_1}{\partial (U_s)_{cd}}, \tag{6.20}$$

where

$$v_{bool}^s = \max\left\{0, (U_s)_{cd} - \eta \frac{\partial J_1}{\partial (U_s)_{cd}}\right\}, \tag{6.21}$$

v_{bool}^s, expressed in (6.21), is used to keep f_s as an LMM. Similarly, we optimize U_t using the following equation.

$$(U_t)_{ce} = (U_t)_{ce} - v_{bool}^t \times \eta \frac{\partial J_1}{\partial (U_t)_{ce}}. \tag{6.22}$$

6.4.3 A Hybrid Optimization Method for GLG

We use a hybrid method of minimizing J_1: 1) an evolutionary algorithm: cuckoo search algorithm (CSA) [Yang and Deb, 2010], is used to find initial solutions $U_s^{(0)}$ and $U_t^{(0)}$; 2) a gradient descent algorithm to find the best solutions. To accelerate the speed of the gradient descent algorithm, we select η from [0.01, 0.05 0.1, 0.2, 0.5, 1, 5, 20] such that it obtains the best (minimum) cost value for each iteration.

For CSA, we set the number of nests as 30, the discovery rate as 0.25, the lowest bound as 0, the highest bound as 1 and the number of iteration as 100. We also apply Simpson's rule to estimate the integration value in J_1. CSA has been widely applied in many fields. Its code can be downloaded from MathWorks.com where readers can also find more detailed information about this algorithm. Algorithm 6.1 presents the pseudo code of the GLG method. $MaxIter$ is set to 100, err is set to 10^{-5} and δ_0 of J_1 is set to 0.01. Ultimately, $\mathbf{D}_s^{Adp} = (X_s^{Adp}, Y_s)$

Algorithm 6.1 Pseudo code of GLG method

1: Input: Source data, target data: X_s, X_t.
2: Initialize $U_s^{(0)}, U_t^{(0)} \leftarrow \text{CSA}(X_s, X_t)$; // Get initial solutions
for $i = 0 : MaxIter$ **do**
  3: Compute $Error_o \leftarrow J_1(X_s, X_t; U_s^{(i)}, U_t^{(i)})$; // Select the best η;
  4: Update $U_s^{(i+1)} \leftarrow U_s^{(i)}$ using (6.20);
  5: Update $U_t^{(i+1)} \leftarrow U_t^{(i)}$ using (6.22);
  6: Compute $Error_n \leftarrow J_1(X_s, X_t; U_s^{(i+1)}, U_t^{(i+1)})$;
  if $|Error_o - Error_n| < err$ **then**
    7: Break; % Terminates the iteration.
  end
end
8: Compute $X_s^{Ho} \leftarrow X_s U_s^T$;
9: Compute $X_t^{Ho} \leftarrow X_t U_t^T$;
10: Compute $[X_s^{Adp}, X_t^{Adp}] \leftarrow \text{GFK}(X_s^{Ho}, X_t^{Ho})$;
11: Output: Source data, target data: X_s^{Adp}, X_t^{Adp}.

can be used to train a classifier, based on X_s^{Adp} and X_t^{Adp}, to predict the labels for $\mathbf{D}_t^{Adp} = (X_t^{Adp})$.

6.5 Experiments

To validate the overall effectiveness of the GLG method, we conducted experiments with five datasets across three fields of application: cancer detection, credit assessment, and text classification. All datasets are publicly available from the UCI Machine Learning Repository (UMLR) and Transfer Learning Resources (TLR). An SVM algorithm was used as the classification engine.

6.5.1 Datasets for HeUDA

The five datasets were reorganized since no real-world datasets directly related to HeUDA. Table 6.1 lists the details of the datasets from UMLR and TLR. Reuters-

21578 is a transfer learning dataset, but we needed to merge the source domain for each category with its corresponding target domain into a new domain, e.g., OrgsPeople_src and OrgPeople_tar were merged into OrgPeople; and similarly for OrgPlaces and PeoplePlaces. Table 6.2 lists the tasks and clarifies the source and target domains. Tasks G2A, Ope2Opl and CO2CD are described in detail below. Other tasks have similar meanings.

1) G2A: Assume that the German data is labeled and the Australian data is unlabeled. Label “1" means “good credit” and label “-1" means “bad credit”. This task is equivalent to the question: “Can we use knowledge from German credit records to label unlabeled Australian data?"

2) Ope2Opl: Assume that in one dataset “Org” is labeled “1" and “People” is labeled “-1" (Ope in Table 6.2). Another unlabeled dataset may contain “Org” labeled as “1". This task is equivalent to the question: “Can we use the knowledge from Ope to label “Org" in the unlabeled dataset?"

3) CO2CD: Assume that in the Breast Cancer Wisconsin (Original) dataset (CO in Table 6.2) “1" represents “malignant” and “-1" represents “benign”. Another unlabeled dataset related to breast cancer also exists. This task is equivalent to the question: “Can we use the knowledge from CO to label “malignant" in the unlabeled dataset?"

Recall $\mathcal{X}$ and ω, in datasets of Breast Cancer and credit assessment, $\mathcal{X}$ is human beings and ω is a subset of $\mathcal{X}$, which means that ω is a set containing many persons. For each person in ω, we may diagnose whether he/she has Breast Cancer using features that CO or CD datasets adopt. Similarly, for each person in ω, we may assess his/her credit using standards in Germany or Australia.

The multivariate random variable $\mathbf{X}$ in datasets of Breast Cancer describes

Table 6.1: Description of the original datasets.

Field	Dataset name	# of instances	# of features	Source
Credit	German Credit Data	1000	24	UMLR
	Australian Credit Approval	690	14	UMLR
Text	Reuters-21578 OrgsPeople_src	1237	4771	TLR
	Reuters-21578 OrgsPeople_tar	1208	4771	TLR
	Reuters-21578 OrgsPlaces_src	1016	4415	TLR
	Reuters-21578 OrgsPlaces_tar	1043	4415	TLR
	Reuters-21578 PeoplePlaces_src	1077	4562	TLR
	Reuters-21578 PeoplePlaces_tar	1077	4562	TLR
Cancer	Breast Cancer Wisconsin (Original)	683	9	UMLR
	Breast Cancer Wisconsin (Diagnostic)	569	30	UMLR

key features to distinguish whether a tumour is benign. Namely, if we can obtain observations from distribution of $\mathbf{X}$, we will perfectly classify benign tumour and malignant tumour. However, these observations cannot be obtained and we can only obtain features in both datasets of Breast Cancer used in this chapter. Features in both datasets can be regarded as observations from $\mathbf{X_s}$ and $\mathbf{X_t}$ which are heterogeneous projections of $\mathbf{X}$.

In datasets of credit assessment, $\mathbf{X}$ describes key features to distinguish whether the credit of a person is good. However, observations from distribution of $\mathbf{X}$ cannot be obtained and we can only obtain features in German and Australian datasets. Features in both datasets can be regarded as observations from $\mathbf{X_s}$ and $\mathbf{X_t}$ which are heterogeneous projections of $\mathbf{X}$.

Table 6.2: Transfer tasks (10 tasks in total).

Field	**Source**	**Target**	**Labels**	**Task**
Credit	German Credit Data	Australian Credit Approval	1: Good	G2A
	Australian Credit Approval	German Credit Data	1: Good	A2G
Text	OrgsPeople	OrgsPlaces	1: Orgs	Ope2Opl
	OrgsPlaces	OrgsPeople	1: Orgs	Opl2Ope
	OrgsPlaces	PeoplePlaces	-1: Places	Opl2Ppl
	PeoplePlaces	OrgsPlaces	-1: Places	Ppl2Opl
	PeoplePlaces	OrgsPeople	-	Ppl2Ope
	OrgsPeople	PeoplePlaces	-	Ope2Ppl
Cancer	Breast Cancer Wisconsin (Original)	Breast Cancer Wisconsin (Diagnostic)	1: Malignant	CO2CD
	Breast Cancer Wisconsin (Diagnostic)	Breast Cancer Wisconsin (Original)	1: Malignant	CD2CO

6.5.2 Experimental Setup

The baselines and their implementation details are described in the following section.

6.5.2.1 Baselines

It was important to consider which baselines to compare the GLG method with. There are two baselines that naturally consider situations where no related knowledge exists in an unlabeled target domain: 1) methods that label all instances as "1", denoted as A1; and 2) methods that cluster the instances with random category labels (the k-means method clusters the instances in the target domain into two categories), denoted as CM. It is important to highlight that A1 and CM are non-transfer methods.

When transferring knowledge from a source domain to a heterogeneous and unlabeled target domain, there is a simple baseline that applies dimensional reduction technology to force the two domains to have the same number of features. Denoted as *Dimensional reduction Geodesic flow kernel* (DG), this method forces the dimensionality of all features to be the same. DG is a useful method to show the difficulties associated with HeUDA problem.

An alternative method, denoted as *Random Maps GFK* (RMG), randomly maps (linear map) features of two domains onto the same dimensional space. The comparison between this method and *Random LMM GFK* (RLG) shows the effect of negative transfer. The RLG method only uses random LMMs to construct the homogeneous representations and does not preserve the distance between the domains (it only considers the variation factor). The KCCA method

with randomly-paired instances is also considered as a baseline.

Although deep-learning based methods were originally designed for homogeneous domains, we only need to change the number of neurons in the second layer of these methods to make them suitable for heterogeneous domains. Consequently, DANN [Ganin et al., 2016a], DAN [Long et al., 2019] and *beyond-sharing-weights domain adaptation* (BSWDA) [Rodriguez and Laio, 2014] are selected to compare with GLG.

The last selected baseline is SFER, which is inspired by the fuzzy co-clustering method (cluster features of two domains). Apart from the deep-learning based methods, the selected domain adaptation methods and GLG methods map two heterogeneous feature spaces onto the same dimensional feature space (i.e., the homogeneous representations) at the lowest dimension of the original feature spaces.

6.5.2.2 Implementation details

Following [Li et al., 2014; Pan et al., 2011; Pan and Yang, 2010; Xiao and Guo, 2015], SVM was trained on homogeneous representations of source domain, then tested on target domain. The following section provides implementation details of our experiments.

The original datasets used in the text classification tasks were preprocessed using SVD (selecting top 50% Eigenvalues) as the dimensionality reduction method for non-deep methods. We randomly selected $1,500$ unbiased instances from each domain to test the proposed method and baselines. The German Credit dataset contains some bias, with 70% of the dataset labeled 1 and 30% labeled -1; however, the Australian Credit Approval dataset is unbiased. Given the basic

assumption that both domains are similar, we needed to offset this dissimilarity by changing the implementation of the experiments with this dataset. Hence, we randomly selected 600 unbiased instances from the German Credit dataset for every experiment and ran the experiment 50 times for each method and each task.

The DAN and BSWDA methods are neural networks including five layers: input layer, hidden layer I, hidden layer II, representation layer and output layer. For the credit and cancer datasets, the number of neurons in hidden layer I (and II) is 200 and the number of neurons in the representation layer is 100. For text classification dataset, the number of neurons in hidden layer I (and II) is $2,000$ and the number of neurons in the representation layer is $1,000$. The classifier for the DANN method is also a neural network (including five layers) and has the same setting with DAN and BSWDA. Its domain classifier is a three-layer neural network. The number of neurons in the hidden layer of DANN's domain classifier is set to $1,000$ for text classification dataset and 100 for other datasets. Following [Saito et al., 2017], Adagrad optimizer is used to optimize parameters of DAN, BSWDA and DANN on text classification datasets, since Adagrad optimizer is suitable for sparse features. On other datasets, Adam optimizer is adopted to optimize parameters of DAN, BSWDA and DANN.

$Accuracy$ was used as the test metric, as it has been widely adopted in the literature [Ghifary et al., 2017; Li et al., 2014; Pan et al., 2011]:

$$Accuracy = \frac{|x \in X_t : g(x) = y(x)|}{|x \in X_t|},$$

where $y(x)$ is the ground truth label of x, while $g(x)$ is the label predicted by the SVM classification algorithm. Since the target domains do not contain any

labeled data, it was impossible to automatically tune the optimal parameters for the target classifier using cross-validation. As a result, we used LIBSVM's default parameters for all classification tasks. Because there were no existing pairs in the 10 tasks, we randomly matched instances from each domain as pairs for the KCCA method. For neural networks, we report the best average accuracy of each dataset (using the same parameters for tasks constructed from the same dataset) by tuning the learning rate of optimizers and the penalty parameters of the regularizers of each method. The batch size was set to 24 and the number of epochs was set to 50 for all datasets.

All experiments were conducted on an Intel(R) Core(TM) i7-4770 CPU at 3.40Ghz with a memory of 64 GB running Windows 7 professional 64-bit operating system. Deep-learning based methods were implemented by Pytorch 0.4.0 and other methods were implemented by Matlab 9.2.0. To show the complexity of each task, we also tested same-domain accuracy with a 5-fold SVM using the default parameters on seven different target domains. We randomly selected unbiased instances from five domains (the cancer datasets were excluded), and ran the experiments 50 times, preprocessing the instances with the zscore function. Table 6.3 shows the average accuracy and standard deviations in terms of AVG±STD. The results show that the German Credit dataset and the Ppl dataset were the hardest to classify and the Cancer-D and Cancer-O datasets were the easiest. In general, the accuracy of the HeUDA methods was lower than the same-domain (target) accuracy due to the lack of labels in the target domain.

Table 6.3: Same-domain accuracy of each target domain using 5-fold SVM.

German	**Australia**	**Opl**	**Ope**	**Ppl**	**CD**	**CO**
71.21% ±1.56%	86.10% ±0.82%	84.97% ±0.88%	85.15% ±0.71%	78.40% ±0.82%	97.01% ±0.00%	96.49% ±0.00%

6.5.3 Experiment I: RMG

This experiment demonstrates a situation in which the transfer process is unreliable. It is a natural idea to propose a HeUDA method that randomly maps two domains onto the same feature space, then uses a HoUDA method to adapt the domains. Hence, the RMG method randomly generated $f_s(X_s) = X_s U_s^T$ and $f_t(X_t) = X_t U_t^T$ to transfer knowledge from the source domain to the target domain. Table 6.4 shows the classification results for RMG compared to CM across 50 tests against three criteria: AVG±STD, max accuracy, and min accuracy. The results indicate that RMG is not a valid option for transferring knowledge from a source domain to a target domain. The average accuracy was low, especially for the CD2CO task, where the minimum accuracy was 7.91%. Namely, label space was greatly changed after the transfer.

Table 6.4: The classification results for RMG and CM.

Field	Task	Average Accuracy		Max Accuracy		Min Accuracy	
		RMG	CM	RMG	CM	RMG	CM
Credit	G2A	**49.46%±13.31%**	44.89%±0.40%	**75.94%**	56.23%	**24.49%**	43.77%
	A2G	**49.34%±5.2%**	50.97%±5.21%	**59.33%**	57.17%	**36.00%**	43.67%
Text	OPe2OPl	**52.36%±5.20%**	49.76%±5.79%	**62.47%**	59.93%	**41.27%**	40.07%
	OPl2OPe	46.14%±5.10%	**49.4%±5.01%**	56.00%	**56.87%**	37.33%	**43.13%**
	OPl2PPl	48.98%±5.84%	**50.70%±5.24%**	**62.67%**	58.40%	36.13%	**41.47%**
	PPl2OPl	49.34%±5.58%	**49.76%±5.79%**	**64.27%**	59.93%	38.40%	**40.07%**
	OPe2PPl	**51.83%±4.99%**	50.70%±5.24%	**60.80%**	58.40%	40.07%	**41.47%**
	PPl2OPe	49.35%±4.88%	**49.4%±5.01%**	**61.40%**	56.87%	**40.47%**	43.13%
Cancer	CD2CO	**58.92%±27.88%**	38.94%±45.17%	**96.49%**	96.19%	**7.91%**	3.81%
	CO2CD	**49.18%±20.87%**	37.25%±33.37%	**89.10%**	85.41%	**14.41%**	14.59%

Table 6.5: The results of the MMD test for the mapped and adapted domains in two extreme situations (lowest and highest accuracy) of task CD2CO among 50-time experiments.

Situation	**Task/Accuracy**	**Homogeneous representations**	**Adapted domains**
Lowest Acc.	CD2CO/7.91%	No	Yes
Highest Acc.	CD2CO/96.49%	No	Yes

The results of the two-sample MMD tests [Gretton et al., 2012] are shown in Table 6.5 to demonstrate the significance of Theorem 3.1. These tests measure the maximum and minimum accuracy of the homogeneous representations for the two CD2CO tasks. In Table 6.5, “No" means that the two domains have different distributions, while “Yes" means the two domains have the same distribution.

It is easy to see that distributions of feature spaces of adapted domains can be regarded as having the same distribution (in terms of MMD) in these two extreme situations (highest and lowest accuracy). However, these identically-distributed domains unexpectedly returned extremely different accuracies at 7.91% and 96.49% when using SVM to label the instances in the target domain. This will result in significant errors even if $P(f_s(X_s)) = P(f_t(X_t))$, which is clearly caused by $P(Y|f_s(X_s)) \neq P(Y|f_t(X_t))$ (significant difference). Thus, this experiment supports our claim that Definition 6.2 (the HeUDA condition) and Theorem 6.1 (the unsupervised knowledge transfer theorem) are both necessary. It also shows the consequences of ignoring Theorem 6.1 - the conditional probability distribution will significantly change.

6.5.4 Experiment II: Overall comparisons

This section presents classification results of the methods presented in Section 6.5.2.1, which are shown in Table 6.6. The results reflect that the GLG method was able to complete these 10 tasks effectively, and with better accuracy than other baselines. Our overall analysis of the comparative results reveals the following insights:

1) The GLG method produced more stable classification results and higher classification accuracy than the other methods.

2) Although the KCCA, DAN, BSWDA and DANN methods outperformed A1, DG, CM, and RMG in some tasks, the classification results were unstable as they did not prevent extreme negative transfer.

3) Since deep-learning based methods (DAN, BSWDA and DANN) do not prevent extreme negative transfer, their classification results are unstable in tasks G2A, CD2CO and CO2CD. This means that a neural network, as a mapping function, cannot be directly used to address the HeUDA problem.

4) Although deep-learning methods have considerable potential for fining a representation of two domains, they cannot be directly used for addressing HeUDA problem. Some constraints should be considered to make a neural network prevent extreme negative transfer.

5) The DANN method produced more stable classification results than DAN and BSWDA, which indicates that adversarial learning method is more suitable for the HeUDA problem than the two-sample-test-based method.

6) GLG performs better than SFER in terms of average accuracy over 10 tasks, which means that principal angles are better than fuzzy equivalence

relations for describing relationships between two heterogeneous feature spaces.

7) GLG can outperform baselines on 9 out of 10 tasks. On these 9 tasks, using Friedman test, the improvement in performance of GLG over all baselines is statistically significant on 7 tasks (p-value is less than 0.05). The remaining 2 tasks are OPl2OPe and CO2CD. On both of tasks, GLG cannot statistically significantly outperform SFER (the strongest baseline).

8) Although SFER outperforms GLG on task OPe2OPl, we use Friedman test to investigate that SFER cannot statistically significantly outperform GLG on this task (p-value is greater than 0.05). In summary, GLG is significantly better than all baselines on 7 out of 10 tasks and has the same performance with SFER on the remaining 3 tasks from statistical view.

9) In comparing same-domain accuracy, the CD2CO task outperformed the CO task. Same-domain accuracy was harder to achieve in the text classification tasks than in the other two tasks. This result indicates that text classification tasks lose more information when transferring knowledge from the source domain to the target domain.

For the runtime of each method, A1, DG, CM, RMG, KCCA and RLG finished the G2A task within 10 *seconds* (s). DAN took 142.35s, BSWDA took 167.88s, DANN took 121.34s, and SFER took 20.38s, and GLG took 82.05s. When running GLG, the CSA algorithm costs 44.82s, and Algorithm 6.1 costs 35.34s, and other procedures cost 1.89s.

Table 6.6: The classification results (AVG±STD) for GLG and benchmark models. Bold values represent the lowest average accuracy in each task.

Field	Credit		Text						Cancer	
Tasks	G2A	A2G	OPe2OPl	OPl2OPe	OPl2PPl	PPl2OPl	OPe2PPl	PPl2OPe	CD2CO	CO2CD
A1	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%	65.01%	62.74%
DG	45.19% ±1.96%	50.92% ±1.06%	47.17% ±3.14%	44.47% ±1.60%	48.38% ±5.51%	46.37% ±4.09%	45.52% ±2.54%	43.07% ±1.75%	34.62% ±17.25%	35.87% ±7.56%
CM	44.89% ±0.4%	50.97% ±5.21%	49.76% ±5.79%	49.40% ±5.01%	50.70% ±5.24%	49.76% ±5.79%	50.70% ±5.24%	49.40% ±5.01%	38.94% ±45.17%	37.25% ±33.37%
RMG	49.46% ±13.31%	49.34% ±5.2%	52.36% ±5.2%	46.14% ±5.1%	48.98% ±5.84%	49.34% ±5.58%	51.83% ±4.99%	49.35% ±4.88%	58.92% ±27.88%	49.18% ±20.87%
KCCA	51.05% ±9.72%	50.52% ±4.64%	49.28% ±3.22%	48.19% ±3.66%	50.27% ±3.21%	50.03% ±3.64%	50.52% ±3.52%	43.07% ±1.75%	75.50% ±15.19%	57.10% ±6.3%
RLG	72.70% ±6.14%	57.21% ±3.96%	60.15% ±3.16%	58.08% ±2.65%	57.71% ±2.5%	63.07% ±3.11%	56.88% ±2.55%	56.89% ±2.9%	96.59% ±0.45%	90.19% ±0.71%
DAN	55.28% ±6.25%	52.45% ±3.46%	52.93% ±1.73%	52.47% ±1.72%	53.67% ±2.03%	54.31% ±1.56%	51.41% ±2.26%	49.77% ±2.29%	58.35% ±26.11%	84.53% ±5.33%
BSWDA	51.08% ±8.04%	52.67% ±6.40%	51.57% ±3.98%	51.33% ±3.76%	51.08% ±3.88%	51.13% ±2.75%	51.59% ±2.49%	50.55% ±2.79%	67.18% ±26.13%	83.03% ±16.18%
DANN	65.72% ±7.43%	56.78% ±2.32%	54.25% ±1.56%	52.59% 1.27%	52.22% ±1.75%	54.56% ±1.74%	52.59% ±2.11%	51.37% ±1.69%	94.55% ±2.31%	87.89% ±1.44%
SFER	75.77% ±0.95%	60.50% ±1.35%	**62.19% ±1.44%**	58.95% ±1.52%	56.91% ±1.28%	64.11% ±0.81%	56.01% ±0.88%	57.51% ±1.15%	96.61% ±0.01%	90.20% 0.03%
GLG	**78.18% ±1.53%**	**61.25% ±2.1%**	62.10% ±1.63%	**59.54% ±0.85%**	**59.62% ±1.54%**	**65.57% ±0.79%**	**58.31% ±0.83%**	**58.81% ±1.38%**	**97.18% ±0.15%**	**90.22% ±0.26%**

6.6 Summary

This chapter fills two theoretical gaps in the field of heterogeneous unsupervised domain adaptation. On a fundamental level, this chapter presents an unsupervised knowledge transfer theorem that outlines the sufficient conditions to guarantee that knowledge is transferred correctly from a source domain to a heterogeneous and unlabeled target domain. Additionally, this chapter proves that the theorem is able to avoid negative transfer with at least one type of mapping function - LMM in this case. The theorem incorporates a distance metric, based on principal angles, to help construct homogeneous representations for heterogeneous domains.

The theorem, the distance metric, and the LMM mapping function are presented within the GLG method, which optimizes (minimizes) the principal angle-based metric to construct homogeneous representations for heterogeneous domains, then transfers knowledge across the homogeneous representations using a geodesic flow kernel. The overall efficacy of the GLG method was tested with five public datasets on three practical tasks: cancer detection, credit assessment, and text classification. The method demonstrates superior performance over the existing baselines in all evaluation criteria.

CHAPTER 7

SHARED FUZZY EQUIVALENCE RELATIONS: A HETEROGENEOUS UNSUPERVISED DOMAIN ADAPTATION APPROACH FOR IMBALANCED DOMAINS

7.1 Introduction

To propose an effective HeUDA method, we successfully designed GLG in Chapter 6, which solves the two aforementioned bottlenecks and has satisfactory classification results on three real applications. However, no matter which HeUDA method (KCCA [Yeh et al., 2014] or GLG) is used, it does not work well when the target domain is a small dataset due to the limitations of CCA and the Grassmann manifold (both of which need two domains with the same number of samples). This will limit the amount of knowledge from the source domain which

is transferred to the target domain. For example, if we have a source domain containing 10,000 samples, we need to label samples of the target domain which only have 50 samples. If we use the KCCA or GLG method, we can only select 50 samples in the source domain and transfer the knowledge from these 50 selected samples to the target domain. This means we waste 99.5% of the information of the source domain. Thus, to leverage more information in the source domain to help train a better classifier for the target domain, this chapter applies the n-dimensional fuzzy geometry and the fuzzy equivalence relations to transfer all of the knowledge from the source domain to the heterogeneous and unlabeled target domain, where two domains have a different number of samples.

Fuzzy technology plays a key role in domain adaptation. Common fuzzy technology includes fuzzy rules, fuzzy relations and Takagi - Sugeno models. Fuzzy rules and Takagi - Sugeno models are frequently used to transfer knowledge across two domains [Behbood et al., 2015, 2014; Zuo et al., 2019, 2017a,b] since they can extract general fuzzy representations of two domains and illustrate how knowledge is transferred across domains. Deng et al. proposed a series of novel methods via the Mamdani-Larsen-type fuzzy system and Takagi-Sugeno-Kang-type fuzzy system [Deng et al., 2014, 2013a, 2016a, 2013b, 2016b]. Liu et al. used fuzzy relations and fuzzy fusion to propose a series of novel methods to address the UDA problem in heterogeneous scenarios [Liu et al., 2018b, 2017, 2018c]. Compared to integral-probability-metric, adversarial-training and pseudo-label based methods, fuzzy methods can bridge features of two heterogeneous domains via fuzzy relations [Liu et al., 2018b], which is the main reason why the fuzzy relation based method can address the UDA problem in heterogeneous scenarios.

Based on these advantages of fuzzy technology, we first propose a new metric

$\mathcal{D}$ on an n-dimensional fuzzy space $\mathcal{F}(\mathbb{R}^n)$ where each feature of a domain is regarded as a fuzzy vector. This new metric contains fuzzy degrees of fuzzy vectors and it is proved that $(\mathcal{D}, \mathcal{F}(\mathbb{R}^n))$ is a metric space. Then, we use this metric to measure the similarity of two fuzzy vectors and build the fuzzy equivalence relations matrix of each domain.

In a traditional fuzzy equivalence relations matrix, we can use an α to cluster these fuzzy vectors (representing features) into several categories and these categories are regarded as more general fuzzy representations. However, traditional fuzzy equivalence relations cannot guarantee that fuzzy equivalence relations matrixes of two domains can have the same number of clustering categories with the same α. Motivated by this, we propose the *shared fuzzy equivalence relations* (SFER), which allows two fuzzy equivalence matrixes of two domains to share the same α. Compared to the traditional fuzzy equivalence relations, the SFER can guarantee that fuzzy equivalence relations matrixes of two domains can have the same number of clustering categories with the same α. Eventually, with the help of the SFER, we can transfer knowledge from the source domain to the target domain via these clustering categories.

The main contributions of this chapter can be summarized as follows.

1) A novel F-HeUDA method is proposed to address the HeUDA problem when two domains have different numbers of samples, via adopting n-dimensional fuzzy geometry and fuzzy equivalence relations to the HeUDA problem. Both fuzzy technologies successfully overcome the drawbacks of CCA and the Granssmann manifold (two domains must have the same number of samples). As a result, the proposed method can transfer more knowledge from a source domain to a target domain than KCCA and GLG methods when there are very few

samples in the target domain.

2) Two important properties of fuzzy equivalence relations are discovered and proved in this chapter, which are the theoretical guarantees of the SFER method and key parts of the proposed method. Based on both properties, it can be guaranteed that fuzzy equivalence relations matrixes of two domains can have the same number of clustering categories with the same and proper α.

3) It is the first time that n-dimensional fuzzy geometry and fuzzy equivalence relations have been applied to address the domain adaptation problem and the proposed method performs well in real applications.

4) The F-HeUDA method provides a user-oriented decision making pattern for decision makers. In the proposed method, we propose a default method by which to automatically select α for different tasks which works well in four real applications. Furthermore, decision makers can still easily select the value of α based on their own experience and requirements. This extends the application scenario of the proposed method.

7.2 Concepts and Notations

This section introduces the two concepts used in this chapter, which are *n-dimensional fuzzy geometry* (*n-D* FG), fuzzy equivalence relations.

7.2.1 Fuzzy Geometry

The geometric properties of fuzzy sets have been researched from various aspects such as fuzzy point, fuzzy line and fuzzy circle [Buckley and Eslami, 1997a,b; Chakraborty and Ghosh, 2014; Ghosh and Chakraborty, 2012, 2016]. The *n-D*

FG theory provides an effective way to analyze and compute fuzzy information in a geometric form [Li et al., 2015]. In this subsection, the definition of fuzzy vector is introduced, which is the core element in an *n-D* FG. Without loss of generality, this chapter uses capital or small letters with a bar to represent the fuzzy subsets or fuzzy points of $\mathbb{R}^n$. The membership function of a fuzzy set $\bar{A}$ is denoted by $\mu(x|\bar{A})$, $x \in R^n$, with $\mu(x|\bar{A}) \subseteq [0,1]$. The definition of fuzzy vector at an *n-D* real valued vector A is presented as follows.

Definition 7.1 (Fuzzy number [Goetschel and Voxman, 1983])**.** *A fuzzy set $\bar{S}$ of $\mathbb{R}$ is called a fuzzy real number (fuzzy number in following) if its membership function μ satisfies the following properties.*

1. $\mu(x|\bar{S}) = 1$ is upper semi-continuous in x.

2. $\mu(x|\bar{S}) = 1$ for x outside some interval $[c,d]$.

3. For some real numbers with $c \le a \le b \le d$, $\mu(x|\bar{S})$ is monotonically increasing in $[c,a]$, monotonically decreasing in $[b,d]$, and $\mu(x|\bar{S}) = 1$ for $x \in [a,b]$.

Then, we can give the definition of the fuzzy vector at an *n-D* vector as follows.

Definition 7.2 (Fuzzy vector [Goetschel and Voxman, 1983])**.** *A fuzzy set $\bar{A}(a_1,a_2,...,a_n)$ of $\mathbb{R}^n$ is called a fuzzy vector at $A = (a_1,a_2,...,a_n) \in \mathbb{R}^n$ if its membership function μ satisfies the following properties.*

1. *$\mu((x_1,x_2,...,x_n)|\bar{A}(a_1,a_2,...,a_n)) = 1$ is upper semi-continuous in $x = (x_1,x_2,...,x_n) \in \mathbb{R}^n$.*

2. *$\mu((x_1,x_2,...,x_n)|\bar{A}(a_1,a_2,...,a_n)) = 1$ if and only if $(x_1,x_2,...,x_n) = (a_1,a_2,...,a_n)$.*

3. *$\bar{A}(\alpha) = \{x|\mu(x|\bar{A}(a_1,a_2,...,a_n)) = \alpha,\ x \in \mathbb{R}^n\}$ is a compact convex subset of $\mathbb{R}^n$ for all α in $[0,1]$.*

The fuzzy vector is the basic element for researching properties of the n-D FG space and the set of all n-D fuzzy vectors is denoted by $F(\mathbb{R}^n)$. The third property of n-D fuzzy vectors implies that $F(\mathbb{R}^n)$ can be connected with $\mathbb{R}^n$ using the membership α. this chapter uses the triangular membership function to construct the membership function of each n-D fuzzy vector in $F(\mathbb{R}^n)$. The detailed form is introduced in next subsection. '

7.2.2 Fuzzy Equivalence Relation

Fuzzy equivalence relations are first mentioned in [Zadeh, 1971] (Zadeh referred to fuzzy equivalence relations as similarity relations in [Zadeh, 1971]) and were studied as a way to measure the similarity among fuzzy sets. Based on the fuzzy equivalence relations, fuzzy equivalence classes can be obtained, which provides a powerful way to analyze the fuzzy partitions. Then, to construct the fuzzy equivalence relations for general fuzzy sets, the max–min operator is proposed to construct max–min transitive closure which is a fuzzy equivalence relations [Mirzaei and Rahmati, 2010]. This section briefly reviews how to generate a fuzzy equivalence relations for fuzzy sets and how to use the fuzzy equivalence relations to partition fuzzy sets.

First, the definition of a fuzzy relation is given as follows.

Definition 7.3 (Fuzzy relations [Zadeh, 1971])**.** *Given N fuzzy sets $\bar{A}_1, \bar{A}_2, ..., \bar{A}_N$, an operator $R : (\bar{A}_i, \bar{A}_j) \rightarrow [0,1]$ is a fuzzy relation on $\bar{A}_1, \bar{A}_2, ..., \bar{A}_N$ if the following properties are satisfied.*

1. $R(\bar{A}_i, \bar{A}_i) = 1, \ \forall \bar{A}_i$ *(reflexivity),*
2. $R(\bar{A}_i, \bar{A}_j) = R(\bar{A}_j, \bar{A}_i), \ \forall \bar{A}_i, \bar{A}_i$ *(symmetry).*

It is obvious that the fuzzy relation R on $\bar{A}_1, \bar{A}_2, ..., \bar{A}_N$ can be expressed by a N-by-N matrix $R^M = (r_{ij}), r_{ij} = R(\bar{A}_i, \bar{A}_j)$. Considering two fuzzy relations R_a and R_b, then the max–min operator $\circ$ is defined for two fuzzy relations matrixes R_a^M and R_b^M as follows.

$$(R_a^M \circ R_b^M)_{ij} = \bigvee_{k=1}^{N} (r_{ik}^{(a)} \wedge_{kj}^{(b)}), \tag{7.1}$$

where $r_{ik}^{(a)}$ is the element of R_a^M and $r_{kj}^{(b)}$ is the element of R_b^M, "$\wedge$" represents the minimize, "$\vee$" represents the maximize. It is clear that $R_a^M \circ R_b^M$ is also a fuzzy relations matrix and $r_{ij}^{(a)} \leq (R_a^M \circ R_b^M)_{ij}$ and $r_{ij}^{(b)} \leq (R_a^M \circ R_b^M)_{ij}$.

Next, the fuzzy equivalence relation is defined as follows.

Definition 7.4 (Fuzzy equivalence relations [Mirzaei and Rahmati, 2010])**.** *Given N fuzzy sets $\bar{A}_1, \bar{A}_2, ..., \bar{A}_N$, an operator $\widetilde{R} : (\bar{A}_i, \bar{A}_j) \rightarrow [0,1]$ is a fuzzy relation on $\bar{A}_1, \bar{A}_2, ..., \bar{A}_N$ if following properties are satisfied.*

1. $\widetilde{R}(\bar{A}_i, \bar{A}_i) = 1, \ \forall \bar{A}_i$ *(reflexivity),*
2. $\widetilde{R}(\bar{A}_i, \bar{A}_j) = R(\bar{A}_j, \bar{A}_i), \ \forall \bar{A}_i, \bar{A}_i$ *(symmetry).*
3. $\widetilde{R}^M = \widetilde{R}^M \circ \widetilde{R}^M$ *(transitivity).*

Here, $\widetilde{R}^M$ is the fuzzy relation matrix on $\widetilde{R}$ and $\circ$ is the max-min operator mentioned above.

Compared to fuzzy equivalence relations, fuzzy relations are much easier to obtain because fuzzy relations do not require transitivity. This leads researchers to find a way to construct the fuzzy equivalence relations based on the fuzzy relations and the max-min operator. The following theorem is provided to show that the max–min transitive closure $\widetilde{R}$ of a fuzzy relation R is a fuzzy equivalence relation.

Theorem 7.1. *Given a fuzzy relation R on $\bar{A}_1, \bar{A}_2, ..., \bar{A}_N$, there must be a finite $m \in \mathbb{Z}^+$ and an operator $\widetilde{R}$ satisfies the following conditions.*

1. $\widetilde{R}^M = \underbrace{R^M \circ R^M \circ ... \circ R^M}_{m}$,
2. $\widetilde{R}^M = \widetilde{R}^M \circ \widetilde{R}^M$,

where R^M is the fuzzy relations matrix of R, and $\widetilde{R}^M$ is the fuzzy relations matrix of $\widetilde{R}$. The operator $\widetilde{R}$ is called the max-min transitive closure of R.

Proof. Based on Theorem 5.1 and Theorem 5.2 in [Klir and Yklsit, 1995], R_T satisfies the following equations.

$$R_T^M = \bigvee_{k=1}^{\infty} R_{(k)}^M,$$
$$R_T^M = R_{(N-1)}^M.$$

Thus, $m = N-1$ (condition 1 is satisfied) and we have

$$\begin{aligned} R_T^M \circ R_T^M &= \bigvee_{k=1}^{\infty} R_{(k)}^M \circ \bigvee_{l=1}^{\infty} R_{(l)}^M \\ &= \bigvee_{k=1}^{\infty} \bigvee_{l=1}^{\infty} R_{(l)}^M \circ R_{(k)}^M \\ &= \bigvee_{l,k=1}^{\infty} R_{(l+k)}^M = R_T^M \end{aligned}$$

Thus, condition 2 is satisfied. ■

Theorem 7.1 provides a way to construct fuzzy equivalence relations through fuzzy relations. With the constructed fuzzy equivalence relations, we can use α-cut of R_T^M to cluster $\overline{A}_1$, $\overline{A}_2$, ..., $\overline{A}_N$. Specifically, the matrix of α-cut of R_T^M can be expressed by the following term.

$$\left(R_T^M(\alpha)\right)_{ij} = \begin{cases} 1, \ if \ \left(R_T^M\right)_{ij} \geq \alpha \\ 0, \ if \ \left(R_T^M\right)_{ij} < \alpha \end{cases}. \tag{7.2}$$

$R_T^M(\alpha)$ is a binary fuzzy equivalence relation matrix. Fuzzy sets that have the same corresponding rows of $R_T^M(\alpha)$ can be regarded as the same cluster. The selection of α is a decision-making process, and users can choose α based on their own requirements.

Traditional fuzzy equivalence relations are only for one type of fuzzy set, such as the set $\overline{S}_1 = \left\{\overline{A}_1, \overline{A}_2, \ldots, \overline{A}_{N_1}\right\}$, $\overline{A}_i \in F(\mathbb{R}^{n_1})$. However, for the HeUDA problem, there is always another set $\overline{S_2} = \left\{\overline{B}_1, \overline{B}_2, \ldots, \overline{B}_{N_2}\right\}$, $\overline{B}_i \in F(\mathbb{R}^{n_2})$ and $n_1 \neq n_2$, $N_1 \neq N_2$. In general, R_T^M of S_1 and S_2 are different and cannot get the same number of clusters through sharing the same α, which is the main obstacle when dealing with the HeUDA problem using fuzzy relations. In order to let R_T^M of S_1 and S_2 share the same α, this chapter designs a new shared fuzzy equivalence relation (SFER) which is based on two sets. Some new properties of traditional fuzzy equivalence are first discussed in Section 7.4, then, SFER is detailed based on these new properties.

7.3 Similarity between Fuzzy Vectors

In this section, we propose a new similarity between two fuzzy vectors in the n-D FG. The first subsection describes a new metric on n-D FG and proves its correctness. The second subsection proposes the new similarity based on the new metric.

7.3.1 A Metric on Fuzzy Geometry

To measure the distance between two fuzzy vectors in the n-D FG, researchers defined several metrics, comprising two main types: fuzzy metrics [Buckley and

Eslami, 1997a; Kaleva and Seikkala, 1984] and de-fuzzy metrics [Li et al., 2015]. Although the literature [Li et al., 2015] proposed a de-fuzzy metric based on the fuzzy metric defined in [Li et al., 2015], this de-fuzzy metric cannot satisfy the second condition of a metric space. In this section, a proper de-fuzzy metric $\mathcal{D}$ is proposed and we prove that $(F(\mathbb{R}^n),\ \mathcal{D})$ is a metric space. First, the detailed expression of a fuzzy vector $\overline{A}_i(a_{i1},\ a_{i2},\ \ldots,\ a_{in}) \in F(\mathbb{R}^n)$ (with the triangular membership function) is given as follows: for each $\overline{a}_{ij} \in F(\mathbb{R})$, its membership function is

$$\mu_{ij}(x|\bar{a}_{ij}) = \begin{cases} 0, & \forall x < a_{ij} - \rho_i \\ 1 - \frac{|x-a_{ij}|}{\rho_i}, & \forall |x - a_{ij}| \le \rho_i \ , x \in \mathbb{R}, \\ 0, & \forall x > a_{ij} + \rho_i \end{cases} \tag{7.3}$$

Based on the $\mu_{ij}(x|\bar{a}_{ij})$, $\mu_i(x|\bar{A}_i)$ is expressed by the following term.

$$\mu_i(x|\bar{A}_i) = \begin{cases} 0, & \exists x_j, x_j < a_{ij} - \rho_i \\ 1 - \frac{\|x-a_{ij}\|_1}{n\rho_i}, & \forall x_j, |x_j - a_{ij}| \le \rho_i \ , \\ 0, & \exists x_j, x_j > a_{ij} + \rho_i \end{cases} \tag{7.4}$$

where $x = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ and $\rho_i > 0$. Then, we define a new metric to measure the distance between two fuzzy vectors.

Definition 7.5. *Given two fuzzy vectors $\bar{A}_i \in F(\mathbb{R}^n)$ and $\bar{A}_j \in F(\mathbb{R}^n)$, the metric between $\bar{A}_i$ and $\bar{A}_j$ is defined by the map $\mathcal{D} : F(\mathbb{R}^n) \times F(\mathbb{R}^n) \to [0, +\infty)$:*

$$\mathcal{D}(\bar{A}_i, \bar{A}_j) = \frac{1}{n} \int_0^1 sup\{\mathcal{D}_\lambda(u,v) : \mathcal{D}_\lambda(u,v) \in \Omega(\lambda)\} d\lambda,$$

where

$$\Omega(\lambda) = \{d(u, \bar{A}_j(\lambda))\} \cup \{d(v, \bar{A}_i(\lambda))\},$$

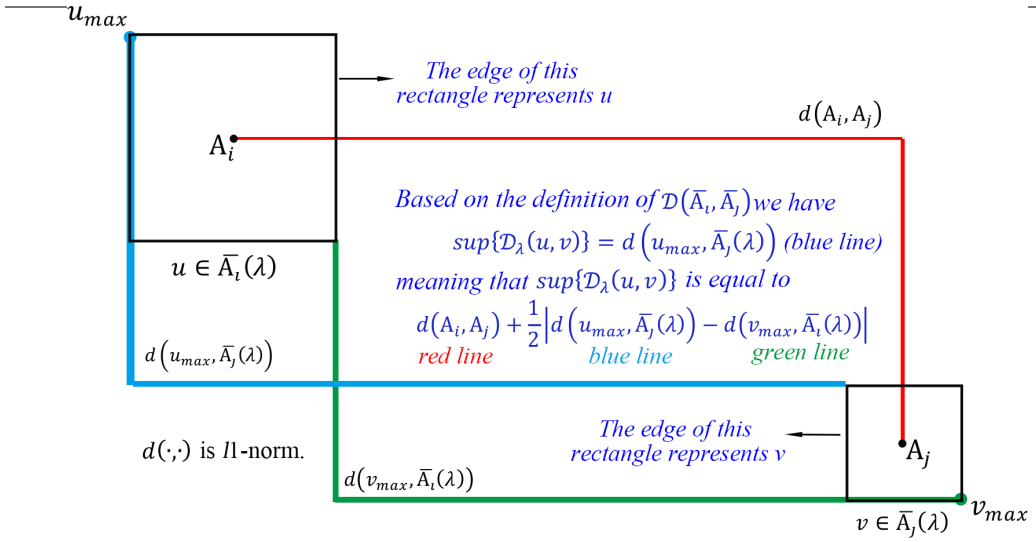


Figure 7.1: Relationships among $\sup\{D_\lambda(u,v) : D_\lambda(u,v) \in \Omega(\lambda)\}$, $d(u,\overline{A}_j(\lambda))$, $d(v,\overline{A}_i(\lambda))$ and $d(A_i,A_j)$

$u \in \bar{A}_i(\lambda), v \in \bar{A}_j(\lambda)$ and the first part of $\Omega(\lambda)$ collects L_1 distances between each u and $\bar{A}_j(\lambda)$ ($d(u,\bar{A}_j(\lambda)) = \min\{d(u,v), v \in \bar{A}_j(\lambda)\}$ means the minimum L_1 distances between u and all elements in $\bar{A}_j(\lambda)$), and the second part of $\Omega(\lambda)$ collects L_1 distances between v and $\bar{A}_i(\lambda)$ ($d(v,\bar{A}_i(\lambda)) = min\{d(v,u), u \in \bar{A}_i(\lambda)\}$ means the minimum L_1 distances between v and all elements in $\bar{A}_i(\lambda)$), and $d(u,v)$ represents the L_1 distance (ℓ_1-norm) between two n-dimension vector (u and v).

Remark 7.1. *To measure the distance between two fuzzy vectors is a key to define the fuzzy relation between them. Thus, we first propose a new measurement represented in Definition 7.5. $\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right)$ is the longest distance among 1) distances between v and $\overline{A}_i$ and 2) distances between u and $\overline{A}_j$.*

Figure 7.1 shows the meaning of $\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right)$ and indicates that the following

equation exists.

$$\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right) = \frac{1}{n}\int_0^1 d\left(A_i, A_j\right) + \frac{1}{2}\left|d\left(u, \overline{A}_j(\lambda)\right) - d\left(v, \overline{A}_i(\lambda)\right)\right| d\lambda. \tag{7.5}$$

Based on $\mu_i(x|\overline{A}_i)$, we derive the following equations:

$$\begin{aligned}\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right) &= \frac{1}{n}d\left(A_i, A_j\right) + \frac{1}{2}\int_0^1 \left|(1-\lambda)\rho_i - (1-\lambda)\rho_j\right| d\lambda \\ &= \frac{1}{n}d\left(A_i, A_j\right) + \frac{1}{2}\left|\rho_i - \rho_j\right| \int_0^1 (1-\lambda) d\lambda \\ &= \frac{1}{n}d\left(A_i, A_j\right) + \frac{1}{4}\left|\rho_i - \rho_j\right|.\end{aligned} \tag{7.6}$$

Theorem 7.2. *$(F(\mathbb{R}^n), \mathcal{D})$ is a metric space.*

Proof. To prove this theorem, we need to prove $\mathcal{D}$ can satisfy the following conditions for $\forall \overline{A}_i$, $\overline{A}_j$ and $\overline{A}_k \in F(\mathbb{R}^n)$.

1) $\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right) \geq 0$;

2) $\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right) = 0$ if and only if $\overline{A}_i = \overline{A}_j$;

3) $\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right) = \mathcal{D}\left(\overline{A}_j, \overline{A}_i\right)$;

4) $\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right) \leq \mathcal{D}\left(\overline{A}_i, \overline{A}_k\right) + \mathcal{D}\left(\overline{A}_k, \overline{A}_j\right)$;

where $\overline{A}_i = \overline{A}_j$ means that $A_i = A_j$ and $\rho_i = \rho_j$.

From the definition of $\mathcal{D}$, it is clear that conditions 1) and 3) are achieved. Because $d(\cdot, \cdot)$ represents the L_1 distance, condition 2) can be satisfied. For condition 4), we have

$$d\left(A_i, A_j\right) \leq d\left(A_i, A_k\right) + d\left(A_k, A_j\right)$$

$$\left|\rho_i - \rho_j\right| \leq \left|\rho_i - \rho_k\right| + \left|\rho_k - \rho_j\right|.$$

Thus, condition 4) is satisfied. ■

Theorem 7.2 shows the correctness of Definition 7.5 and gives a new perspective on the measurement of two fuzzy vectors.

7.3.2 Similarity between Fuzzy Vectors

The metric $\mathcal{D}$ can map two fuzzy vectors to a real positive number, which cannot be directly used as a fuzzy relation. Thus, this subsection transforms $\mathcal{D}$ into a fuzzy relation $R_{\mathcal{D}}$ as follows.

Lemma 7.1. *Given two fuzzy vectors where $\overline{A}_i \in F(\mathbb{R}^n)$ and $\overline{A}_j \in F(\mathbb{R}^n)$, if an operator $R_{\mathcal{D}}$: $\left(\overline{A}_i, \overline{A}_j\right) \mapsto [0,1]$ derived by $\mathcal{D}$ is defined as follows,*

$$R_{\mathcal{D}}\left(\overline{A}_i, \overline{A}_j\right) = e^{-\frac{\mathcal{D}\left(\overline{A}_i, \overline{A}_j\right)}{2\sigma^2}}, \tag{7.7}$$

then $R_{\mathcal{D}}$ is a fuzzy relation.

Proof. Because $f(x) = e^{\frac{-x}{2\sigma^2}}$ is monotonic when $x \in [0, +\infty)$ and $(F(\mathbb{R}^n), \mathcal{D})$ is a metric space, we have

$$R_{\mathcal{D}}\left(\overline{A}_i, \overline{A}_j\right) = R_{\mathcal{D}}\left(\overline{A}_j, \overline{A}_i\right). \tag{7.8}$$

Also, we have

$$R_{\mathcal{D}}\left(\overline{A}_i, \overline{A}_i\right) = e^{-\frac{0}{2\sigma^2}} = 1. \tag{7.9}$$

So, based on Definition 7.3 and Eqs. (7.8) and (7.9), the operator $R_{\mathcal{D}}$ satisfies reflexivity and symmetry, meaning that $R_{\mathcal{D}}$ is a fuzzy relation. ■

Based on the $R_{\mathcal{D}}$, we can obtain the fuzzy relations matrix of $R_{\mathcal{D}}$ and denote this matrix by $R_{\mathcal{D}}^M$, where $R_{\mathcal{D}}^M$ is a squared matrix and $\left(R_{\mathcal{D}}^M\right)_{ij} = R_{\mathcal{D}}\left(\overline{A}_i, \overline{A}_j\right)$,

$i, j = 1,2, \ldots, N$. Furthermore, the max-min transitive closure of $R_{\mathcal{D}}$ is denoted by $R_{T\mathcal{D}}$ and the $R^M_{T\mathcal{D}}$ is the fuzzy relations matrix of $R_{T\mathcal{D}}$, where $R^M_{T\mathcal{D}}$ is a squared matrix and $\left(R^M_{T\mathcal{D}}\right)_{ij} = R_{T\mathcal{D}}\left(\overline{A}_i, \overline{A}_j\right)$, , $i, j = 1,2, \ldots, N$. Based on Theorem 7.1, we also know that $R^M_{T\mathcal{D}} = \underbrace{R^M_{\mathcal{D}} \circ R^M_{\mathcal{D}} \circ \cdots \circ R^M_{\mathcal{D}}}_{N-1}$.

7.4 F-HeUDA via Shared Fuzzy Equivalence Relations

The HeUDA method is demonstrated as follows. Considering two feature sets of source and target domains: $S_1 = \{A_1, A_2, \ldots, A_{N_1}\}$, $A_i \in \mathbb{R}^{n_1}$, $S_2 = \{B_1, B_2, \ldots, B_{N_2}\}$, $B_i \in \mathbb{R}^{n_2}$ and $n_1 \neq n_2$, $N_1 \neq N_2$, the HeUDA methods aim to find a common feature set $S_c = \{C_1, C_2, \ldots, C_{N_c}\}$, $C_i \in \mathbb{R}^{n_c}$, $n_c = n_1 + n_2$, and use the labeled information of S_c (knowledge from the source domain) to label the unlabeled information of S_c (unlabeled target domain), where n_1 represents the number of samples in the source domain, n_2 represents the number of samples in the target domain, N_1 is the number of features in the source domain and N_2 is the number of features in the target domain.

Clearly, the core function of HeUDA methods is to simultaneously transform S_1 and S_2 to S_c but traditional fuzzy equivalence relations can only do this separately. Thus, the most important work of this chapter is to determine how to apply fuzzy equivalence relations to simultaneously transform S_1 and S_2 to S_c. To provide more detail, we propose SFER to let $R^M_{T\mathcal{D}}$s of S_1 and S_2 share the same α, which guarantees that S_1 and S_2 can be clustered as N_c categories with the same α. Figure 7.2 illustrates the difference between traditional fuzzy

equivalence relations and SFER.

Subsection 7.4.1 is the theoretical guarantee for SFER, which is formally demonstrated in subsection 7.4.2. Subsection 7.4.3 proposes an algorithm to learn the parameters of SFER, and the last subsection introduces how to transfer knowledge from a source domain to a target domain using SFER.

7.4.1 Theoretical Guarantees

This subsection gives two properties of fuzzy equivalence relations. The first (Theorem 7.3) demonstrates how many different real numbers exist in the fuzzy equivalence relations matrix and the second (Theorem 7.4) demonstrates how the number of clusters changes when α changes.

Lemma 7.2. *Given* $S = \{A_1, A_2, \ldots, A_N\}$ *and a fuzzy set* $\overline{S} = \left\{\overline{A}_1, \overline{A}_2, \ldots, \overline{A}_N\right\}$ *on* S*, then, for* $R^M_{T\mathcal{D}}$ *of* $\overline{S}$*, we have*

$$R^M_{T\mathcal{D}} = \begin{pmatrix} R^{N-1}_{(N)} \bigvee \left(R^{N-1}_{(N-2)} \circ r_N \circ r_N^T \circ R^{N-1}_{(N-2)}\right) & R^{N-1}_{(N-1)} \circ r_N \\ r_N^T \circ R^{N-1}_{(N-1)} & 1 \end{pmatrix},$$

where

$$R^M_{\mathcal{D}} = \begin{pmatrix} R^{N-1} & r_N \\ r_N^T & 1 \end{pmatrix}_{N\times N}, \; R^{N-1}_{(N)} = \underbrace{R^{N-1} \circ \ldots \circ R^{N-1}}_{N}, \; R^{N-1}_{(0)} = \boldsymbol{I}$$

and R^{N-1} *is a* $N-1$ *by* $N-1$ *matrix,* r_N *is a* N *dimensional vector.*

Proof. Based on the max-min operator $\circ$, we arrive at the following equation:

$$R^M_{\mathcal{D}} \circ R^M_{\mathcal{D}} = \begin{pmatrix} R^{N-1}_{(2)} \vee \left(r_N \circ r_N^T\right) & R^{N-1}_{(2)} \circ r_N \\ r_N^T \circ R^{N-1}_{(2)} & 1 \end{pmatrix}. \tag{7.10}$$

In terms of Eq. (7.10), $R^M_{T\mathcal{D}} = \underbrace{R^M_{\mathcal{D}} \circ R^M_{\mathcal{D}} \circ \cdots \circ R^M_{\mathcal{D}}}_{N}$ can be expressed as follows

$$R^M_{T\mathcal{D}} = \begin{pmatrix} R^{N-1}_{(N)} \bigvee_{k=0}^{N-2} \left(R^{N-1}_{(k)} \circ r_N \circ r_N^T \circ R^{N-1}_{(N-2-k)} \right) & R^{N-1}_{(N-1)} \circ r_N \\ r_N^T \circ R^{N-1}_{(N-1)} & 1 \end{pmatrix},$$

where $R^{N-1}_{(k)} \circ r_N \circ r_N^T \circ R^{N-1}_{(N-2-k)}$, $k = 0, \ldots, N-2$, cannot satisfy the symmetry but $\bigvee_{k=0}^{N-2} \left(R^{N-1}_{(k)} \circ r_N \circ r_N^T \circ R^{N-1}_{(N-2-k)} \right)$ is a symmetrical matrix. Based on the meaning of operator $\vee$, we have

$$\bigvee_{k=0}^{N-2} \left(R^{N-1}_{(k)} \circ r_N \circ r_N^T \circ R^{N-1}_{(N-2-k)} \right) = R^{N-1}_{(N-2)} \circ r_N \circ r_N^T \circ R^{N-1}_{(N-2)}.$$

So, this lemma is proved. ■

Lemma 7.2 demonstrates the structure of the $R^M_{T\mathcal{D}}$ of $\overline{S}$ from the perspective of the block matrix and provides a useful way to prove Theorem 7.3.

Theorem 7.3. *Given $S = \{A_1, A_2, \ldots, A_N\}$ and a fuzzy set $\overline{S} = \left\{\overline{A}_1, \overline{A}_2, \ldots, \overline{A}_N\right\}$ on S, if the fuzzy relations matrix $R^M_{\mathcal{D}}$ of $\overline{S}$ has $N(N-1)/2+1$ different elements, then $R^M_{T\mathcal{D}}$ of $\overline{S}$ only has N different elements: $r_1 < r_2 < \cdots < r_{N-1} < r_N = 1$.*

Proof. We use mathematical induction to prove this theorem based on Lemma 7.2.

1) First, when N=2, obviously, $R^M_{T\mathcal{D}}$ only has 2 different elements $r_1^2 < r_2^2 = 1$;

2) Then, we assume that the R^{N-1}[1] has $(N-1)(N-2)$/2+1 different elements and $R^{N-1}_{(N-2)}$ (the $R^M_{T\mathcal{D}}$ of the subset $\overline{S^{N-1}}$) only has $N-1$ elements: $r_1^{N-1} < r_2^{N-1} < \cdots < r_{N-1}^{N-1} = 1$.

3) Based on Lemma 2, we have

$$R^M_{T\mathcal{D}} = \begin{pmatrix} R^{N-1}_{(N)} \bigvee \left(R^{N-1}_{(N-2)} \circ r_N \circ r_N^T \circ R^{N-1}_{(N-2)} \right) & R^{N-1}_{(N-1)} \circ r_N \\ r_N^T \circ R^{N-1}_{(N-1)} & 1 \end{pmatrix}$$

[1] the $R^M_{\mathcal{D}}$ of the subset $\overline{S^{N-1}} = \left\{\overline{A}_1, \overline{A}_2, \ldots, \overline{A_{N-1}}\right\}$.

and we know $R_{(N)}^{N-1}$ only has $N-1$ elements. So,

(a) we first need to prove $R_{(N)}^{N-1} \bigvee \left(R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1} \right)$ only has $N-1$ different elements,

(b) then, we need to prove one of the elements in $R_{(N-1)}^{N-1} \circ r_N$ do not exist in

$$R_{(N)}^{N-1} \bigvee \left(R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1} \right).$$

Proof of (a). We use $r_{ij}^{(N1)}$ to express elements in $R_{(N-2)}^{N-1}$ and $R_{(N)}^{N-1}$ ($R_{(N-2)}^{N-1} = R_{(N)}^{N-1}$), and use $r_{ij}^{(rr)}$ to express elements in $r_N \circ r_N^T$, and use $r_{ij}^{(RrR)}$ to express elements in $R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1}$. Then, we have

$$r_{ij}^{(RrR)} = \bigvee_{k=1}^{N-1} \bigvee_{k_1=1}^{N-1} r_{ik}^{(N1)} \wedge r_{kk_1}^{(rr)} \wedge r_{k_1 j}^{(N1)}, \tag{7.11}$$

which means that we need to consider if $r_{ij}^{(RrR)}$ is greater than $r_{ij}^{(N1)}$. Without loss of generality, we select elements (more than 2) equaling r_m^{N-1} ($m < N-1$) as examples: $r_{i_1 j_1}^{(N1)} = r_{i_2 j_1}^{(N1)} = \cdots = r_{i_t j_1}^{(N1)} = r_{i_t j_2}^{(N1)} = \cdots = r_{i_t j_z}^{(N1)} = r_m^{N-1}$ (because $R_{\mathcal{D}}^M$ of $\overline{S}$ has $N(N-1)/2+1$ different elements, there are two of the same elements in the same rows or columns of $R_{T\mathcal{D}}^M$). Based on Eq. (7.11), if one element of $r_{ij}^{(N1)}$ equaling r_m^{N-1} is lower than $r_{ij}^{(RrR)}$, then we have

$$r_{i_1 j_1}^{(RrR)} > r_m^{N-1},\ r_{i_2 j_1}^{(RrR)} > r_m^{N-1}, \ldots r_{i_t j_1}^{(RrR)} > r_m^{N-1}, \tag{7.12}$$

$$r_{i_t j_2}^{(RrR)} > r_m^{N-1}, \ldots r_{i_t j_z}^{(RrR)} > r_m^{N-1}, \tag{7.13}$$

meaning that all elements of $r_{ij}^{(N1)}$ equaling r_m^{N-1} will be lower than $r_{ij}^{(RrR)}$ and will be replaced with one element of r_N and elements of $R_{(N-2)}^{N-1}$ in

$$R_{(N)}^{N-1} \bigvee \left(R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1} \right).$$

Thus, $R_{(N)}^{N-1} \bigvee \left(R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1} \right)$ still has $N-1$ different elements (If there only are two elements equaling r_m^{N-1}, they only can be replaced with an element of r_N, meaning that $R_{(N)}^{N-1} \bigvee \left(R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1} \right)$ still has $N-1$ different elements).

Proof of (b). Because $R_{\mathcal{D}}^M$ of $\overline{S}$ has $N(N-1)/2+1$ different elements, there is only one maximum element in r_N and this maximum element will only exist in the diagonal of $r_N \circ r_N^T$, denoted by $r_{k_m,\, k_m}^{(rr)}$. Based on the max-min operator $\circ$, it is easy to know that the k_m^{th} element of $R_{(N-1)}^{N-1} \circ r_N$ is $r_{k_m,\, k_m}^{(rr)}$. Then, we prove that $r_{k_m,\, k_m}^{(rr)}$ does not exist in $R_{(N)}^{N-1} \bigvee \left(R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1} \right)$.

If $\exists i,\, j,\, s.t.\, r_{ij}^{(RrR)} = r_{k_m,\, k_m}^{(rr)}$, we have

$$r_{ij}^{(RrR)} = r_{ik_m}^{(N1)} \wedge r_{k_m,\, k_m}^{(rr)} \wedge r_{k_m j}^{(N1)}, \tag{7.14}$$

meaning that $r_{ik_m}^{(N1)} \wedge r_{k_m j}^{(N1)} > r_{k_m,\, k_m}^{(rr)}$. Because

$$r_{ij}^{(N1)} = \bigvee_{k=1}^{N-1} r_{ik}^{(N1)} \wedge r_{kj}^{(N1)} \geq r_{ik_m}^{(N1)} \wedge r_{k_m j}^{(N1)}, \tag{7.15}$$

we have $r_{ij}^{(N1)} > r_{ij}^{(RrR)} = r_{k_m,\, k_m}^{(rr)}$, which means that $r_{k_m,\, k_m}^{(rr)}$ does not exist in

$$R_{(N)}^{N-1} \bigvee \left(R_{(N-2)}^{N-1} \circ r_N \circ r_N^T \circ R_{(N-2)}^{N-1} \right).$$

Based on (a) and (b), we prove that $R_{T\mathcal{D}}^M$ has only N different elements. Combining 1), 2) and 3), this theorem is proved. ■

Theorem 7.3 demonstrates an important property for a fuzzy equivalence matrix and derives Lemma 7.3 and Theorem 7.4.

Lemma 7.3. *Given a fuzzy set* $\overline{S} = \left\{ \overline{A}_1,\, \overline{A}_2,\, \ldots,\, \overline{A}_N \right\}$, *if the fuzzy relations matrix* $R_{\mathcal{D}}^M$ *of* $\overline{S}$ *has* $N(N-1)/2+1$ *different elements, then, for* $R_{T\mathcal{D}}^M$ *of* $\overline{S}$, *we have*

$$rank\left(R_{T\mathcal{D}}^M(r_i) \right) = rank\left(R_{T\mathcal{D}}^M(r_{i-1}) \right) + 1,\ i = 2,\, \ldots,$$

Proof. Because $R^M_{T\mathcal{D}}(r_{i-1}) \geq R^M_{T\mathcal{D}}(r_i)$ holds for $r_i,\ i = 2,\ \ldots,\ N$, the element "1" of $R^M_{T\mathcal{D}}(r_i)$ will change to "0" when i increasing, meaning that we have that $rank\left(R^M_{T\mathcal{D}}(r_i)\right) \geq rank\left(R^M_{T\mathcal{D}}(r_{i-1})\right)$ holds. Then, we will prove that $rank\left(R^M_{T\mathcal{D}}(r_i)\right)$ is greater than $rank\left(R^M_{T\mathcal{D}}(r_{i-1})\right)$.

We use $r^{(i)}_{ls}$ to represent the element in the i^{th} row and s^{th} column of $R^M_{T\mathcal{D}}(r_i)$, and $r^{(i)}_{l*}$ to represent elements in i^{th} of $R^M_{T\mathcal{D}}(r_i)$. For a specific i, we assume $rank\left(R^M_{T\mathcal{D}}(r_{i-1})\right) = k$ and $\forall l_1,\ l_2 \in I^{(i-1)}_t,\ r^{(i-1)}_{l_1*} = r^{(i-1)}_{l_2*}$, where $I^{(i-1)}_t$ is a set to collect indicators of same rows of $R^M_{T\mathcal{D}}(r_{i-1})$ and $t = 1,\ 2,\ \ldots,\ k$. Without loss of generality, we assume that $r^{(i-1)}_{ls} = 1$ but $r^{(i)}_{ls} = 0$ and analyze the value of $rank\left(R^M_{T\mathcal{D}}(r_i)\right)$.

Because the reflexivity of $R^M_{T\mathcal{D}}(r_{i-1})$, we know $\exists t_0\ s.\ t.$

$$\forall l_1,\ l_2 \in I^{(i-1)}_{t_0},\ r^{(i-1)}_{l_1 l_2} = 1,$$

and $l \in I^{(i-1)}_{t_0},\ s \in I^{(i-1)}_{t_0}$. This means $r^{(i)}_{l*} \neq r^{(i)}_{s*}$ (because $r^{(i)}_{ls} = 0 \neq 1 = r^{(i)}_{ss}$). So, $I^{(i-1)}_{t_0}$ will be divided into two sets when r_{i-1} is replaced with r_i, meaning that $rank\left(R^M_{T\mathcal{D}}(r_i)\right) > rank\left(R^M_{T\mathcal{D}}(r_{i-1})\right)$.

Based on Theorem 7.3, we know $rank\left(R^M_{T\mathcal{D}}(r_i)\right)$ at most has N values: $rank\left(R^M_{T\mathcal{D}}(r_1)\right), \ldots, rank\left(R^M_{T\mathcal{D}}(r_N)\right)$. Since $rank\left(R^M_{T\mathcal{D}}(r_i)\right) > rank\left(R^M_{T\mathcal{D}}(r_{i-1})\right)$ and $rank\left(R^M_{T\mathcal{D}}(r_i)\right) \leq N$, we have $rank\left(R^M_{T\mathcal{D}}(r_i)\right) = rank\left(R^M_{T\mathcal{D}}(r_{i-1})\right) + 1$. ■

Theorem 7.4. *Given a fuzzy set* $\overline{S} = \left\{\overline{A}_1,\ \overline{A}_2,\ \ldots,\ \overline{A}_N\right\}$*, if the fuzzy relations matrix* $R^M_{\mathcal{D}}$ *of* $\overline{S}$ *has* $N(N-1)/2+1$ *different elements and* $\alpha \in (r_{i-1},\ r_i]$*, then* $\overline{S}$ *will be clustered as* i *categories, where* $\{r_i,\ i = 1,\ 2\ldots,\ N\}$ *is the set of* N *different elements of* $R^M_{T\mathcal{D}}$ *of* $\overline{S}$ *and* $r_1 < r_2 < \cdots < r_{N-1} < r_N = 1$.

Proof. When $\alpha \in (r_{i-1},\ r_i]$, $R^M_{T\mathcal{D}}(\alpha)$ of $\overline{S}$ is equal to $R^M_{T\mathcal{D}}(r_i)$. Based on Lemma 3, we know $rank\left(R^M_{T\mathcal{D}}(r_i)\right) = i$, which means that $\overline{S}$ will be clustered as i categories

$I_t^{(i)}$, where $I_t^{(i-1)}$ is a set to collect indicators of same rows of $R_{T\mathcal{D}}^M(r_{i-1})$ and $\forall t_1,\ t_2 \in \{1,\ 2,\ \ldots,\ i\},\ I_{t_1}^{(i)} \cap I_{t_2}^{(i)} = \emptyset$ and $\bigcup I_t^{(i)} = \{1,\ 2,\ \ldots,\ N\}$. ■

Theorem 7.4 demonstrates a significant property for fuzzy equivalence: the number of clusters is decided by the value of r_i, which means that we can use r_i of two domains to represent how many clusters both domains have when two domains are applied by the same α.

7.4.2 Shared Fuzzy Equivalence Relations (SFER)

Based on subsection 7.4.1, the aim of the SFER is to let two domains S_1 and S_2 have almost the same $r_i,\ i = 1,\ \ldots, M = min(N_1,\ N_2)$, denoted by r_i^{S1} for S_1 and r_i^{S2} for S_2. Formally, we define a cost function $\boldsymbol{J_1}\left(S_1,\ S_2,\ \rho_l^{S1}, \rho_k^{S2}\right)$ to express the divergence between r_i^{S1} and r_i^{S2}, which is shown as follows.

$$\boldsymbol{J_1}\left(S_1,\ S_2,\ \rho_l^{S1}, \rho_k^{S2}\right) = \sum_{i=1}^{M-1}\left(r_i^{S1}\left(S_1,\ \rho_l^{S1}\right) - r_i^{S2}\left(S_2,\ \rho_k^{S2}\right)\right)^2, \quad (18)$$

where ρ_l^{S1} is the parameter vector for elements in S_1 and ρ_k^{S2} is the parameter vector for elements in S_2 (parameters of the triangular membership function). Thus, the SFER aims to minimize the $\boldsymbol{J}\left(S_1,\ S_2,\ \rho_l^{S1}, \rho_k^{S2}\right)$ by tuning ρ_l^{S1} and ρ_k^{S2}, expressed by

$$\min_{\rho_l^{S1}, \rho_k^{S2}} \boldsymbol{J_1}\left(S_1,\ S_2,\ \rho_l^{S1}, \rho_k^{S2}\right)$$

$$subject\ to\ \rho_l^{S1} > 0, \rho_k^{S2} > 0. \quad (7.16)$$

If the cost function $\boldsymbol{J_1}\left(S_1,\ S_2,\ \rho_l^{S1}, \rho_k^{S2}\right)$ is approaching 0, r_i^{S1} is almost same as r_i^{S2}, which means that domains S_1 and S_2 will have the same number of clusters when applying the same α to two domains (Figure 7.2b).

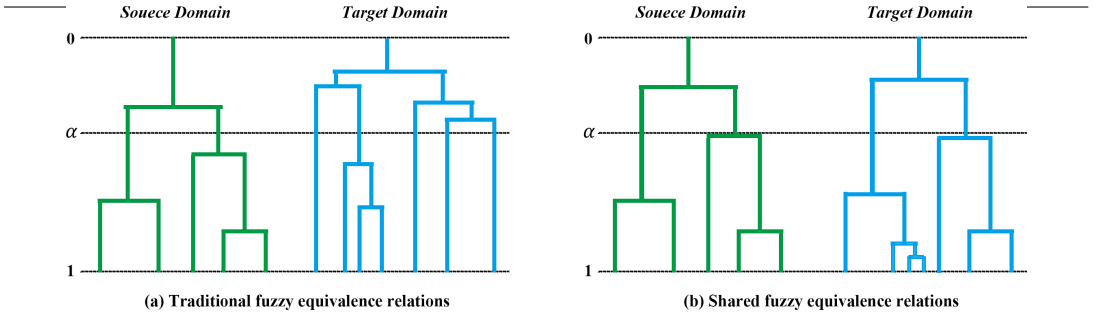


Figure 7.2: Traditional fuzzy equivalence relations v.s. SFER. In subfigure (a), two domains clearly cannot use the same α to obtain the same number of clusters. But, in SFER, two domains have a much bigger probability of using the same α to obtain the same number of clusters.

7.4.3 Learning Process of SFER

To learn the best ρ_l^{S1} and ρ_k^{S2}, we first consider how to minimize $\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2$, where $1 \leq i_0 \leq M-1$ and $r_{i_0}^{S1} < r_{i_0+1}^{S1}$ and $r_{i_0}^{S2} < r_{i_0+1}^{S2}$. Because of the nature of max-min operator, $r_{i_0}^{S1}$ equals one element in $R_{\mathcal{D}}^M$ of S_1 and $r_{i_0}^{S2}$ equals one element in $R_{\mathcal{D}}^M$ of S_2, which means

$$r_{i_0}^{S1} = exp\left(-\frac{\frac{1}{n}d\left(A_{l_0}, A_{l'_0}\right) + \frac{1}{4}\left|\rho_{l_0}^{S1} - \rho_{l'_0}^{S1}\right|}{2\sigma^2}\right),$$

$$r_{i_0}^{S2} = exp\left(-\frac{\frac{1}{n}d\left(B_k, B_{k'_0}\right) + \frac{1}{4}\left|\rho_{k_0}^{S2} - \rho_{k'_0}^{S2}\right|}{2\sigma^2}\right).$$

It is obvious that $r_{i_0}^{S1}$ is related to $\left|\rho_{l_0}^{S1} - \rho_{l'_0}^{S1}\right|$, so the constrained conditions of $\boldsymbol{J_1}\left(S_1,\, S_2,\, \rho_l^{S1}, \rho_k^{S2}\right)$ do not affect the value of $\boldsymbol{J_1}$. Thus, we can define a new optimization problem (no constrained condition) related to ρ_l^{S1} and ρ_k^{S2} as follows:

$$\min_{\rho_l^{S1}, \rho_k^{S2}} J_1\left(S_1,\, S_2,\, \rho_l^{S1}, \rho_k^{S2}\right).$$

Then, we can obtain the gradients of $\rho_{l_0}^{S1}$, $\rho_{l_0'}^{S1}$, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$ with respect to $\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2$:

$$\frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{l_0}^{S1}} = \frac{-2\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2 r_{i_0}^{S1}\left|\rho_{l_0}^{S1} - \rho_{l_0'}^{S1}\right|}{2\sigma^2 sign\left(\rho_{l_0}^{S1} - \rho_{l_0'}^{S1}\right)},$$

$$\frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{l_0'}^{S1}} = \frac{2\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2 r_{i_0}^{S1}\left|\rho_{l_0}^{S1} - \rho_{l_0'}^{S1}\right|}{2\sigma^2 sign\left(\rho_{l_0}^{S1} - \rho_{l_0'}^{S1}\right)},$$

$$\frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{k_0}^{S2}} = \frac{-2\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2 r_{i_0}^{S1}\left|\rho_{k_0}^{S2} - \rho_{k_0'}^{S2}\right|}{2\sigma^2 sign\left(\rho_{k_0}^{S2} - \rho_{k_0'}^{S2}\right)},$$

$$\frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{k_0'}^{S2}} = \frac{-2\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2 r_{i_0}^{S1}\left|\rho_{k_0}^{S2} - \rho_{k_0'}^{S2}\right|}{2\sigma^2 sign\left(\rho_{k_0}^{S2} - \rho_{k_0'}^{S2}\right)}.$$

Inspired by incremental gradient descent, for each $r_{i_0}^{S1}$, we can optimize $\rho_{l_0}^{S1}$ and $\rho_{l_0''}^{S1}$ once using the following equations.

$$\rho_{l_0}^{S1} = \rho_{l_0}^{S1} - \eta \frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{l_0}^{S1}}, \tag{7.17}$$

$$\rho_{l_0'}^{S1} = \rho_{l_0'}^{S1} - \eta \frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{l_0'}^{S1}}. \tag{7.18}$$

Similarly, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$ are optimized once using the following equations.

$$\rho_{k_0}^{S2} = \rho_{k_0}^{S2} - \eta \frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{k_0}^{S2}}, \tag{7.19}$$

$$\rho_{k_0'}^{S1} = \rho_{k_0'}^{S1} - \eta \frac{\partial\left(r_{i_0}^{S1} - r_{i_0}^{S2}\right)^2}{\partial \rho_{k_0'}^{S2}}. \tag{7.20}$$

Algorithm 7.1 Learning process of MsSFER

1: Input: Source feature sets S_1, target feature set S_2 and IterM.
2: Initialize parameter sets ρ_l^{S1} and ρ_k^{S2}
3: Generate a small positive real number ϵ
for $iter = 1 : IterM$ **do**
  4: Calculate $\widetilde{R}_{\mathcal{D}}^{M}$ of S_1 and S_2
  5: Extract r_{i_0} and r_{i_0} from $\widetilde{R}_{\mathcal{D}}^{M}$ of S_1 and S_2
  for $j_0 = 1 : N_{\min} - 1$ **do**
    6: Find l_0 and $l_0^{'}$ such that $(R_{\mathcal{D}}^{M}(S_1))_{l_0,l_0'} = r_{i_0}^{S1}$
    7: Find k_0 and $k_0^{'}$ such that $(R_{\mathcal{D}}^{M}(S_2))_{k_0,k_0'} = r_{i_0}^{S2}$
    8: Update $\rho_{l_0}^{S1}$ and $\rho_{l_0'}^{S1}$ using Eqs. (7.17) and (7.18)
    9: Update $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$ using Eqs. (7.19) and (7.20)
  end
end
10: Update $\rho_l^{S1} = \rho_l^{S1} + \min\{\rho_l^{S1}\} - \epsilon$; // Ensure $\rho_l^{S1} > 0$.
11: Update $\rho_k^{S2} = \rho_k^{S2} + \min\{\rho_k^{S1}\} - \epsilon$; // Ensure $\rho_k^{S2} > 0$.
12: Output: ρ_l^{S1}, ρ_k^{S2}.

So, using $r_{i_0}^{S1}$ and $r_{i_0}^{S2}$, we can optimize $\rho_{l_0}^{S1}$, $\rho_{l_0'}^{S1}$, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$ once (no iterations). This means $\rho_{l_0}^{S1}$, $\rho_{l_0'}^{S1}$, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$ can be optimized $M-1$ times using different $r_{i_0}^{S1}$ and $r_{i_0}^{S2}$, where $1 \le i_0 \le M-1$. With the optimized $\rho_{l_0}^{S1}$, $\rho_{l_0'}^{S1}$, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$, $R_{\mathcal{D}}^{M}$ of S_1 and $R_{\mathcal{D}}^{M}$ of S_2 will be updated. Then $\rho_{l_0}^{S1}$, $\rho_{l_0'}^{S1}$, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$ will be optimized M times using different $r_{i_0}^{S1}$ and $r_{i_0}^{S2}$ again. Within limited iterations (or reaching a termination condition), we can obtain the optimized $\rho_{l_0}^{S1}$, $\rho_{l_0'}^{S1}$, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$. Based on these procedures, Algorithm 7.1 is designed to optimize $\rho_{l_0}^{S1}$, $\rho_{l_0'}^{S1}$, $\rho_{k_0}^{S2}$ and $\rho_{k_0'}^{S2}$ as follows.

7.4.4 F-HeUDA via SFER (F-HeUDA)

In this section, we introduce how to select α, how to generate S_C and how to transfer knowledge from the source domain to the target domain.

After executing Algorithm 7.1, we obtain $r_{i_0}^{S1}$ and $r_{i_0}^{S2}$ generated by the best ρ_l^{S1} and ρ_k^{S2}, $i_0 = 0, \ldots, M-1$. So, the intervals of sharing α between two domains can be calculated (two domains can share the same α when α is in these intervals). These intervals can be expressed as follows:

$$\left[\max\left(r_{i_0}^{S1}, r_{i_0}^{S2}\right), min\left(r_{i_0+1}^{S1}, r_{i_0+1}^{S2}\right)\right),$$

where $r_0^{S1} = r_0^{S2} = 0$ and $r_M^{S1} = r_M^{S2} = 1$. Specifically, we can obtain following intervals,

$$\left[0, min\left(r_1^{S1}, r_1^{S2}\right)\right), \left[\max\left(r_1^{S1}, r_1^{S2}\right), min\left(r_2^{S1}, r_2^{S2}\right)\right), \ldots, \left[\max\left(r_{M-1}^{S1}, r_{M-1}^{S2}\right), 1\right).$$

If the α, in these intervals, is selected, S_1 and S_2 have the same number of clusters. Then, we select the α which is in the largest interval as the best α (because two domains can share most information in this largest interval).

Remark 7.2. *If we select the* $\alpha \in \left[r_{i_0}^{S1}, r_{i_0+1}^{S1}\right)$, *we do know* S_1 *is clustered into* $i_0 + 1$ *clusters but do not guarantee that* S_2 *also has* $i_0 + 1$ *clusters. Thus, we need to select* α *belonging to* $\left[\max\left(r_{i_0}^{S1}, r_{i_0}^{S2}\right), min\left(r_{i_0+1}^{S1}, r_{i_0+1}^{S2}\right)\right)$ *to make sure* S_1 *and* S_2 *have* $i_0 + 1$ *clusters.*

Based on the SFER (with the best α), both $S_1 = \{A_1, \ldots, A_{N_1}\}$ and $S_2 = \{B_1, \ldots, B_{N_2}\}$ can be clustered as N_c clusters, such as

$$S_1 = \left\{\{A_1, A_2\}_1, \{A_3, A_4, A_5\}_2, \ldots, \{A_{N_1}\}_{N_c}\right\},$$

$$S_2 = \left\{\{B_1\}_1, \{B_2, B_3, B_4\}_2, \ldots, \{B_{N_1-1}, B_{N_1}\}_{N_c}\right\}.$$

Next, we generate the latent features of the two domains using an addition operator, such as

$$S_1^{latent} = \{A_1 + A_2, A_3 + A_4 + A_5, \ldots, A_{N_1}\},$$

$$S_2^{latent} = \left\{B_1,\, B_2+B_3+B_4,\, \ldots,\, B_{N_1-1}+B_{N_1}\right\}.$$

Then, the standard score is used to normalize each latent feature and we have

$$S_1^{common} = \left\{f(A_1+\ A_2),\, f(A_3+A_4+A_5),\, \ldots, f\left(A_{N_1}\right)\right\},$$

$$S_2^{common} = \left\{f(B_1),\, f(B_2+B_3+B_4),\, \ldots, f\left(\ B_{N_1-1}+B_{N_1}\right)\right\}.$$

where $f(\cdot)$ represents the standard score function. Thus, we obtain the common feature set $S_c = \left\{C_1,\, C_2,\, \ldots,\, C_{N_c}\right\}$, $C_i \in \mathbb{R}^{n_c}$, $n_c = n_1 + n_2$ (without loss of generality, the former n_1 samples in S_c comes from source domain). Finally, we can use the labeled information of S_c (knowledge from the source domain) to label the unlabeled information of S_c (unlabeled target domain) using a *support vector machine* (SVM).

7.5 Experiments

In this section, we use four real datasets to test the proposed F-HeUDA method and analyze the convergence of Algorithm 7.1 and how α influences the method's performance.

7.5.1 Dataset Description and Parameters Setting

To validate the effectiveness of the proposed method on small datasets, we select four datasets from the UCI Machine Learning Repository[2]. The details of these datasets are provided in Table 6.1 (Credit and Cancer datasets). For both dataset, we generate four transfer tasks, as shown in Table 6.2 (Credit and Cancer tasks). Each task is described in detail as follows:

[2]The line of UCI Machine Learning Repository is http://archive.ics.uci.edu/ml/index.html

1) Task 1- G2A: Assume that the German data is labeled and the Australian data is unlabeled and has far fewer samples than the German data. Label "1" means "good credit" and label "2" means "bad credit". This task aims to answer the question: "Can we use knowledge from German credit records to label unlabeled Australian data (small dataset)?"

2) Task 2-A2G: Assume that the Australian data is labeled and the German data is unlabeled and has far fewer samples than the Australian data. Label "1" means "good credit" and label "2" means "bad credit". This task aims to answer the question: "Can we use knowledge from Australian credit records to label unlabeled German data (small dataset)?"

3) Task 3-CO2CD: Assume that in the Breast Cancer Wisconsin (Original) dataset (CO in Table 6.2) "1" represents "malignant" and "0" represents "benign". Another unlabeled dataset related to breast cancer also exists but has far fewer samples than the CO dataset. This task aims to answer the question: "Can we use the knowledge from CO to label 'malignant' in the unlabeled small dataset?"

4) Task 4-CD2CO: Assume that in the Breast Cancer Wisconsin (Diagnostic) dataset (CD in Table 6.2) "1" represents "malignant" and "0" represents "benign". Another unlabeled dataset related to breast cancer also exists. This task aims to answer the question: "Can we use the knowledge from CD to label 'malignant' in the small unlabeled dataset?"

For the Algorithm 7.1, this chapter sets IterM as 1000, σ as 3 and η as 0.5. For SVM, we adopt LIBSVM with default parameters: the SVM type is C-SVM with C equaling to 1, kernel function is radial basis function with gamma equaling to $\frac{1}{\#\ of\ Attributes}$, epsilon, the tolerance of termination criterion, is set as 0.001.

7.5.2 Prediction Performance

In this section, we describe the prediction performance of the proposed method and baselines. We select two non-transfer methods, A1 and CM, as baselines and three transfer methods, dimension reduction GFK (DG), KCCA, random linear monotonic maps GFK (RLG), GLG as the main baselines. The A1 method labels all samples of a target domain with 1 and the CM method clusters samples of a target domain and gives each sample a pseudo label (clustering method is the fuzzy c-means method). The DG method is based on dimension reduction technology, introduced in as a baseline of HeUDA methods, and KCCA is based on canonical correlation analysis and requires both domains to have the same number of samples. The RLG and GLG methods are proposed in Chapter 6. The former does not require both domains to have the same number of samples but the latter requires this condition. To test these methods' prediction performance when a target domain is a small dataset, we carried out experiments when n_2 (the number of samples in a target domain) is 40, 100, 200, 300, 400. For each method, we carried out the experiments 20 times to obtain reliable results.

The results, as illustrated in Figure 7.3, clearly show the prediction performance of each method. From this figure, it is apparent that the proposed method outperforms the others when n_2 is small. The CM and DG methods have lower mean accuracy and higher standard deviation than the other methods in most cases. For the KCCA method, its performance increases when n_2 increases but its accuracy is always lower than RLG, GLG and F-HeUDA. This is because the KCCA method cannot reliably transfer knowledge from the source domain to the target domain. The RLG method has better performance than the GLG method

Table 7.1: The overall performance of each method on four tasks.

Tasks	A1	CM	DG	KCCA	RLG	GLG	F-HeUDA
A2G	50.00%	49.15%	51.48%	50.01%	58.46%	58.98%	59.74%
G2A	50.00%	50.75%	47.02%	51.52%	72.10%	72.12%	75.38%
CD2CO	50.00%	22.04%	19.54%	65.00%	96.59%	96.57%	96.93%
CO2CD	50.00%	23.40%	26.96%	55.24%	87.94%	87.98%	88.51%

when n_2 is 40 and 100 in some cases, but it is still unstable when n_2 increases. The GLG method has good performance when n_2 increases but it cannot guarantee reliable results when n_2 is small (see task G2A). From Figure 7.3, the following results can be observed.

1) The KCCA method has a worse performance than non-transfer methods in some cases;

2) The RLG, GLG and F-HeUDA methods have more stable results than the non-transfer methods and DG and KCCA;

3) Compared to RLG and GLG, the proposed method has a better performance when the number of samples in the target domain is small (<200);

4) Although the GLG method is worse than the proposed method when n_2 is small, the performance of the GLG method improves when n_2 increases;

5) When n_2 increases, the performance of the proposed method approaches the performance of the GLG method.

Table 7.1 shows the overall prediction performance of each method (averaging the mean accuracy of each method when n_2 is 40, 100, 200, 300, 400). It is apparent that the proposed method is better than the baselines. The proposed method needs much less running time than the GLG method: the F-HeUDA takes 16 seconds to label 400 samples of the target domain while the GLG method needs 107 seconds to finish the same work with the proposed method, which

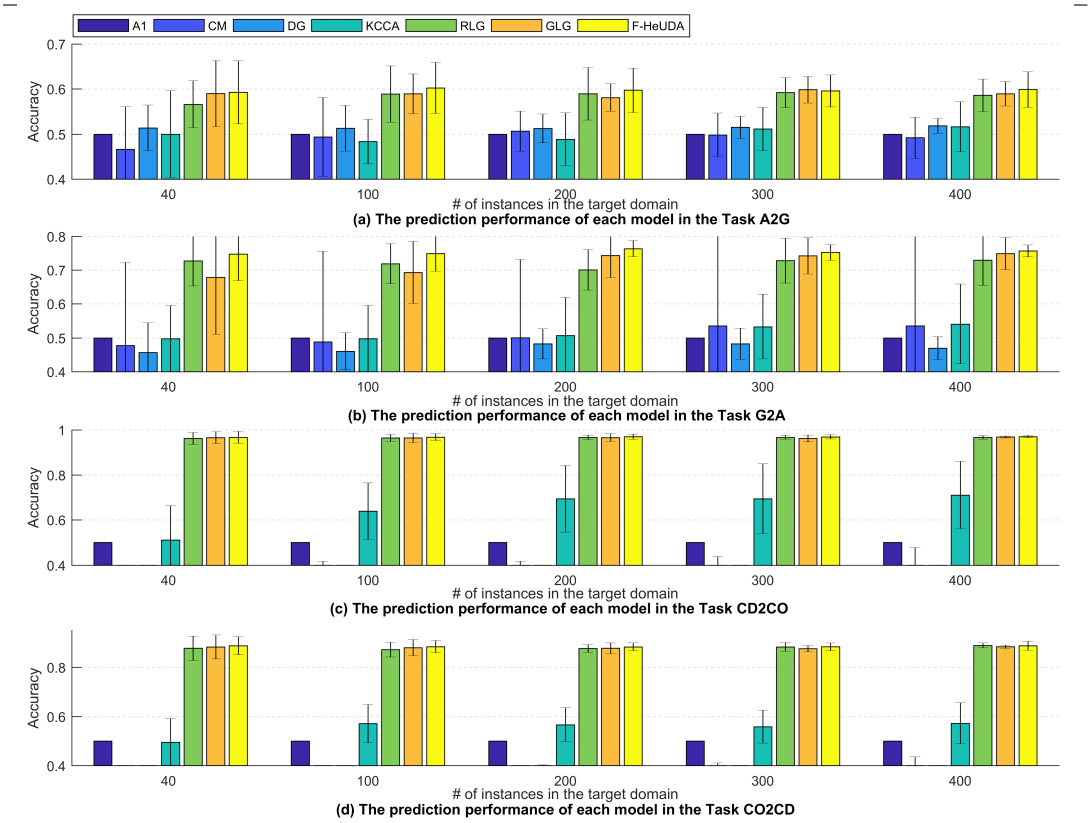


Figure 7.3: The performance of each method (mean accuracy and standard deviation) on 4 tasks. In each subfigure, the minimum mean accuracy is set as 0.4.

means that the proposed method is still a potential choice even when n_2 is a large number. Except for accuracy, this section also discusses the stability of each method using standard deviation. The mean overall standard deviation of each method is listed in Table 7.2. From Table 7.1 and Table 7.2, we can obtain following results.

1) The F-HeUDA method is the most stable method for three tasks: G2A, CD2CO and CO2CD;

2) For tasks CD2CO and CO2CD, RLG, GLG and F-HeUDA methods have much better performance in terms of stability than the other methods;

Table 7.2: The overall STD value of each method on four tasks.

Tasks	A1	CM	DG	KCCA	RLG	GLG	F-HeUDA
A2G	-	6.43%	3.49%	6.17%	4.83%	4.07%	4.99%
G2A	-	25.78%	5.37%	10.45%	6.70%	8.52%	3.90%
CD2CO	-	20.29%	12.12%	14.65%	1.37%	1.65%	1.33%
CO2CD	-	15.76%	8.65%	7.88%	2.45%	2.46%	2.23%

3) The KCCA method is the most unstable method among DG, KCCA, RLG, GLG and F-HeUDA, which means that the KCCA method experiences difficulty in correctly transferring knowledge from the source domain to the target domain;

4) When n_2 is small, the GLG method becomes unstable, mainly because the GLG method only uses a few samples of the source domain (the transferable knowledge of the GLG method is limited by n_2);

5) Although the RLG method uses all samples of the source domain, its random map nature limits its ability to obtain good feature representation.

7.5.3 Convergence of Learning Algorithm

This section discusses the convergence of the learning algorithm for the parameters of the SFER. Figure 7.4 illustrates the convergence of Algorithm 7.1 on four tasks when the target domain has 400 samples. It is apparent that the proposed algorithm can effectively optimize the parameters of SFER. From subfigures Figure 7.4e and 7.4h, we can see that the two domains have almost the same r_i after optimizing SFER, which means that two domains can share almost every $\alpha \in [0,\ 1]$ (like Figure 7.2b). For a different task, Algorithm 7.1 has a different performance. For example, for 1000 iterations, the G2A and A2G tasks have fewer errors than the CD2CO and CO2CD tasks, mainly because the divergence of the number of features of the CD and CO datasets is greater than that of the

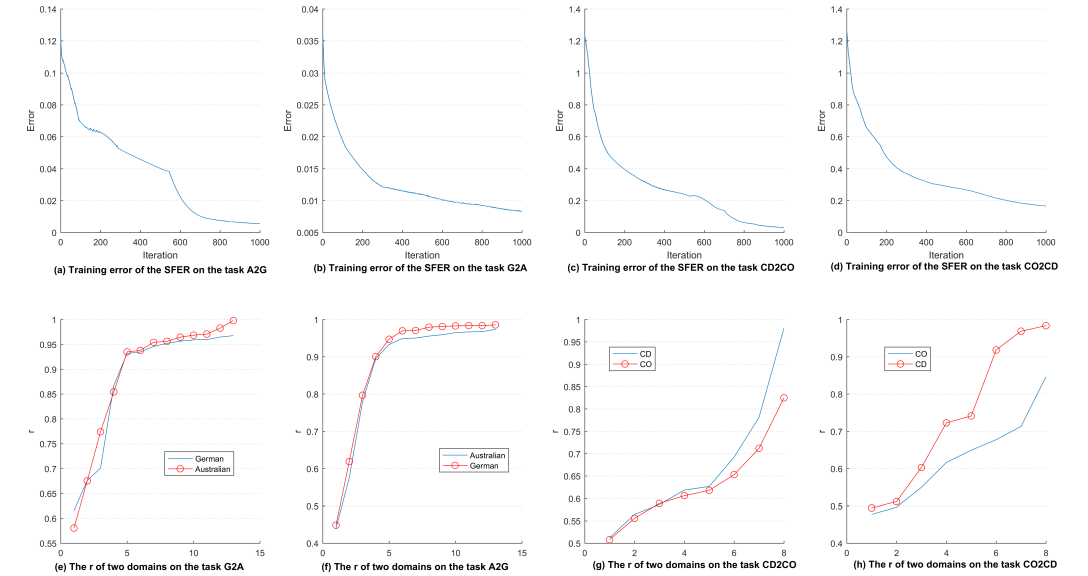


Figure 7.4: The convergence of Algorithm 7.1 on four tasks. Subfigures (a)-(d) illustrate the value of the cost function J_1 and subfigures (e)-(f) illustrate the r_i of R_{TD}^M of the source domain and the target domain.

German and Australian datasets. This indicates that for a high divergence task (divergence of the number of features of two domains), the IterM should be set as a larger number to ensure the performance of Algorithm 7.1.

7.5.4 User-oriented Decision Making Pattern

When Algorithm 7.1 has optimized the parameters of the SFER, two domains can apply the same α to obtain the same number of clusters, which provides a way to transfer knowledge from the source domain to the target domain. In Section 7.4.4, we propose to adopt the α which is in the largest interval as the best α because two domains can share the most information in the largest interval. However, the selection of α is actually a decision-making issue (different users may select different α based on their requirements), so it is necessary to discuss how the

Table 7.3: The prediction performance of F-HeUDA When chaning α.

Tasks	$\alpha = 0.1$	$\alpha = 0.3$	$\alpha = 0.5$	$\alpha = 0.7$	$\alpha = 0.9$
A2G	59.9%±4.2%	59.1%±3.7%	62.4%±4.5%	60.9%±4.7%	58.3%±6.0%
G2A	74.2%±5.6%	75.8%±4.8%	75.7%±4.2%	73.6%±9.2%	56.8%±15.3%
CD2CO	96.6%±1.5%	97.4%±1.3%	96.9%±1.5%	97.0%±1.4%	95.0%±2.7%
CO2CD	89.5%±3.0%	88.7%±2.7%	91.4%±2.6%	92.2%±3.8%	90.8%±3.8%

selection of α influences the performance, which provides another way to analyze the SFER method. In this section, we let the number of samples of the target domain be 100 and test how the prediction performance changes when α is set as 0.1, 0.3, 0.5, 0.7 and 0.9. Similarly, we demonstrate the results on four tasks and each experiment is carried out 20 times.

Table 7.3 gives the detailed prediction performance of F-HeUDA when α changes on each task. First, we analyze the relationship between the α selected by the F-HeUDA and the best α shown in Table 7.3. 1) For task A2G, the α selected by the F-HeUDA is around *0.6112* (after running the experiment 20 times) and Table 7.3 shows that the best α is around 0.5. 2) For task G2A, the F-HeUDA selects α as *0.2674* and Table 7.3 shows that the best α is around 0.3. 3) For task CD2CO, the α selected by the F-HeUDA is around *0.3729* and the best α shown in Table 7.3 is around 0.3. 4) For task CO2CD, the F-HeUDA selected *0.4689* as the α and Table 7.3 shows that the best α is around 0.7. Based on these results and Table 7.3, we can conclude the following:

1) Except for task CO2CD, the α selected by the F-HeUDA is near the best α as shown in Table 7.3;

2) When α lies in the interval [0.3 0.7], the F-HeUDA method obtains a better performance;

3) When α increases, the standard deviation of the proposed method is higher,

especially for task G2A.

These results demonstrate that the best α will not be near to extreme numbers (such as 0 or 1), which is consistent with normal decision-making scenarios (extreme decisions rarely occur). So, the F-HeUDA is a suitable method for decision making and its method of automatically selecting α is effective and accurate.

7.6 Summary

This chapter shows the potentiality of fuzzy technologies to address the HeUDA problem when two domains are imbalanced and outlines a new way (like the SFER) to obtain common representations of two different domains using fuzzy technologies. The SFER method is proposed to let two domains share the same α, and we can obtain the same number of clustering categories. Through these clustering categories, knowledge from the source domain is successfully transferred to the target domain where two domains have a different number of samples.

Compared to existing HeUDA methods, the proposed F-HeUDA method overcomes the drawbacks of CCA and the Grassmann manifold (both need two domains that have the same number of samples) and works well when the target domain has very few samples. This chapter proves two important properties of the fuzzy equivalence relations. Both properties are key to the performance of the SFER method. To validate the performance of the proposed method, we selected four real datasets to generate four transfer tasks. The prediction accuracy and stability of the proposed method are better than those of the baselines. These results show the superiority of the proposed method lies in transferring more knowledge from the source domain to the target domain.

CHAPTER 8

FUZZY-RELATION NEURAL NETWORKS: A MULTI-SOURCE HETEROGENEOUS UNSUPERVISED DOMAIN ADAPTATION APPROACH

8.1 Introduction

For existing *unsupervised domain adaptation* (UDA) methods, they assume that source data come from the single source domain (i.e., single-source scenario [Liu et al., 2018b; Long et al., 2019; Saito et al., 2017]) or from multiple source domains whose feature spaces have the same dimension (i.e., multi-homogeneous-source scenario [Duan et al., 2012c; Hoffman et al., 2018a; Zhang et al., 2015]). Namely, we can only use labeled data from single source domain or from multiple homogeneous source domains to train a classifier for unlabeled data in target domain.

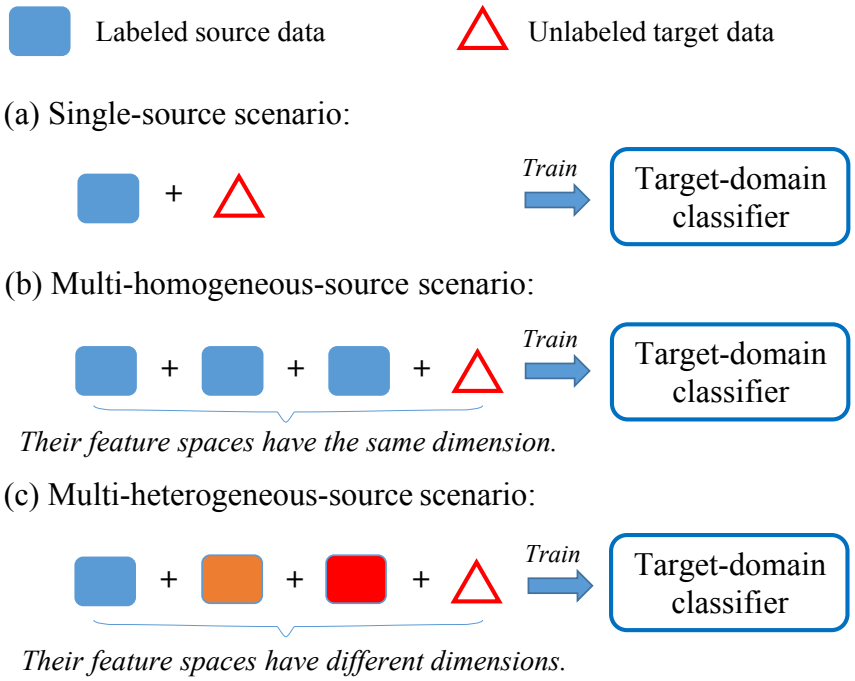


Figure 8.1: Three scenarios for the *unsupervised domain adaptation* (UDA) problem. Rectangles represent the labeled source data and triangle represents the unlabeled target data. Labeled source data may come from a) a single source domain (i.e., single-source scenario) or b) multiple source domains whose feature spaces have the same dimension (i.e., multi-homogeneous-source scenario) or c) multiple source domains whose feature spaces have different dimensions (i.e., multi-heterogeneous-source scenario). Given a target domain, since we probably have many different-dimension source domains (i.e., multi-heterogeneous-source scenario), the third scenario is more general than the other scenarios.

However, in real world, given a target domain, we probably have multiple different-dimension (*heterogeneous*) source domains, which does not satisfy the assumption of existing UDA methods. For example, assume that our task is to assess Japanese (J) credit using 15 features (a target domain only containing unlabeled data), where the meanings of these features are masked to protect private information. Furthermore, we have German (G) and Australian (A) credit assessment datasets (two source domains containing labeled data), where the

number of features in G and A are 24 and 14, respectively. Then, we want to train a classifier with data from G, A and J to assess Japanese credit (a UDA problem). However, existing UDA methods can only use knowledge from G or A rather than G and A. This results in a serious problem: *we can only use part of labeled data to help assess Japanese credit and we must abandon the other part.*

To remove the common assumption of existing UDA methods, we have two challenges: 1) how to extract shared information contained in multiple heterogeneous domains and 2) how to transfer knowledge across domains via this shared information. However, existing techniques (such as the four aforementioned techniques) can only extract shared information contained in two heterogeneous domains [Liu et al., 2018b] or multiple homogeneous domains [Zhang et al., 2015]. Hence, existing UDA methods does not even solve the first challenge.

To address both challenges and move towards to a realistic problem (i.e., UDA in multi-heterogeneous-source scenario), this chapter presents a *shared-fuzzy-equivalence-relations neural network* (SFERNN) to address *multi-source heterogeneous unsupervised domain adaptation* (MsHeUDA) problem. In the MsHeUDA problem, we have c different-dimension (*heterogeneous*) source domains and one target domain, where the target domain only contains unlabeled data. SFERNN, as a solution for the MsHeUDA problem, is a classifier for data in target domain and is trained with labeled data in these heterogeneous source domains and unlabeled data in target domain.

SFERNN is a five-layer neural network containing c source branches and one target branch. The network structure (i.e., number of nodes in each layer and how to connect two adjacent layers) of SFERNN is confirmed by a proposed fuzzy equivalence relation method called multi-source shared fuzzy equivalence

relations. The loss function of SFERNN comprises two parts. The first part represents supervised loss on labeled data from c source domains (cross-entropy loss in this chapter). The second part represents distributional discrepancy between source domains and target domain. In this chapter, distributional discrepancy between each source domain and target domain is calculated using *maximum mean discrepancy* (MMD) between the outputs of representation layer of each source branch and target branch. By minimizing this loss function, we can obtain the optimized parameters of SFERNN.

The reason why SFERNN is effective in addressing the MsHeUDA problem has its roots in *the proposed fuzzy equivalent relations*. The proposed fuzzy equivalent relations can extract shared fuzzy information contained in heterogeneous domains, which successfully solves the first challenge. Then, through this shared fuzzy information, we can construct latent features among $c+1$ heterogeneous domains. By using the constructed latent features to replace with the original features, the distributional discrepancy among these $c+1$ heterogeneous domains will vanish. Finally, knowledge from c source domains can be transferred to target domain via these latent features. Namely, we solve the second challenge.

To validate the efficacy of SFERNN, three real-world datasets are used to construct three MsHeUDA tasks. The experimental results reveal that SFERNN can reliably transfer knowledge from multiple heterogeneous domains to the unlabeled target domain. The main contributions of this chapter are as follows.

1) A novel problem setting, MsHeUDA, is proposed in this chapter. Compared to existing problem settings in the UDA field, MsHeUDA is a more realistic and difficult problem and cannot be solved using existing UDA methods.

2) SFERNN is proposed to solve this novel problem. SFERNN is based on a

novel fuzzy equivalence relation, called *multi-source shared fuzzy equivalence relations* (MsSFER). Compared to traditional fuzzy equivalence relations and shared fuzzy equivalence relations, MsSFER can extract shared fuzzy information contained in multiple heterogeneous domains. Via using this shared fuzzy information, we can construct latent features among $c+1$ heterogeneous domains.

3) Compared to existing single-source *heterogeneous UDA* (HeUDA) methods, no matter which source domain these HeUDA methods select as their "single source", SFERNN performs better than them on three real-world MsHeUDA tasks in terms of classification accuracy.

4) It is the first time that we have the ability to extract the shared fuzzy information of multiple heterogeneous domains. Previous methods can only extract the shared information of multiply homogeneous domains or two heterogeneous domains.

8.2 Concepts and Problem Setting

In this chapter, we will use three concepts: *n-dimensional fuzzy geometry* (*n-D* FG), fuzzy equivalence relations, similarity between fuzzy vectors and multi-source unsupervised domain adaptation. Since fuzzy geometry has been introduced in Section 7.2.1, this section will introduce the other in the following.

8.2.1 Similarity between Fuzzy Vectors

In Chapter 7, a new metric between two *n-D* fuzzy vectors was proposed to construct the similarity between two fuzzy vectors. We briefly restate the metric

and the similarity in this subsection. First, the detailed expression of a fuzzy vector $\bar{A}_i(a_{i1}, a_{i2}, ..., a_{in})$ (with the triangular membership function) is given as follows: for each $\bar{a}_{ij} \in F(\mathbb{R})$, its membership function is

$$\mu_{ij}(x|\bar{a}_{ij}) = \begin{cases} 0, & \forall x < a_{ij} - \rho_i \\ 1 - \frac{|x - a_{ij}|}{\rho_i}, & \forall |x - a_{ij}| \leq \rho_i \\ 0, & \forall x > a_{ij} + \rho_i \end{cases}, x \in \mathbb{R}, \tag{8.1}$$

Based on the $\mu_{ij}(x|\bar{a}_{ij})$, $\mu_i(x|\bar{A}_i)$ is expressed by the following term.

$$\mu_i(x|\bar{A}_i) = \begin{cases} 0, & \exists x_j, x_j < a_{ij} - \rho_i \\ 1 - \frac{\|x - a_{ij}\|_1}{n\rho_i}, & \forall x_j, |x_j - a_{ij}| \leq \rho_i \\ 0, & \exists x_j, x_j > a_{ij} + \rho_i \end{cases}, \tag{8.2}$$

where $x = (x_1, x_2, ..., x_n) \in \mathbb{R}^n$ and $\rho_i > 0$. Then, we define a new metric to measure the distance between two fuzzy vectors.

Definition 8.1. *Given two fuzzy vectors $\bar{A}_i \in F(\mathbb{R}^n)$ and $\bar{A}_j \in F(\mathbb{R}^n)$, the metric between $\bar{A}_i$ and $\bar{A}_j$ is defined by the map $\mathcal{D} : F(\mathbb{R}^n) \times F(\mathbb{R}^n) \to [0, +\infty)$:*

$$\mathcal{D}(\bar{A}_i, \bar{A}_j) = \frac{1}{n} \int_0^1 sup\{\mathcal{D}_\lambda(u, v) : \mathcal{D}_\lambda(u, v) \in \Omega(\lambda)\} d\lambda,$$

where

$$\Omega(\lambda) = \{d(u, \bar{A}_j(\lambda))\} \cup \{d(v, \bar{A}_i(\lambda))\},$$

$u \in \bar{A}_i(\lambda), v \in \bar{A}_j(\lambda)$ and the first part of $\Omega(\lambda)$ collects L_1 distances between each u and $\bar{A}_j(\lambda)$ ($d(u, \bar{A}_j(\lambda)) = \min\{d(u, v), v \in \bar{A}_j(\lambda)\}$ means the minimum L_1 distances between u and all elements in $\bar{A}_j(\lambda)$), and the second part of $\Omega(\lambda)$ collects L_1 distances between v and $\bar{A}_i(\lambda)$ ($d(v, \bar{A}_i(\lambda)) = min\{d(v, u), u \in \bar{A}_i(\lambda)\}$ means

the minimum L_1 distances between v and all elements in $\bar{A}_i(\lambda)$), and $d(u,v)$ represents the L_1 distance (ℓ_1-norm) between two n-dimension vector (u and v).

Then, the following equation is obtained to calculate $\mathcal{D}$ (based on Chapter 7).

$$\mathcal{D}(\bar{A}_i,\bar{A}_j)=\frac{1}{n}d(A_i,A_j)+\frac{1}{4}|\rho_i-\rho_j|. \tag{8.3}$$

In Chapter 7, we proved that $(F(\mathbb{R}^n),\mathcal{D})$ is a metric space [Liu et al., 2018b], which gives a new perspective on the measurement of two fuzzy vectors. Based on the metric $\mathcal{D}$, similarity between two fuzzy vectors is expressed as follows.

$$R_{\mathcal{D}}(\bar{A}_i,\bar{A}_j)=e^{-\frac{\mathcal{D}(\bar{A}_i,\bar{A}_j)}{2\sigma^2}}. \tag{8.4}$$

Using a fixed σ, similarity between $\bar{A}_i$ and $\bar{A}_j$ increases when $\mathcal{D}(\bar{A}_i,\bar{A}_j)$ decreases.

8.2.2 Fuzzy Equivalence Relations and Partitioning of Fuzzy Sets

Fuzzy equivalence relations were studied as a way to measure the similarity among fuzzy sets [Zadeh, 1971]. Since fuzzy equivalent relation have been introduced in Section 7.2.2, this subsection only reviews how to use the fuzzy equivalence relations to partition fuzzy sets.

With the constructed fuzzy equivalence relations, we can use α-cut of $\widetilde{R}^M$ to cluster $\bar{A}_1,\bar{A}_2,...,\bar{A}_N$. Specifically, the matrix of α-cut of $\widetilde{R}^M$ can be expressed by the following term.

$$\left(\widetilde{R}^M(\alpha)\right)_{ij}=\begin{cases}1, & \text{if } (\widetilde{R}^M)_{ij}\geq\alpha\\ 0, & \text{if } (\widetilde{R}^M)_{ij}\leq\alpha\end{cases}, \tag{8.5}$$

where $\widetilde{R}^M(\alpha)$ is a binary fuzzy equivalence relation matrix. Fuzzy sets that have the same corresponding rows of $\widetilde{R}^M(\alpha)$ can be regarded as the same cluster. The selection of α is a decision-making process, and users can choose α based on their own requirements.

Traditional fuzzy equivalence relations are only for one type of fuzzy set. For example, a set $\bar{S} = \{\bar{A}_1, \bar{A}_2, ..., \bar{A}_N\}$ collects N elements from $F(\mathbb{R}^n)$, then N elements of $\bar{S}$ can be partitioned into three categories (e.g., $\{\bar{A}_1\}$, $\{\bar{A}_2, \bar{A}_3\}$, $\{\bar{A}_4, ..., \bar{A}_N\}$) using $\widetilde{R}^M(\alpha)$ and a proper α, where $\bar{A}_i \in F(\mathbb{R}^n)$, $i = 1, 2, ..., N$. However, for the multi-source HeUDA problem, there are many sets like $\bar{S}$. For example, there could be c sets: $\bar{S}_i = \{\bar{A}_{i1}, \bar{A}_{i2}, ..., \bar{A}_{iN_i}\}$, where $\bar{A}_{ij} \in F(\mathbb{R}^{n_i})$ and $i = 1, 2, ..., c$, $j = 1, 2, ..., N_i$. Under the heterogeneous setting, $N_i \neq N_k$, $n_i \neq n_k$ if $i \neq k$. Clearly, $\widetilde{R}^M$ of these sets ($\bar{S}_i$) are different and cannot get the same number of categories through sharing the same α. This is the main obstacle when dealing with the multi-source HeUDA problem using fuzzy relations. In Chapter 7, a shared fuzzy equivalence relation (SFER) is designed to solve the single-source HeUDA problem but not for the multi-source HeUDA problem. In the next section, we propose a multi-source SFER to cluster features in each domain (including source domains and target domain) as N_l categories.

8.2.3 Multi-source Unsupervised Domain Adaptation

In this subsection, we give a formal definition of the multi-source unsupervised domain adaptation problem.

Definition 8.2 (Multi-Source Unsupervised Domain Adaptation)**.** *Let 1) X_t be*

a multivariate random variable defined on $\mathbb{R}^{N_t}$ *with respective a probability measure* P_{X_t} *and 2)* (X_i, Y_i) *be a multivariate random variable defined on* $\mathbb{R}^{N_i} \times \mathbb{C}$ *with respective a probability measure* P_i *and 3)* P_{X_i} *be probability measure of* X_i *(marginal probability measure of* P_i*), where* $P_{X_t} \neq P_{X_i}$, $\mathbb{C} = \{-1, +1\}$ *and* $i = 1, \ldots, c$. *Given i.i.d. data* $D_i = \{(x_{ij}, y_{ij})\}_{j=1}^{n_i}$ *and* $D_t = \{x_{tj}\}_{j=1}^{n_t}$ *drawn from* P_i *and* P_{X_t}, *a multi-source unsupervised domain adaptation aims to train with* $\{D_i\}_{i=1}^{c}$ *and* D_t *to accurately annotate data drawn from* P_{x_t}.

In a homogeneous setting, i.e., feature spaces of all domains have the same dimension, we have $N_1 = N_2 = \cdots = N_c = N_t$. In a heterogeneous setting, i.e., feature spaces of all domains have different dimensions, we have that $N_i = N_k$ if and only if $i = k$ and $N_i \neq N_t$. For a specific target domain, we probably have multiple different-dimension (*heterogeneous*) source domains in real world. Thus, this chapter focuses on multi-source heterogeneous unsupervised domain adaptation.

8.3 Shared Fuzzy Equivalence Relations for Multi-source Domains

We assume there are c feature sets of c source domains: $S_i = \{A_{i1}, A_{i2}, ..., A_{iN_i}\}$, where $A_{ij} \in \mathbb{R}^{n_i}$ and $i = 1, 2, ..., c$, $j = 1, 2, ..., N_i$, and one feature set of a target domain: $S_t = \{A_{t1}, A_{t2}, ..., A_{tN_t}\}$, where $A_{tj} \in \mathbb{R}^{n_t}$ and $j = 1, 2, ..., N_t$. Under the heterogeneous setting, $\forall i, k \in \{1, 2, ..., c\}$, $n_i = n_k$ and $N_i = N_k$ if and only if $i = k$, and $\forall i \in \{1, 2, ..., c\}$, $n_t \neq n_i$ and $N_t \neq N_i$. Multi-source SFER aims to find a latent feature set $S_l = \{A_{l1}, A_{l2}, ..., A_{lN_l}\}, A_{lN_l} \in \mathbb{R}^{n_l}, n_l = \sum_{i=1}^{c} n_i + n_t$, and use the

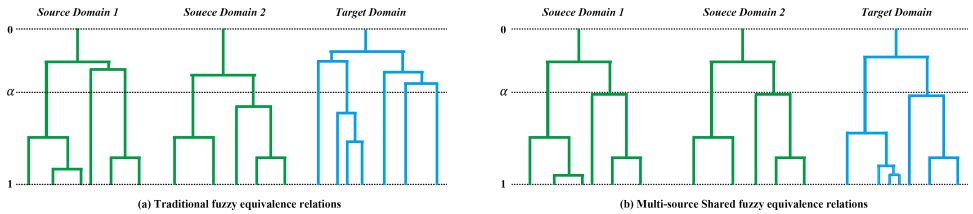


Figure 8.2: Target domain has 7 features while *source domain 1* has 6 and *source domain 2* has 5. Given the same α, traditional fuzzy equivalence relations can only simultaneously cluster 1) features in *source domain 1* as 3 categories, and 2) features in *source domain 2* as 2 categories, and 3) features in target domain as 5 categories. However, multi-source shared fuzzy equivalence relations can simultaneously cluster features in *source domain 1*, *source domain 2* and target domain as 2 categories.

labeled information of S_l (knowledge from source domains) to label the unlabeled information of S_l (unlabeled target data), where n_i represents the number of data in the i^{th} source domain, n_t represents the number of data in target domain, and N_l is the number of features in the latent feature set S_l.

Specifically, we want to simultaneously transform all of S_i and S_t to S_l. However, traditional fuzzy equivalence relations can only do this separately and SFER can only do this between two domains (e.g., S_1 and S_t). Thus, an important work of this chapter is to determine how to apply fuzzy equivalence relations to simultaneously transform all of S_i and S_t to S_l. To provide more details, we propose multi-source SFER (MsSFER) to let $\widetilde{R}^M$ of each S_i and S_t share the same α, which guarantees that all of S_i and $\bar{S}_t$ can be clustered as N_l categories with the same α. Figure 8.2 illustrates the difference between traditional fuzzy equivalence relations and MsSFER.

The first subsection is the theoretical guarantee for MsSFER, which is formally demonstrated in the second subsection. The third subsection proposes an algorithm to learn the parameters of MsSFER.

8.3.1 Theoretical Guarantees for Multi-source SFER

This subsection provides a theorem to show that we always find an α to let all of S_i and S_t are clustered as N_l categories ($N_l > 1$) by changing parameters of the triangular membership function, which is a theoretical foundation for MsSFER. In this chapter, we apply triangular membership function to convert a real-value vector to a fuzzy vector, shown in Eq. (8.2). Source feature set S_i is converted to $\bar{S}_i = \{\bar{A}_{i1}, \bar{A}_{i2}, ..., \bar{A}_{iN_i}\}$ using a parameter set $P_i = \{\rho_{i1}, \rho_{i2}, ..., \rho_{i,N_i}\}$, where ρ_{ij} is a parameter of triangular membership function of $\bar{A}_{ij}$, $i = 1, 2, ..., c$, $j = 1, 2, ..., N_i$. The target feature set S_t is converted to $\bar{S}_t = \{\bar{A}_{t1}, \bar{A}_{t2}, ..., \bar{A}_{tN_t}\}$ using a parameter set $P_t = \{\rho_{t1}, \rho_{t2}, ..., \rho_{t,N_t}\}$, where ρ_{tj} is a parameter of triangular membership function of $\bar{A}_{tj}$, $j = 1, 2, ..., N_t$. $R_{\mathcal{D}}$ is used to calculate a fuzzy relation between two fuzzy vectors. First, we restate a theorem to support our main theorem.

Theorem 8.1 ([Liu et al., 2018b]). *Given a set $\bar{S}_1 = \{\bar{A}_{11}, \bar{A}_{12}, ..., \bar{A}_{1N_1}\}$, if the fuzzy relation matrix $R^M_{\mathcal{D}}$ of $\bar{S}_1$ has $N_1(N_1 - 1)/2 + 1$ different elements and $\alpha \in (r_{1,j-1}, r_{1,j}]$, then $\bar{S}_1$ will be clustered as j categories, where $\{r_{1j}, j = 1, 2, ..., N_1\}$ is the set of N_1 different elements of $\widetilde{R}^M_{\mathcal{D}}$ of $\bar{S}_1$ and $r_{11} < r_{12} < ... < r_{1,N_1-1} < r_{1,N_1} = 1$.*

Proof. The proof is based on two facts:

1) there are only N_1 different elements in $\widetilde{R}^M_{\mathcal{D}}$ of $\bar{S}_1$ and $r_{11} < r_{12} < ... < r_{1,N_1-1} < r_{1,N_1} = 1$;

2) $rank(\widetilde{R}^M_{\mathcal{D}}(r_{1,j})) = rank(\widetilde{R}^M_{\mathcal{D}}(r_{1,j-1}) + 1)$, $j = 2, ...N_1$.

Both facts and the whole proof of this theorem are presented in Theorem 7.3 in Chapter 7. ■

Theorem 8.1 demonstrates that $\bar{S}_1$ is clustered to j categories by selecting an

$\alpha \in (r_{1,j-1}, r_{1,j}]$ (we denote the selected α by α_1). However, for a different set, such as $\bar{S}_2$, $r_{1,j}$ is probably not equal to $r_{2,j}$. If $r_{2,j} > r_{1,j}$ and $r_{2,j} - r_{1,j} > r_{2,j} - r_{2,j-1}$, α_1 will belong to the interval $(r_{2,j-2}, r_{2,j-1}]$, which means that $\bar{S}_2$ will be clustered as $j-1$ categories using α_1. It is obvious that $\bar{S}_1$ and $\bar{S}_2$ cannot share the α_1, which is the main drawback of traditional fuzzy equivalence relation.

A main purpose of MsSFER is that, for almost $\alpha \in (0,1)$, $\bar{S}_i$ $(i = 1,2,...,c)$ shares the same α with $\bar{S}_t$, i.e., $\bar{S}_i$ and $\bar{S}_t$ have the same number of clustered categories using the same α. To prove that this purpose can be achieved, the following theorem is provided to show that, for a specific number, such as $1 < N_l \leq \min\{N_i, N_t\}$, we always find an α to ensure $\bar{S}_i$ and $\bar{S}_t$ have the N_l clustered categories by changing parameters of the triangular membership functions.

Theorem 8.2. *Given two sets:* $\bar{S}_i = \{\bar{A}_{i1}, \bar{A}_{i2}, ..., \bar{A}_{iN_i}\}$, $\bar{S}_t = \{\bar{A}_{t1}, \bar{A}_{t2}, ..., \bar{A}_{tN_t}\}$ *and* N_l *(*$1 < N_l \leq \min\{N_i, N_t\}$*),* $\bar{S}_i$ *has* N_i *parameters with respective to triangular membership functions:* $P_i = \{\rho_{i1}, \rho_{i2}, ..., \rho_{i,N_i}\}$ *and* $\bar{S}_t$ *has* N_t *parameters:* $P_t = \{\rho_{t1}, \rho_{t2}, ..., \rho_{t,N_t}\}$. *If the fuzzy relation matrix* $R^M_{\mathcal{D}}$ *of* $\bar{S}_i$ *has* $N_i(N_i-1)/2+1$ *different elements and the fuzzy relation matrix* $R^M_{\mathcal{D}}$ *of* $\bar{S}_t$ *has* $N_t(N_t-1)/2+1$ *different elements, then there always an* α *to let* $\bar{S}_i$ *and* $\bar{S}_t$ *be clustered to* N_l *categories by changing elements of two parameter sets* P_i *and* P_t*, respectively.*

Proof. Based on Theorem 8.1, $\widetilde{R}^M_{\mathcal{D}}$ of $\bar{S}_i$ only has N_i different elements: $r_{i,1} < r_{i,2} < ... < r_{i,N_i}$ and $\widetilde{R}^M_{\mathcal{D}}$ of $\bar{S}_t$ only has N_t different elements: $r_{t,1} < r_{t,2} < ... < r_{t,N_t}$. First, we assume that there is an α to cluster $\bar{S}_t$ to N_l categories, which means that the selected $\alpha \in (r_{t,N_l-1}, r_{t,N_l})$.

Hence, we need to validate whether r_{i,N_l-1} can approach to r_{t,N_l-1} and r_{i,N_l} can approach to r_{t,N_l} by changing elements of P_i and P_t. Because r_{t,N_l-1} is a

similarity between two fuzzy vectors, without loss of the generality, we express r_{t,N_l-1} using the following equation.

$$r_{t,N_l-1} = exp\Big(-\frac{\mathcal{D}\Big(\bar{A}_{t,k_0}, \bar{A}_{t,k'_0}\Big)}{2\sigma^2}\Big), \tag{8.6}$$

where $0 < k_0, k'_0 < N_t$ and $\mathcal{D}$ is defined in Eq. (8.3). Recall Eq. (8.3), we know that

$$\mathcal{D}(\bar{A}_{t,k_0}, \bar{A}_{t,k'_0}) = \frac{1}{n_t} d(A_{t,k_0}, \bar{A}_{t,k'_0}) + \frac{1}{4}|\rho_{t,k_0} - \rho_{t,k'_0}|.$$

So, r_{t,N_l-1} can be changed by changing values of ρ_{t,k_0} and ρ_{t,k'_0} that are elements in P_t. Next, we express r_{i,N_l-1} using the following equation.

$$r_{i,N_l-1} = exp\Big(-\frac{\mathcal{D}\Big(\bar{A}_{i,l_0}, \bar{A}_{i,l'_0}\Big)}{2\sigma^2}\Big), \tag{8.7}$$

where

$$\mathcal{D}(\bar{A}_{i,l_0}, \bar{A}_{i,l'_0}) = \frac{1}{n_i} d(A_{i,l_0}, \bar{A}_{i,l'_0}) + \frac{1}{4}|\rho_{i,l_0} - \rho_{i,l'_0}|.$$

Since ρ_{t,k_0}, ρ_{t,k'_0}, ρ_{i,l_0} and ρ_{i,l'_0} can be any positive real numbers, it is clear that $\mathcal{D}(\bar{A}_{i,l_0}, \bar{A}_{i,l'_0})$ can approach $\mathcal{D}(\bar{A}_{t,k_0}, \bar{A}_{t,k'_0})$ by changing values of ρ_{t,k_0}, ρ_{t,k'_0}, ρ_{i,l_0} and ρ_{i,l'_0} that are elements in P_t and P_i. That is, r_{i,N_l-1} can approach to r_{t,N_l-1} and r_{i,N_l} can approach to r_{t,N_l} by changing elements of P_i and P_t. ■

Theorem 8.2 presents that, for a specific N_l, we can always find an α to let two fuzzy feature sets $\bar{S}_i$ and $\bar{S}_t$ be clustered to N_l categories by changing parameters (elements of P_i and P_t shown in Theorem 8.2) of triangular membership functions of two sets, respectively. This inspires us to optimize parameters of triangular membership functions of c source fuzzy feature sets and one target fuzzy feature set to let them share more α, which is a main purpose of MsSFER.

8.3.2 Multi-source SFER

Based on Theorem 8.1, a significant property for fuzzy equivalence relation is observed: the number of clusters of a source fuzzy feature set $\bar{S}_1$ is decided by the value of r_{1j}, which means that we can use r_{1j} of a source domain and r_{tj} of a target domain to represent how many clustered categories each of them has when an α is selected. Thus, r_{ij} and r_{tj} are applied to define a cost function of MsSFER as follows.

$$J(\{S_i,P_i\}_{i=1}^{c},S_t,P_t)=\sum_{i=1}^{c}\sum_{j=1}^{N_{\min}}\left(r_{ij}(S_i,P_i)-r_{tj}(S_t,P_t)\right)^2, \tag{8.8}$$

where $N_{\min}=\min\{N_1,N_2,...,N_c,N_t\}$, r_{ij} is an element of $\widetilde{R}_{\mathcal{D}}^{M}$ of $\bar{S}_i$ and r_{tj} is an element of $\widetilde{R}_{\mathcal{D}}^{M}$ of $\bar{S}_t$.

The cost function J represents the difference between r_{ij} and r_{tj}. Thus, the MsSFER aims to minimize the $J(\{S_i,P_i\}_{i=1}^{c},S_t,P_t)$ by tuning P_i and P_t, $i=1,2,...,c$, expressed by

$$\begin{aligned}&\min_{\{P_i\}_{i=1}^{c},P_t} J(\{S_i,P_i\}_{i=1}^{c},S_t,P_t)\\ &s.\,t.\quad P_i>0,\ P_t>0.\end{aligned} \tag{8.9}$$

If the cost function J is approaching 0, r_{ij} is almost same as r_{tj}, $i=1,2,...,c$, $j=1,2,...,N_{\min}$. Based on Theorem 8.1, all of S_i and S_t will have the same number of clustered categories when applying the same α to these $c+1$ domains.

8.3.3 Learning Process of Multi-source SFER

To learn P_i and P_t, we first consider, for a specific j_0, how to minimize $(r_{ij_0}-r_{tj_0})^2$, where $1\le J_0\le N_{\min}-1$ and $i=1,2,...,c$. Because of the nature of max-min

operator, r_{ij_0} equals one element in $R_{\mathcal{D}}^{M}$ of S_i and r_{tj_0} equals one element in $R_{\mathcal{D}}^{M}$ of S_t. Without loss of generality, we assume $r_{ij_0} = (R_{\mathcal{D}}^{M}(S_i))_{l_0,l_0'}$ and $r_{tj_0} = (R_{\mathcal{D}}^{M}(S_t))_{k_0,k_0'}$, which means

$$r_{ij_0} = exp\left(-\frac{\frac{1}{2}d\left(A_{i,l_0},A_{i,l_0'}\right)+\frac{1}{4}|\rho_{i,l_0}-\rho_{i,l_0'}|}{2\sigma^2}\right), \tag{8.10}$$

$$r_{tj_0} = exp\left(-\frac{\frac{1}{2}d\left(A_{t,k_0},A_{t,k_0'}\right)+\frac{1}{4}|\rho_{t,k_0}-\rho_{t,k_0'}|}{2\sigma^2}\right). \tag{8.11}$$

It is obvious that r_{ij_0} is related to $|\rho_{i,l_0}-\rho_{i,l_0'}|$, so the constrains of $J(\{S_i,P_i\}_{i=1}^{c},S_t,P_t)$ do not affect the value of J. Then, a new optimization problem (no constraint) related to P_i and P_t is defined as follows.

$$\min_{\{P_i\}_{i=1}^{c},P_t} J(\{S_i,P_i\}_{i=1}^{c},S_t,P_t). \tag{8.12}$$

Then, the gradients of $\rho_{i,l_0},\rho_{i,l_0'},\rho_{t,k_0}$, and $\rho_{t,k_0'}$ with respective to $(r_{ij_0}-r_{tj_0})^2$ are obtained:

$$\frac{\partial(r_{ij_0}-r_{tj_0})^2}{\partial\rho_{i,l_0}} = \frac{-2(r_{ij_0}-r_{tj_0})^2 r_{ij_0}|\rho_{i,l_0}-\rho_{i,l_0'}|}{2\sigma^2 sign(\rho_{i,l_0}-\rho_{i,l_0'})}, \tag{8.13}$$

$$\frac{\partial(r_{ij_0}-r_{tj_0})^2}{\partial\rho_{i,l_0'}} = \frac{2(r_{ij_0}-r_{tj_0})^2 r_{ij_0}|\rho_{i,l_0}-\rho_{i,l_0'}|}{2\sigma^2 sign(\rho_{i,l_0}-\rho_{i,l_0'})}, \tag{8.14}$$

$$\frac{\partial(r_{ij_0}-r_{tj_0})^2}{\partial\rho_{t,k_0}} = \frac{2(r_{ij_0}-r_{tj_0})^2 r_{tj_0}|\rho_{t,k_0}-\rho_{t,k_0'}|}{2\sigma^2 sign(\rho_{t,k_0}-\rho_{t,k_0'})}, \tag{8.15}$$

$$\frac{\partial(r_{ij_0}-r_{tj_0})^2}{\partial\rho_{t,k_0'}} = \frac{-2(r_{ij_0}-r_{tj_0})^2 r_{tj_0}|\rho_{t,k_0}-\rho_{t,k_0'}|}{2\sigma^2 sign(\rho_{t,k_0}-\rho_{t,k_0'})}. \tag{8.16}$$

Inspired by incremental gradient decent, for each r_{ij_0}, we optimize ρ_{i,l_0} and $\rho_{i,l_0'}$ once using following equations.

$$\rho_{i,l_0} = \rho_{i,l_0} - \eta \frac{\partial (r_{ij_0} - r_{tj_0})^2}{\partial \rho_{i,l_0}}, \tag{8.17}$$

$$\rho_{i,l_0'} = \rho_{i,l_0'} - \eta \frac{\partial (r_{ij_0} - r_{tj_0})^2}{\partial \rho_{i,l_0'}}. \tag{8.18}$$

Based on definition of the cost function J, we optimize ρ_{t,k_0} and $\rho_{t,k_0'}$ once using following equations.

$$\rho_{t,k_0} = \rho_{t,k_0} - \eta \sum_{i=1}^{c} \frac{\partial (r_{ij_0} - r_{tj_0})^2}{\partial \rho_{t,k_0}}, \tag{8.19}$$

$$\rho_{t,k_0'} = \rho_{t,k_0'} - \eta \sum_{i=1}^{c} \frac{\partial (r_{ij_0} - r_{tj_0})^2}{\partial \rho_{t,k_0'}}. \tag{8.20}$$

Hence, we optimize $\rho_{i,l_0}, \rho_{i,l_0'}, \rho_{t,k_0}$, and $\rho_{t,k_0'}$ once using r_{ij_0} and r_{tj_0} (no iteration). This means $\rho_{i,l_0}, \rho_{i,l_0'}, \rho_{t,k_0}$, and $\rho_{t,k_0'}$ can be optimized $N_{\min} - 1$ times using different r_{ij_0} and r_{tj_0}, where $1 \le j_0 \le N_{\min} - 1$. Using optimized $\rho_{i,l_0}, \rho_{i,l_0'}, \rho_{t,k_0}$, and $\rho_{t,k_0'}$, $R_{\mathcal{D}}^M$ of S_i and $R_{\mathcal{D}}^M$ of S_t will be updated. Then $\rho_{i,l_0}, \rho_{i,l_0'}, \rho_{t,k_0}$, and $\rho_{t,k_0'}$ will be optimized $N_{\min} - 1$ again. After limited iterations, the best $\rho_{i,l_0}, \rho_{i,l_0'}, \rho_{t,k_0}$, and $\rho_{t,k_0'}$ will be obtained. Following above procedures, Algorithm 8.1 is designed to optimize $\rho_{i,l_0}, \rho_{i,l_0'}, \rho_{t,k_0}$, and $\rho_{t,k_0'}$.

After obtaining optimized $\{P_i\}_{i=1}^c$ and P_t, for almost $\alpha \in (0,1)$, all of S_i and S_t will have the same number of clustered categories, e.g., N_l categories, after using the same α. This means that we may transfer knowledge from c source domains to one target domain through these $c+1$ clustered feature sets, where each clustered feature set has N_l features subsets. For example,

Algorithm 8.1 Learning process of MsSFER

1: Input: Source feature sets S_i, $i = 1, ..., c$, target feature set S_t and IterM.
2: Initialize parameter sets $\{P_i\}_{i=1}^{c}$ and P_t;
3: Generate a small positive real number ϵ;
for $iter = 1 : IterM$ **do**
  4: Calculate $\widetilde{R}_{\mathcal{D}}^{M}$ of S_i and S_t;
  5: Extract r_{ij_0} and r_{tj_0} from $\widetilde{R}_{\mathcal{D}}^{M}$ of S_i and S_t;
  for $j_0 = 1 : N_{\min} - 1$ **do**
    for $i = 1 : c$ **do**
      6: Find l_0 and $l_0^{'}$ such that $(R_{\mathcal{D}}^{M}(S_i))_{l_0,l_0^{'}} = r_{ij_0}$;
      7: Update $\rho_{i,l_0}, \rho_{i,l_0^{'}}$ using (8.17) and (8.18);
    end
    8: Find k_0 and $k_0^{'}$ such that $(R_{\mathcal{D}}^{M}(S_t))_{k_0,k_0^{'}} = r_{tj_0}$;
    9: Update $\rho_{t,k_0}, \rho_{t,k_0^{'}}$ using (8.19) and (8.20);
  end
end
for $i = 1 : c$ **do**
  10: Compute $P_i = P_i - \min\{\rho_{i1}, ..., \rho_{i,N_i}\} + \epsilon$; // Ensure $P_i > 0$.
end
11: Compute $P_t = P_t - \min\{\rho_{t1}, ..., \rho_{t,N_i}\} + \epsilon$; // Ensure $P_t > 0$.
12: Output: $\{P_i\}_{i=1}^{c}$, P_t.

$S_1 = \{A_{11}, A_{12}, A_{13}, A_{14}\}$ is one source feature set and is clustered to a new feature set: $S_{l1} = \{\{A_{11}, A_{12}\}, \{A_{13}, A_{14}\}\}$ and $S_t = \{A_{t1}, A_{t2}, A_{t3}, A_{t4}, A_{t5}\}$ is a target feature set and is clustered to a new feature set: $S_{lt} = \{\{A_{t1}, A_{t2}\}, \{A_{t3}, A_{t4}, A_{t5}\}\}$. In the clustered feature sets S_{l1} and S_{lt}, there are only two elements, i.e., two feature subsets. Next section will introduce how to convert N_l feature subsets of S_i and S_t to N_l features and transfer knowledge from c source domains to one target domain through these N_l features.

8.4 Shared Fuzzy Equivalence Relations Neural Network for Multi-source HeUDA

This section presents a novel neural network to transfer knowledge from labeled data in c source domains to a target domain, where any two domains are heterogeneous. Inputs of this neural network include labeled data from c source domains and a few unlabeled data from target domain, and outputs of this network are labels of data from target domain. Because the structure of this network is determined by results of MsSFER proposed in Section 8.3.2, this network is named as shared fuzzy equivalence relations neural network, SFERNN for short.

8.4.1 Structure of the Proposed Neural Network

This subsection presents structure of SFERNN, including the number of layers, the number of neurons of each layer, the types of activation functions between adjacent layers, whether adjacent layers are fully connected and constraints of weights of SFERNN. SFERNN is a five-layer neural network with $c+1$ branches, where c branches come from c source domains and the other branch comes from target domain. Figure 8.3 illustrates the structure of SFERNN for $c = 2$. In this section, we use $D_i = \{(x_{ij}, y_{ij})\}_{j=1}^{n_i}$ to represent the i^{th} source domain and its corresponding instances and $D_t = \{x_{tj}\}_{j=1}^{n_t}$ to represent unlabeled data from target domain, where $i = 1, ..., c$ $x_{ij} \in \mathbb{R}^{N_i \times 1}$, $x_{tj} \in \mathbb{R}^{N_t \times 1}$ and $y_{ij} \in \{-1, +1\}$.

Layer I (Input layer): the input layer receives $c+1$ instances where c instances come from the c source domains, e.g., $x_{11}, x_{21}, ..., x_{c1}$, and one instance comes from the target domain, e.g., x_{t1}. The input layer thus has $\sum_{i=1}^{c} N_i + N_t$

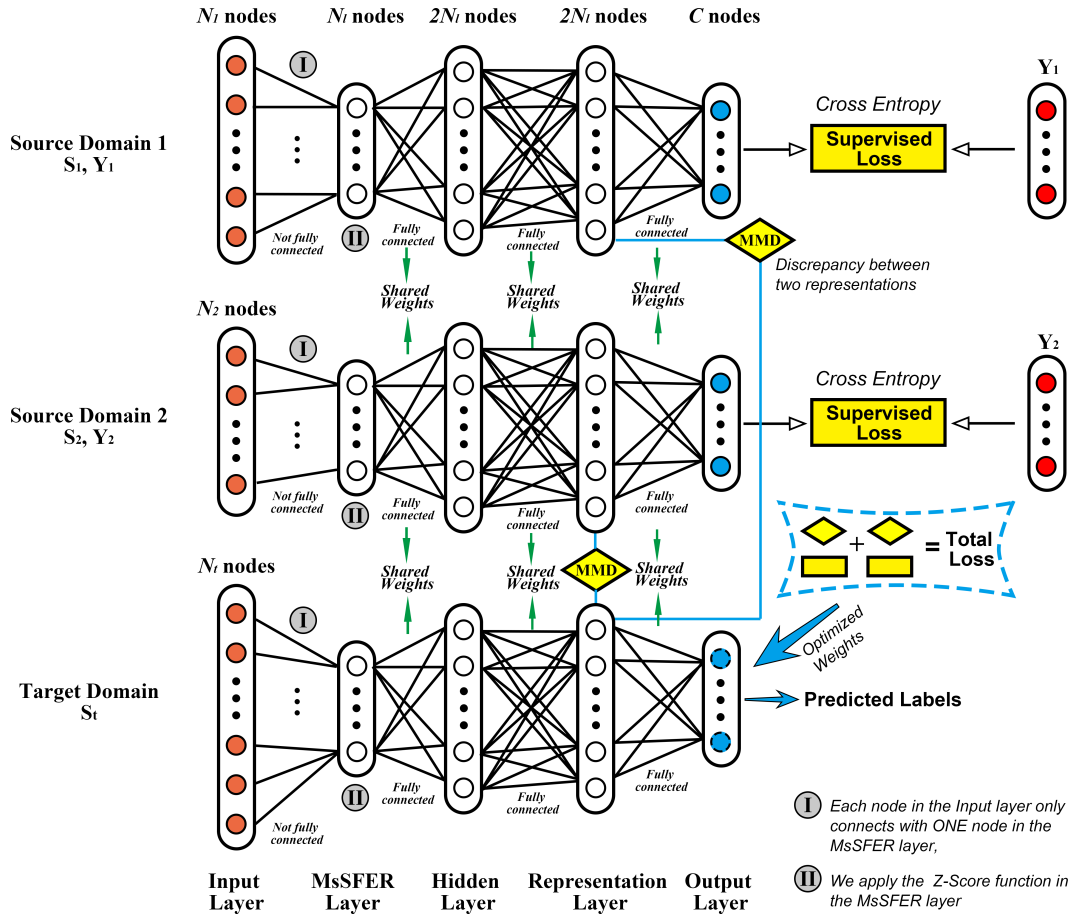


Figure 8.3: Network struture of SFERNN. SFERNN is a five-layer neural network containing c source branches and one target branch ($c = 2$ in this figure). Network structure (i.e., N_l in this figure and how to connect two adjacent layers) of SFERNN is confirmed by MsSFER. Loss function of SFERNN is composed of two parts. The first part represents cross-entropy loss on labeled data from c source domains. The second part represents distributional discrepancy (MMD) between source domains and target domain.

neurons. The i^{th} source-domain branch has N_i neurons and the target-domain branch has N_t neurons.

Layer II (MsSFER layer): the MsSFER layer has $(c+1)\times N_l$ neurons, where each branch has N_l neurons and N_l is confirmed by the MsSFER algorithm and a specific α. The values of neurons in the i^{th} source-domain branch forms a N_l-by-1

vector and is denoted by o_2^{ij} (corresponding to the j^{th} instance in the i^{th} source domain). Using a similar notation, $o_2^{tj} \in \mathbb{R}^{N_l \times 1}$ corresponds to values of neurons in the target-domain branch for the j^{th} instance in target domain. o_2^{ij} and o_2^{tj} are calculated by following equations.

$$o_2^{ij} = W_1^i x_{ij},\ o_2^{tj} = W_1^t x_{tj}, \tag{8.21}$$

where W_1^i is a N_l-by-N_i matrix and W_1^t is a N_l-by-N_t matrix. W_1^i and W_1^t have following constraints.

$$\sum_{p=1}^{N_l} 1_{(W_1^i)_{pq}=0} = N_l - 1,\ \sum_{q=1}^{N_i} 1_{(W_1^i)_{pq}=0} \neq 0, \tag{8.22}$$

and

$$\sum_{p=1}^{N_l} 1_{(W_1^t)_{pq}=0} = N_l - 1,\ \sum_{q=1}^{N_t} 1_{(W_1^t)_{pq}=0} \neq 0, \tag{8.23}$$

where $1_{(W_1^i)_{pq}=0} = 1$ if $(W_1^i)_{pq} = 0$, otherwise, $1_{(W_1^i)_{pq}=0} = 0$.

This means that the Input layer and the MsSFER layer are not fully connected in each branch, and each input neuron only connects with one neuron in the MsSFER layer. To confirm how to connect neurons between the Input layer and the MsSFER layer, c source feature sets $S_i = \{A_{i1}, A_{i2}, ..., A_{iN_i}\}$ and a target feature set $S_t = \{A_{t1}, A_{t2}, ..., A_{tN_t}\}$ are generated using D_i and D_t. Then, using the MsSFER and a specific α, N_i features in S_i are clustered to N_l subsets $A_{i1}^l, A_{i2}^l, ..., A_{iN_l}^l \subset S_i$, where $A_{ip}^l \cap A_{iq}^l = \emptyset$ if $p \neq q$. Let $S_{li} = \{A_{i1}^l, A_{i2}^l, ..., A_{iN_l}^l\}$. It is clear that each feature in S_i only belongs to one element in S_{li}, which means that, with the help of MsSFER, we confirm how to connect the Input layer (features in S_i) and MsSFER layer (elements in S_{li}).

Since each neuron in Input layer only connects with one neuron in MsSFER layer, there are only $\sum_{i=1}^{c} N_i + N_t$ parameters between two layers. We denote these parameters by $w_i^F = (w_{i1}^F, ..., w_{iN_i}^F)$ for the i^{th} source-domain branch and $w_t^F = (w_{t1}^F, ..., w_{tN_t}^F)$ for the target-domain branch, where $w_i^F \in \mathbb{R}^{1\times N_i}$ and $w_t^F \in \mathbb{R}^{1\times N_t}$. In Chapter 7, every element in w_i^F and w_t^F is set to 1. However, in this work, w_i^F and w_t^F are optimized under a constraint (see Section 8.4.3). After obtaining o_2^{ij} and o_2^{tj}, z-score function is applied to normalize o_2^{ij} and o_2^{tj}.

Layer III (Hidden layer): the Hidden layer has $2(c+1)\times N_l$ neurons, where each branch has $2N_l$ neurons. The values of neurons in the i^{th} source-domain branch forms a $2N_l$-by-1 vector and is denoted by o_3^{ij} (corresponding to the j^{th} instance in the i^{th} source domain). Using the similar notation, $o_3^{tj} \in \mathbb{R}^{N_l\times 1}$ corresponds to values of neurons in the target-domain branch for the j^{th} instance in the target domain. o_3^{ij} and o_3^{tj} are calculated by following equations.

$$o_3^{ij} = g(W_2 o_2^{ij} + b_2),\ o_3^{tj} = g(W_2 o_2^{tj} + b_2), \tag{8.24}$$

where W_2 is a $2N_l$-by-N_l matrix, b_2 is a $2N_l$-by-1 vector, o_2^{ij} and o_2^{tj} are values of neurons in the Layer II and $g(\cdot)$ is a rectifier linear unit (ReLU) function, $g(x) = \max(0, x)$. It is clear that these $c+1$ branches share the weight matrix W_2 and bias vector b_2.

Layer IV (Representation layer): the Representation layer has $2(c+1)\times N_l$ neurons, where each branch has $2N_l$ neurons. Similar with the Hidden layer, values of neurons in this layer, o_4^{ij} and o_4^{tj}, are calculated by following equations.

$$o_4^{ij} = g(W_3 o_3^{ij} + b_3),\ o_4^{tj} = g(W_3 o_3^{tj} + b_3), \tag{8.25}$$

where W_3 is a $2N_l$-by-$2N_l$ matrix, b_3 is a $2N_l$-by-1 vector, o_3^{ij} and o_3^{tj} are values

of neurons in the Layer III and $g(\cdot)$ is a ReLU function. These $c+1$ branches share the weight matrix W_3 and bias vector b_3.

Layer V (Output layer): the Output layer has Cc neurons, where each branch has C neurons. Similar with the Hidden layer, values of neurons in this layer, o_5^{ij} and o_5^{tj}, are calculated by following equations.

$$o_5^{ij} = W_4 o_4^{ij} + b_4,\ o_5^{tj} = W_4 o_4^{tj} + b_4, \tag{8.26}$$

where W_4 is a C-by-$2N_l$ matrix, b_4 is a C-by-1 vector and o_4^{ij} and o_4^{tj} are values of neurons in the Layer III. These $c+1$ branches share the weight matrix W_4 and bias vector b_4.

8.4.2 Loss Function of the Proposed Neural Network

This section presents the loss function of the SFERNN, which is a summation of c supervised loss (cross-entropy loss) and c regularizers (MMD). These supervised loss are to measure the disagreement between predicted labels and true labels in c source domains and these regularizers are to measure the discrepancy between outputs of representation layer of each source branch and that of the target branch. First, cross-entropy loss is selected as the supervised loss for each source branch, denoted as ℓ_i:

$$\ell_i(o_5^{ij}, y_{ij}) = -\sum_{\tau=1}^{C} 1_{y_{ij}=\tau} \log(h_\tau(x_{ij})),$$

where $h_\tau(x_{ij}) = (o_5^{ij})_\tau / \sum_{\tau'=1}^{C} (o_5^{ij})_\tau$ is the softmax function that computes the probability of predicting sample x_{ij} for the τ^{th} class [Long et al., 2016a]. Second, MMD between O_4^i and O_4^t (values of neurons in representation layers) is calculated by

the following equation:

$$\begin{aligned}\mathrm{MMD}_i^2(O_4^i, O_4^t) = &\frac{1}{n_i^2}\sum_{p,q=1}^{n_i} k(o_4^{ip}, o_4^{iq}) - \frac{2}{n_i n_t}\sum_{p,q=1}^{n_i,n_t} k(o_4^{ip}, o_4^{tq}) \\ &+ \frac{1}{n_t^2}\sum_{p,q=1}^{n_t} k(o_4^{tp}, o_4^{tq})),\end{aligned}$$

where $O_4^i = \{o_4^{ip}\}_{p=1}^{n_i}$ and $O_4^t = \{o_4^{tp}\}_{p=1}^{n_t}$. Minimizing $\mathrm{MMD}(O_4^i, O_4^t)$ will reduce the discrepancy between outputs of representation layers of i^{th} source-domain branch and the target-domain branch, which is beneficial for improving the classification accuracy on target domain [Long et al., 2015, 2017]. Thus, the loss function for the SFERNN is defined as follows.

$$\begin{aligned}&\ell_{total}(\{D_i\}_{i=1}^c, D_t; \{w_i^F\}_{i=1}^c, w_t^F, \{W_k, b_k\}_{k=2}^4) \\ &= \sum_{i=1}^{c}\sum_{j=1}^{n_i} \ell_i(o_5^{ij}, y_{ij}) + \sum_{i=1}^{c} \lambda_i \mathrm{MMD}_i(O_4^i, O_4^t).\end{aligned} \tag{8.27}$$

8.4.3 Learning Process of the Proposed Neural Network

This subsection presents how to optimize parameters of SFERNN to minimize the ℓ_{total} defined in Eq. (8.27). Different with normal optimization procedures of neural networks, two constraints are needed to consider to avoid the negative transfer when minimizing the ℓ_{total}, which is expressed as follows.

$$w_i^F > 0,\ w_t^F > 0,\ i = 1, ..., c. \tag{8.28}$$

This follows the results in [Liu et al., 2020b] which shows that avoiding negative transfer is a key to an HeUDA problem. Thus, optimized parameters of SFERNN are obtained by solving the following optimization problem.

$$\begin{aligned}&\textbf{min}\ \ell_{total}(\{D_i\}_{i=1}^c, D_t; \{w_i^F\}_{i=1}^c, w_t^F, \{W_k, b_k\}_{k=2}^4) \\ &s.t.\ w_i^F > 0,\ w_t^F > 0,\ i = 1, ..., c.\end{aligned}$$

Algorithm 8.2 Learning process of SFERNN

1: Input: Source data, target data: D_i,D_t and α.
2: Generate the source feature set S_i using D_i;
3: Generate the target feature set S_t using D_t;
4: Compute $[\{P_i\}_{i=1}^c, P_t]$ = MsSFER(S_i,S_t);
5: Use $\{P_i\}_{i=1}^c$, P_t and α to confirm positions of non-zero elements in W_1^i and W_1^t (i.e., w_i^F,w_t^F), $i = 1,...,c$;
for $iter = 0 : MaxIter$ **do**
  6: Assign $w_i^{tem} = w_i^F, i = 1,...,c$;
  7: Assign $w_t^{tem} = w_t^F$;
  8: Update W_k: $W_k = W_k - \eta\nabla_{W_k}\ell_{total}$, $k = 2,3,4$;
  9: Update b_k: $b_k = b_k - \eta\nabla_{b_k}\ell_{total}$, $k = 2,3,4$;
  10: Update w_i^F: $w_i^F = w_i^F - \eta\nabla_{w_i^F}\ell_{total}$, $i = 1,...,c$;
  11: Update w_t^F: $w_t^F = w_t^F - \eta\nabla_{w_t^F}\ell_{total}$;
  12: Find negative value in w_i^F: $\mathcal{I}_i$ = find($w_i^F < 0$), $i = 1,...,c$;
  13: Find negative value in w_t^F: $\mathcal{I}_t$ = find($w_t^F < 0$);
  14: Update w_i^F: $w_i^F[\mathcal{I}_i] = w_i^{tem}[\mathcal{I}_i]$, $i = 1,...,c$;
  15: Update w_i^F: $w_t^F[\mathcal{I}_t] = w_t^{tem}[\mathcal{I}_t]$;
end
16: Output: $\{W_k, b_k\}_{k=2}^4, w_i^F, w_t^F, i = 1,...,c$.

A stochastic gradient decent (SGD) algorithm is applied to solve the above problem and the pseudo code of the learning process is demonstrated in Algorithm 8.2. SFERNN is implemented by Pytorch 0.4.0.

Using the outputs of Algorithm 8.2, labels of instances $\{x_{tj}\}_{j=1}^{N_t}$ are predicted as

$$\hat{y}_{tj} = arg\max_{\tau}\{(o_5^{tj})_\tau\},$$

where $\tau = 1,...,C$ and $j = 1,...,N_t$.

8.5 Experiments

In this section, three real datasets are used to test the proposed SFERNN in terms of classification accuracy. We also analyze sensitivity of hyperparameters of SFERNN. All datasets are publicly available from the UCI Machine Learning Repository (UMLR).

8.5.1 Datasets and Tasks

The details of these datasets are provided in Table 8.1. For each dataset, we generate one multi-to-one transfer task (i.e., MsHeUDA tasks) and three one-to-one transfer tasks (i.e., HeUDA tasks). Each task is described in detail as follows:

1) Task 1 - G2A, J2A, GJ2A: Assume that German data and Japanese data are labeled, and the Australian data is unlabeled and has far fewer instances than the German and Japanese data. This task aims to answer two questions: a) “Can we use knowledge from German and Japanese credit records to classify unlabeled Australian data (small dataset)?”; and b) "is multi-source better than single-source for classifying unlabeled Australian data (small dataset)?"

2) Task 2 - A2G, J2G, AJ2G: Assume that Australian data and Japanese data are labeled, and the German data is unlabeled and has far fewer instances than the Australian and Japanese data. This task aims to answer two questions: a) “Can we use knowledge from Australian and Japanese credit records to classify unlabeled German data (small dataset)?”; and b) "is multi-source better than single-source for classifying unlabeled German data (small dataset)?"

3) Task 3 - A2J, G2J, AG2J: Assume that Australian data and German data

Table 8.1: Description of the three datasets.

Dataset name	**# of instances**	**# of features**	**Source**
German Credit Data	1000	24	UMLR
Australian Credit Approval	690	14	UMLR
Japanese Credit Data	690	15	UMLR

are labeled, and the Japanese data is unlabeled and has far fewer instances than the Australian and German data. This task aims to answer two questions: a) "Can we use knowledge from Australian and German credit records to classify unlabeled Japanese data (small dataset)?"; and b) "is multi-source better than single-source for classifying unlabeled Japanese data (small dataset)?"

8.5.2 Experimental Setup

This section presents baselines and implementation details of all baselines and SFERNN.

8.5.2.1 Baselines

It is important to consider with which baselines to compare the SFERNN. Dimensional reduction technology can be applied to force domains to have the same number of features. We call this method dimension reduction geodesic flow kernel (DG) [Liu et al., 2018b]. KCCA [Yeh et al., 2014], FFF-GFK [Liu et al., 2017], UFFFG [Liu et al., 2018c], RLG [Liu et al., 2020b], GLG [Liu et al., 2020b] and F-HeUDA [Liu et al., 2018b], as existing HeUDA methods, are also selected as baselines. Following [Liu et al., 2018b, 2017, 2018c], two non-transfer methods are considered: A1 and CM.

Although deep-learning based methods were originally designed for homogeneous domains, we only need to change the number of neurons in the second layer of these methods to make them suitable for heterogeneous domains. Consequently, DANN [Ganin et al., 2016a] and DAN [Long et al., 2019] are selected as baselines.

8.5.2.2 Implementation details

For Algorithm 8.1, we set IterM as 1000, δ as 4 and η as 0.5. For Algorithm 8.2, we set N_l as 50, MaxIter as 1000, η as 0.1, λ_1 as 1, λ_2 as 1.5. Following [Li et al., 2014; Pan et al., 2011; Pan and Yang, 2010; Xiao and Guo, 2015], SVM (required by DG, FFF-GFK, UFFFG, GLG and F-HeUDA) was implemented using LIBSVM. We adopted the default parameters for SVM since there are no labeled data in the target domain. Since there are no existing pairs on these datasets, we randomly matched instances from each domain as pairs for the KCCA method. We ran the experiments 50 times for each method on each task.

DAN is a neural network with five layers: input layer, hidden layer I, hidden layer II, representation layer and output layer. The number of neurons in hidden layer I (and II) is 50 and the number of neurons in the representation layer is 100. The classifier for the DANN methods is also a neural network (including five layers) and has the same settings as DAN. The domain classifier for DANN is a three-layer neural network. The number of neurons in the hidden layer of DANN's domain classifier is set to 100. Adam optimizer is adopted to optimize parameters of DAN and DANN.

Accuracy was used as the test metric as it has been widely adopted in the literature [Ghifary et al., 2017; Li et al., 2014; Pan et al., 2011]. The definition

follows.

$$Accuracy = \frac{|x \in X_t : g(x) = y(x)|}{|x \in X_t|},$$

where $y(x)$ is the ground truth label of x, while $g(x)$ is the label predicted by the SVM classification algorithm or SFERNN. All experiments were conducted on an Intel(R) Core(TM) i7-4770 CPU at 3.40Ghz with a memory of 64 GB running Windows 7 professional 64-bit operating system.

8.5.3 Experiment I: Classification Accuracy

We compare SFERNN with baselines in terms of *Accuracy* on three tasks in this section. Tables 8.2, 8.3 and 8.4 show accuracy of baselines and SFERNN on Tasks 1, 2 and 3 (target domains are Australian Credit, German Credit and Japanese Credit, respectively). From results in three tables, it is clear that SFERNN outperforms all baselines in terms of average accuracy. Our overall analysis of the comparative results reveals the following insights:

1) SFERNN performs stably when n_t changes from 40 to 400. Although GLG performs better than SFERNN when $n_t = 400$, its accuracy varies greatly when n_t changes from 40 to 400.

2) When we have multiple source domains, it is hard to select a source domain that is better than the others. However, the performance of SFERNN indicates that we can directly use knowledge from these domains to help us annotate data in target domain with better accuracy. Accuracy of SFERNN is higher than that of all single-source HeUDA methods;

3) Although deep-learning methods have considerable potential for finding a domain-invariant representation of both domains, they cannot be directly used

Table 8.2: Classification accuracy of the baselines and SFERNN on the Australian credit task (Task 1). SFERNN can extract useful information from both the *source domains* (SDs) and obtain better average accuracy on the *target domain* (TD) than all the baselines, no matter which source domain these baselines select.

SDs	Methods	n_t of TD: Australian Credit					Average
		40	100	200	300	400	accuracy
G	A1	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%
	CM	47.75%	48.80%	50.05%	53.56%	53.56%	50.75%
	DR	45.63%	46.00%	48.23%	48.25%	46.98%	47.02%
	KCCA	49.75%	49.75%	50.70%	53.30%	54.09%	51.52%
	DAN	50.37%	45.90%	47.43%	50.47%	49.38%	48.63%
	DANN	50.68%	52.95%	53.53%	54.03%	56.43%	53.39%
	FFF	68.38%	70.00%	71.00%	69.48%	69.73%	69.72%
	UFFF	72.50%	68.25%	72.68%	71.08%	69.35%	70.77%
	RLG	72.75%	71.90%	70.10%	72.80%	72.95%	72.10%
	GLG	67.88%	69.30%	74.30%	74.23%	74.88%	72.12%
	F-HeUDA	74.75%	74.90%	76.35%	75.23%	75.69%	75.38%
J	A1	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%
	CM	47.75%	48.80%	50.05%	53.56%	53.56%	50.75%
	DR	66.88%	66.35%	66.33%	66.73%	66.15%	66.49%
	KCCA	51.88%	51.95%	52.70%	55.02%	52.20%	52.75%
	DAN	53.00%	51.45%	49.88%	57.38%	46.81%	51.70%
	DANN	45.19%	59.80%	60.75%	63.08%	61.31%	58.03%
	FFF	69.13%	71.80%	72.60%	72.50%	71.88%	71.58%
	UFFF	69.13%	70.85%	73.43%	67.47%	68.08%	69.79%
	RLG	72.25%	73.50%	73.78%	70.73%	72.46%	72.54%
	GLG	76.88%	78.95%	77.83%	79.55%	78.68%	78.38%
	F-HeUDA	79.13%	78.35%	78.18%	79.83%	79.89%	79.07%
G+J	SFERNN	79.13%	81.15%	78.93%	79.93%	79.53%	**79.73%**

to address HeUDA problem. The main reason for this is that deep-learning based methods (DAN and DANN) do not prevent negative transfer in heterogeneous scenarios. Some constraints should be considered to ensure a neural network prevents negative transfer.

4) The DANN method produces higher classification accuracy than DAN, which indicates that adversarial learning method is more suitable for the HeUDA problem compared to DAN method.

5) For a specific target domain, the classification accuracy of a HeUDA method varies greatly due to different source domains. For example, on the task G2A, the average accuracy of F-HeUDA is 75.38%. However, the average accuracy of F-HeUDA is 79.07% on task J2A. Namely, F-HeUDA cannot perform stably when the source domain changes. Nonetheless, SFERNN overcomes this drawback of single-source HeUDA methods and performs well on target domain via using multiple source domains.

8.5.4 Experiment II: Stability Analysis

In this section, We compare SFERNN with baselines in terms of standard deviation (STD) of accuracy on three tasks. Table 8.5, 8.6 and 8.7 show STD of accuracy of baselines and SFERNN on Tasks 1, 2 and 3 (target domains are Australian Credit, German Credit and Japanese Credit, respectively). From results in three tables, it is clear that SFERNN achieves performance. Our overall analysis of the comparative results reveals the following insights:

1) DAN, as a deep learning method, has a high STD of classification accuracy. It means that DAN has an unstable classification performance. The main reason for this is that DAN does is not able to prevent the negative transfer, which causes that its accuracy is very high or very low (especially on task A2J);

2) By leveraging knowledge from both domains, SFERNN has lower average STD than all baselines have on most tasks;

3) Since deep-learning based methods (DAN and DANN) do not prevent negative transfer, their classification results are unstable in most tasks. This means that a neural network, as a mapping function, cannot be directly used to

Table 8.3: Classification accuracy of the baselines and SFERNN on the German credit task (Task 2). SFERNN can extract useful information from both the *source domains* (SDs) and obtain a better average accuracy on the *target domain* (TD) than all the baselines, no matter which source domain these baselines select.

SDs	Methods	n_t of TD: German Credit					Average
		40	100	200	300	400	accuracy
A	A1	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%
	CM	46.63%	49.40%	50.68%	49.85%	49.21%	49.15%
	DR	51.38%	51.35%	51.28%	51.53%	51.86%	51.48%
	KCCA	50.00%	48.35%	48.85%	51.15%	51.68%	50.01%
	DAN	49.19%	50.00%	57.55%	40.55%	42.79%	48.02%
	DANN	50.23%	52.30%	55.18%	55.00%	55.24%	53.59%
	FFF	55.63%	56.65%	57.30%	58.58%	59.28%	57.49%
	UFFF	58.88%	57.60%	57.88%	56.72%	56.66%	57.55%
	RLG	56.63%	58.90%	58.95%	59.22%	58.60%	58.46%
	GLG	59.00%	58.95%	58.13%	59.85%	58.96%	58.98%
	F-HeUDA	59.25%	60.25%	59.73%	59.58%	59.89%	59.74%
J	A1	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%
	CM	46.63%	49.40%	50.68%	49.85%	49.21%	49.15%
	DR	46.88%	44.45%	46.85%	47.35%	46.45%	46.40%
	KCCA	48.00%	51.05%	50.83%	50.23%	50.90%	50.20%
	DAN	49.25%	51.70%	49.05%	48.13%	52.70%	50.17%
	DANN	51.23%	52.60%	51.53%	54.28%	52.98%	52.52%
	FFF	59.50%	56.70%	58.18%	58.45%	59.00%	58.37%
	UFFF	54.63%	54.20%	54.05%	56.58%	55.84%	55.06%
	RLG	57.50%	56.35%	55.40%	56.02%	59.03%	56.86%
	GLG	58.25%	60.20%	58.23%	58.27%	59.78%	58.94%
	F-HeUDA	59.63%	59.00%	61.03%	59.05%	59.59%	59.66%
A+J	SFERNN	60.25%	58.95%	59.42%	60.57%	59.69%	**59.78%**

Table 8.4: Classification accuracy of the baselines and SFERNN on the Japanese credit task (Task 3). SFERNN can extract useful information from both the *source domains* (SDs) and obtain a better average accuracy on the *target domain* (TD) than all the baselines, no matter which source domain these baselines select.

SDs	Methods	n_t of TD: Japanese Credit					Average
		40	100	200	300	400	accuracy
A	A1	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%
	CM	46.63%	47.50%	49.08%	49.10%	49.34%	48.33%
	DR	54.50%	53.55%	54.25%	53.45%	54.23%	54.00%
	KCCA	52.50%	49.85%	52.78%	54.57%	56.23%	53.18%
	DAN	46.13%	51.40%	45.45%	53.82%	49.96%	49.35%
	DANN	48.50%	51.80%	61.65%	60.03%	59.95%	56.39%
	FFF	67.25%	64.25%	64.75%	58.05%	57.61%	62.38%
	UFFF	68.75%	66.00%	64.00%	65.70%	63.45%	65.58%
	RLG	67.63%	62.65%	62.53%	63.43%	65.43%	64.33%
	GLG	65.00%	62.65%	64.95%	64.62%	65.04%	64.45%
	F-HeUDA	69.13%	68.30%	68.65%	69.32%	69.86%	69.05%
G	A1	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%
	CM	46.63%	47.50%	49.08%	49.10%	49.34%	48.33%
	DR	45.50%	45.20%	46.85%	45.38%	46.49%	45.88%
	KCCA	56.75%	51.25%	53.90%	49.98%	53.14%	53.00%
	DAN	49.36%	47.90%	50.53%	49.92%	49.31%	49.40%
	DANN	50.10%	49.10%	50.80%	50.47%	51.04%	50.30%
	FFF	58.88%	62.40%	59.10%	58.77%	56.19%	59.07%
	UFFF	65.00%	60.30%	63.13%	57.67%	62.00%	61.62%
	RLG	61.38%	59.25%	64.10%	60.18%	62.80%	61.54%
	GLG	61.13%	59.50%	63.50%	62.53%	64.28%	62.19%
	F-HeUDA	66.13%	66.85%	66.88%	68.42%	67.34%	67.12%
A+G	SFERNN	71.37%	67.65%	69.95%	69.87%	70.76%	**69.92**%

address the HeUDA problem. Although deep-learning methods have considerable potential for finding a representation of two domains, some constraints should be considered to ensure a neural network has stable performance;

4) Compared to DAN, DANN method produces lower STD of accuracy, which is another evidence that adversarial learning method may be suitable for the HeUDA problem. In the future, adversarial learning based HeUDA methods should be developed;

5) For a specific target domain, the STD of classification accuracy of a HeUDA method varies greatly due to different source domains. For example, on task A2G, STD of accuracy of F-HeUDA is 4.91% ($n_t = 200$). However, STD of accuracy of F-HeUDA is 3.33% on the task J2G. It is an evidence that F-HeUDA cannot perform stably when the source domain changes. Nonetheless, SFERNN can always maintain a low STD values.

8.5.5 Experiment III: Parameters Sensitivity

In this section, we analyze how the learning rate in Algorithm 8.2 and λ_2 influence the average accuracy of SFERNN on 3 tasks (λ_1 is set to 1 in this section).

Figure 8.4 shows average accuracy of SFERNN on three tasks when learning rate of Algorithm 8.2 changes from 0.01 to 0.4 and λ_2 changes from 0.1 to 2.5. According to Tables 8.2, 8.3 and 8.4, it is clear that SFERNN still outperforms all baselines on tasks 1 and 3 even when both parameters change a lot.

On task 1, it is clear that SFERNN will have lower accuracy when λ_2 is around 0.5 (the blue valley in Figure 8.4-(a)). On task 2, if the value of λ_2 is around 0.5 and the value of learning rate is around 0.2, SFERNN will have low

Table 8.5: STD of classification accuracy of baselines and SFERNN on the Australian credit task (Task 1). SFERNN can use knowledge in the *source domains* (SDs) to obtain more stable accuracy across 50 experiments on the *target domain* (TD) than most baselines.

SDs	Methods	n_t of TD: Australian Credit					Average
		40	100	200	300	400	STD
G	A1			-			-
	CM	24.52%	26.76%	23.10%	27.26%	27.26%	25.78%
	DR	8.84%	5.50%	4.45%	4.63%	3.41%	5.37%
	KCCA	9.80%	9.80%	11.29%	9.58%	11.77%	10.45%
	DAN	7.54%	22.57%	24.89%	21.83%	23.40%	20.05%
	DANN	9.63%	8.24%	8.00%	6.68%	6.93%	7.90%
	FFF	9.40%	6.87%	4.75%	6.94%	8.67%	7.33%
	UFFF	8.23%	6.11%	7.31%	8.32%	9.45%	7.88%
	RLG	7.47%	5.94%	5.98%	6.64%	7.47%	6.70%
	GLG	16.77%	9.17%	6.52%	5.39%	4.74%	8.52%
	F-HeUDA	7.86%	5.19%	2.41%	2.28%	1.77%	3.90%
J	A1			-			-
	CM	24.52%	26.76%	23.10%	27.26%	27.26%	25.78%
	DR	7.16%	4.61%	2.78%	2.28%	1.80%	3.73%
	KCCA	9.52%	10.89%	7.39%	10.19%	9.91%	9.58%
	DAN	13.48%	11.53%	16.95%	20.13%	21.52%	16.72%
	DANN	8.15%	11.91%	10.15%	9.00%	8.62%	9.57%
	FFF	10.43%	10.99%	7.62%	9.34%	9.40%	9.56%
	UFFF	8.56%	8.82%	7.74%	9.09%	8.46%	8.53%
	RLG	9.14%	5.82%	6.13%	6.09%	8.14%	7.06%
	GLG	8.77%	3.15%	3.32%	1.70%	1.30%	3.65%
	F-HeUDA	6.19%	3.27%	2.92%	1.97%	1.49%	**3.17%**
G+J	SFERNN	6.35%	4.20%	3.04%	1.87%	1.52%	3.40%

average accuracy (the blue valley in Figure 8.4-(b)). On task 3, if values of λ_2 and learning are high, SFERNN will have lower accuracy (the blue valley in Figure 8.4-(c)).

After analyzing these three subfigures, we can see that a high value of λ_2 and a high value of learning rate will result in the low accuracy of SFERNN, which should be avoided. Low value of λ_2 is suggested according to this figure.

Table 8.6: The STD of the classification accuracy of the baselines and SFERNN on the German credit task (Task 2). SFERNN can use knowledge in *source domains* (SDs) to obtain more stable accuracy across 50 experiments on the target domain (TD) than most baselines.

SDs	Methods	n_t of TD: German Credit					Average
		40	100	200	300	400	STD
A	A1			-			-
	CM	9.50%	8.77%	4.46%	4.84%	4.58%	6.43%
	DR	5.10%	5.08%	3.21%	2.42%	1.62%	3.49%
	KCCA	9.60%	4.97%	5.91%	4.84%	5.54%	6.17%
	DAN	8.59%	7.56%	6.63%	6.82%	6.52%	7.23%
	DANN	4.50%	3.35%	3.51%	3.16%	3.63%	3.63%
	FFF	7.90%	5.86%	3.57%	4.21%	3.32%	4.97%
	UFFF	10.31%	5.91%	4.52%	4.90%	4.86%	6.10%
	RLG	5.21%	6.23%	5.81%	3.30%	3.58%	4.83%
	GLG	7.32%	4.38%	3.02%	2.92%	2.69%	4.07%
	F-HeUDA	6.98%	5.64%	4.91%	3.53%	3.92%	4.99%
J	A1			-			-
	CM	9.50%	8.77%	4.46%	4.84%	4.58%	6.43%
	DR	7.52%	4.88%	3.58%	2.82%	1.53%	4.07%
	KCCA	9.51%	7.82%	7.43%	5.15%	6.36%	7.25%
	DAN	4.39%	11.83%	17.28%	18.96%	18.05%	14.10%
	DANN	7.92%	7.04%	5.37%	3.88%	4.61%	5.76%
	FFF	7.59%	5.34%	4.15%	3.57%	3.94%	4.92%
	UFFF	6.99%	5.75%	5.35%	5.82%	5.07%	5.79%
	RLG	9.18%	5.27%	5.40%	5.57%	4.45%	5.97%
	GLG	5.97%	5.90%	3.11%	2.65%	2.17%	3.96%
	F-HeUDA	6.60%	4.60%	3.33%	2.22%	2.03%	3.76%
A+J	SFERNN	5.73%	4.57%	2.58%	1.83%	2.39%	**3.42%**

8.6 Summary

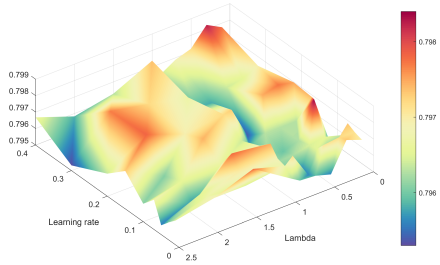
In this chapter, a novel and realistic problem is formalized, called *multi-source heterogeneous unsupervised domain adaptation* (MsHeUDA). In MsHeUDA problem, there are c different-dimension (i.e., heterogeneous) source domains and one target domain, where source domains contains abundant labeled data and target domain only contains few unlabeled data. We aim to train a classifier for data in target domain with labeled data in source domains and unlabeled data

Table 8.7: The STD of classification accuracy of the baselines and SFERNN on the Japanese credit task (Task 3). SFERNN can use knowledge in *source domains* (SDs) to obtain more stable accuracy across 50 experiments on the *target domain* (TD) than most baselines.

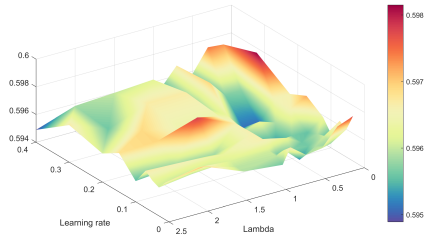
SDs	Methods	n_t of TD: Japanese Credit					Average
		40	100	200	300	400	STD
A	A1			-			-
	CM	5.92%	4.52%	4.39%	3.56%	2.50%	4.18%
	DR	10.63%	5.39%	5.43%	4.82%	4.97%	6.25%
	KCCA	12.49%	10.80%	9.20%	8.68%	8.37%	9.91%
	DAN	25.16%	21.67%	23.62%	25.06%	22.79%	23.66%
	DANN	5.28%	6.01%	5.41%	10.31%	8.59%	7.12%
	FFF	10.38%	13.05%	10.37%	11.32%	7.66%	10.56%
	UFFF	7.14%	7.95%	8.16%	9.50%	12.95%	9.14%
	RLG	12.91%	7.27%	7.79%	7.10%	9.74%	8.96%
	GLG	10.42%	7.10%	5.52%	6.76%	6.79%	7.32%
	F-HeUDA	7.23%	3.79%	4.33%	3.54%	2.45%	4.27%
G	A1			-			-
	CM	5.92%	4.52%	4.39%	3.56%	2.50%	4.18%
	DR	13.97%	10.97%	11.56%	11.02%	10.64%	11.63%
	KCCA	11.64%	8.17%	9.24%	8.58%	8.65%	9.26%
	DAN	9.01%	8.84%	6.69%	5.72%	5.08%	7.07%
	DANN	0.79%	7.40%	7.48%	7.75%	9.04%	6.49%
	FFF	11.16%	9.72%	9.02%	8.95%	9.79%	9.73%
	UFFF	11.36%	7.41%	7.59%	8.98%	6.79%	8.42%
	RLG	10.02%	9.22%	7.84%	6.79%	7.14%	8.20%
	GLG	10.34%	5.63%	5.72%	5.62%	6.18%	6.70%
	F-HeUDA	10.04%	4.59%	2.84%	2.42%	1.75%	4.33%
A+G	SFERNN	7.15%	5.49%	3.86%	1.75%	1.69%	**3.99%**

in target domain.

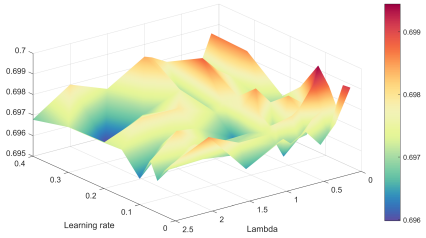
To solve this problem, this chapter first presents a new fuzzy equivalence relation, called *multi-source shared fuzzy equivalence relation* (MsSFER), to extract shared fuzzy information in c source domains and target domain. With help of MsSFER, we can cluster features in each domain (including c source domains and target domain) as N_l categories (see N_l in Figure 8.3). Then, a novel neural network, called *shared-fuzzy-equivalence-relation neural network*



(a) Task 1: GJ2A



(b) Task 2: AJ2G



(c) Task 3: AG2J

Figure 8.4: Analysis of parameters' sensitivity on 3 tasks. Learning rate represents η in Algorithm 8.2 and Lambda represents λ_2 in loss function of SFERNN.

(SFERNN), is proposed to solve the MsHeUDA problem.

Since there are no existing methods for the MsHeUDA problem, this study is the first to propose an effective solution to address this problem. The success of SFERNN is due to the proposed fuzzy equivalence relation, MsSFER. MsSFER can help extract shared fuzzy information contained in multiple heterogeneous domains. Based on this shared fuzzy information, knowledge from c heterogeneous source domain can be transferred to target domain. The efficacy of SFERNN is validated using three real-world MsHeUDA tasks. The experimental results reveal that SFERNN can reliably transfer knowledge from multiple heterogeneous source domains to target domain.

CHAPTER 9

CONCLUSION AND FUTURE STUDY

This chapter concludes the entire thesis and provides some further research directions for realistic transfer learning.

9.1 Conclusions

Recent research of transfer learning has shown a decent ability to transfer knowledge from a source domain to a target domain, however most research require certain assumptions to ensure their efficacy. These assumptions are probably not realistic, which means that existing transfer learning methods still face several unsolved and challenging problems in real world. To sum up, existing transfer learning methods still face the following problems in real world: i) How to test if feature spaces of two domains are from different distributions; ii) How to transfer knowledge when labels in the source domain cannot be

perfectly annotated (i.e., the source domain contains noisy labels); iii) How to transfer knowledge when source domains and the target domain have different dimensions (i.e., the target domain is heterogeneous); and iv) How to transfer knowledge across multiple source domains and a heterogeneous target domain.

To solve the above-mentioned challenges, this thesis proposed four concrete research questions and corresponding research objectives. The findings of this study are summarized as follows:

1. *Discrepancy of diverse subsets and deep-kernel maximum mean discrepancy are proposed as new two-sample tests for modern machine-learning datasets.* (to achieve objective 1)

   We address key problems faced by function-based tests and subset-based tests. As results, we propose discrepancy of diverse subsets for low-dimension data and deep-kernel maximum mean discrepancy for high-dimension data. Corresponding theories have also be proven to ensure the efficacy of the proposed tests. Both new tests are validated by several real-world datasets.

2. *The theoretical foundation is built for HoUDA problem under noisy scenario, and we propose a new method to this new setting.* (to achieve RQ2)

   Since the foundation of HoUDA methods is all based on [Ben-David et al., 2010b] that assumes that source domain only contains clean labels, we first, in theory, to propose a new theoretical bound for HoUDA problem under noisy scenario, and formalize this new problem as wildly unsupervised domain adaptation. The proposed bound reveals why existing HoUDA methods do not work well under noisy scenario. Guided by the new bound, we propose a solution called Butterfly to wildly unsupervised

domain adaptation and validate the efficacy of Butterfly using real-world datasets.

3. *We build the theoretical foundation and propose new methods for heterogeneous unsupervised domain adaptation problem.* (to achieve RQ3)

   Due to the lack of theoretical foundations for *heterogeneous unsupervised domain adaptation* (HeUDA) problem, HeUDA methods assume the existence of parallel sets that can bridge two heterogeneous domains, which is not realistic. To remove this assumption, we first build the theoretical foundation to show when and how we can transfer transfer knowledge from a source domain to a heterogeneous and unlabeled target domain. The theory helps us prevent the negative transfer and reliably transfer knowledge across two domains. Based on the proposed theory, we propose two valid HeUDA methods: GLG and F-HeUDA. The first one focuses on the situation where two domains are balanced and the second one focuses on the imbalanced situation. Finally, we evaluate the efficacy of GLG and F-HeUDA using real-world datasets.

4. *We develop a new method to transfer knowledge from multiple source domains to a heterogeneous target domain where there are no labeled samples in target domain.* (to achieve RQ4)

   In real world, given a target domain, we probably have multiple different-dimension (heterogeneous) source domains. Thus, to step towards realistic transfer learning, we extend the HeUDA problem to *multi-source HeUDA* (MsHeUDA) problem. After formalizing MsHeUDA problem, we propose

a novel fuzzy-relation neural network to address this problem and use real-world datasets to evaluate its efficacy.

This thesis also applies the proposed tests and methods on many real-world applications:

1. *To distinguish the signature of processes producing Higgs bosons.*

   It is a classical problem to distinguish the signature of processes producing Higgs bosons from those background processes that do not in the field of high-energy physics [Baldi et al., 2014]. In this thesis, the proposed two-sample tests (DDS and deep-kernel-based MMD) are used to solve this real-world problem.

2. *To test if there exists dataset shift between CIFAR-10 and CIFAR-10.1.*

   CIFAR-10.1 [Recht et al., 2019] is an attempt to collect a new testing set for the very popular *CIFAR-10* image classification dataset [Krizhevsky, 2009]. In this thesis, the proposed deep-kernel-based MMD is used to successfully show that *CIFAR-10.1* and *CIFAR-10* are significantly different.

3. *To recognize objects using images labeled by Bing search engine.*

   Images are autonomously labeled by Bing search engine, and we regarded these labeled images as a source domain. There are also a lot of images of objects, which are regarded as a target domain. In this thesis, the Butterfly-Net is proposed to use the knowledge of the Bing-labeled source domain to recognize images of objects in the target domain.

4. *To diagnose if a patient has Breast Cancer by using existed diagnostic data.*

Diagnostic features are computed from a digitized image of a fine needle aspirate of a breast mass, which contains more information related to the cancer. The proposed GLG and F-HeUDA methods successfully use the knowledge of the existed labeled diagnostic data (i.e., the source domain) to help annotate unlabeled digitized images (i.e., to diagnose if a patient has Breast Cancer).

9.2 Future Study

This thesis identifies the following directions as future work:

1. *Autonomous selection of source domains.*

   Since existing transfer learning methods require users to select a source domain for a specific target domain, the human-nomination process will significantly influence the performance of transfer learning methods when we have a lot of candidate source domains. In the future, we aim to propose a method to autonomously select the most-suitable source domains for a target domain, without needing human-nomination of the source domains for a given target domain.

2. *A new two-sample test for imprecise observations.*

   In existing non-parametric two-sample tests, they aim to determine whether two sets of *precise* observations (i.e., samples) are from the same distribution. However, in real world, precise observations are often unavailable. For example, readings on an analogue measurement equipment are not precise numbers but intervals since there is only a finite number of deci-

mals available. Hence, we will consider a new and more realistic problem setting—*two-sample testing on imprecise observations* and propose a new two-sample test to address this problem.

3. *A WUDA method for the situation that source domain contains instance-dependent noise and feature noise.*

   Chapter 5 formalizes the WUDA problem and propose a method to address it. In the future, we will propose more methods to address this problem under the situation that noise contained in source domain is more complicated, such as instance-dependent noise and feature noise.

4. *A unified learning framework and generalization bound for HeUDA problem.*

   Chapter 6 presents a theorem to show when we can transfer knowledge from a source domain to a heterogeneous and unlabeled target domain. This theorem ensures positive transfer. In the future, based on this theorem, we will propose a unified learning framework and generalization bound for HeUDA problem.

5. *More industry-level applications of the proposed methods*

   Although the proposed tests and methods are successfully used to solve many real-world problems (such as object recognition, cancer detection and credit assessment), we still need to develop more prototypes of these methods and apply these prototypes in the industry. Moreover, the proposed methods will be used to address more real-world problems in fields of Agronomy, Biology, Cybersecurity, Energy and Meteorology.

APPENDIX

A.1 Appendix of Chapter 3

A.1.1 A Corollary of Radon-Nikodym Theorem

First, we recall the Radon-Nikodym theorem.

Theorem A.1 (Radon-Nikodym theorem)**.** *Let $(\mathcal{X}, \mathcal{A})$ be a measurable space and let μ and ν be σ-finite positive measures on $(\mathcal{X}, \mathcal{A})$. If ν is absolutely continuous with respect to μ, then there is an $\mathcal{A}$-measurable function $g : \mathcal{X} \to [0, +\infty)$ such that $\nu(A) = \int_A g d\mu$ holds for each A in $\mathcal{A}$. the function g is unique up to μ-almost everywhere equality.*

Please see the detailed proof of this theorem in the page 123 of Cohn [2013]. Then, we prove following corollary.

Corollary A.1. *Given two measurable functions p and q defined in a measurable space $(\mathcal{X}, \mathcal{A})$, if, $\forall A \in \mathcal{A}$, a σ-finite positive measure $\mu(A) = \int_{x \in A}(p(x) - q(x))dx = 0$, then $p = q$ almost everywhere.*

Proof. $\forall A \in \mathcal{A}$, we define $v(A) = \int_A p d\mu$. We will prove that p is unique up to μ-almost everywhere (this is the desired result). First, we prove v is absolutely continuous with respect to μ. Considering following sets

$$A_n = \{x | n - 1 < p(x) \leq n\},$$

so, we have

$$p(x) \leq \sum_{n=1}^{\infty} n \mathbf{1}_{A_n}(x).$$

Then, we know

$$v(A) = \int_A p d\mu \leq \sum_{n=1}^{\infty} n \int_A \mathbf{1}_{A_n} d\mu \leq \sum_{n=1}^{\infty} n\mu(A_n) = 0.$$

Hence, v is absolutely continuous with respect to μ. Following Radon-Nikodym theorem, we know p has a unique form. If we have $v = \int_A p d\mu = \int_A q d\mu$, then $p = q$ almost everywhere. ■

This corollary demonstrates one fact: if we have the same number of observations in any measurable subsets from p and q, then $p = q$ almost everywhere.

A.1.2 Proofs of Theorems

A.1.2.1 Proof of Theorem 3.1

Proof. We denote $\Delta(p, q, X_1, X_2) = |\text{DDS}(p, q) - \hat{\text{DDS}}(X_1, X_2)|$. Then, in terms of Lemma 3.1, changing either x_{1i} or x_{2i} in $\Delta(p, q, X_1, X_2)$ results in changes in

magnitude of at most $2M_{pq}/\gamma^d n_1$ or $2M_{pq}/\gamma^d n_2$, respectively. We can apply McDiarmid's inequality Gretton et al. [2012]; McDiarmid [1989], given a denominator in the exponent of

$$n_1\left(\frac{2M_{pq}}{\gamma^d n_1}\right)^2 + n_2\left(\frac{2M_{pq}}{\gamma^d n_2}\right)^2 = 4M_{pq}^2\left(\frac{1}{\gamma^{2d} n_1} + \frac{1}{\gamma^{2d} n_2}\right) = 4M_{pq}^2\frac{n_2+n_1}{\gamma^{2d} n_1 n_2}.$$

to obtain

(A.1)

$$\Pr_{X_1,X_2}(\Delta(p,q,X_1,X_2) - \mathbb{E}_{X_1,X_2}(\Delta(p,q,X_1,X_2)) > \epsilon) \leq exp\left(\frac{-\epsilon^2\gamma^{2d} n_1 n_2}{2M_{pq}^2(n_1+n_2)}\right).$$

Next, we need to calculate $\mathbb{E}_{X_1,X_2}(\Delta(p,q,X_1,X_2))$. First, we need to prove the following formula.

$$\sqrt{n_1}\Big(\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{1i},x_{1j}) - \int_{\mathbb{R}^d} L_{x_{1i}}(x_1')dp_{x_1'}\Big) \xrightarrow{\mathcal{D}} \mathcal{N}(0,\sigma_{1i}^2), \tag{A.2}$$

where $\sigma_{1,1i}^2 = \mathrm{Var}(L(x_{1i},x_{1j})$. Because

$$\begin{aligned}\mathbb{E}\Big(\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{1i},x_{1j})\Big) &= \frac{1}{n_1}\sum_{j=1}^{n_1}\mathbb{E}\Big(L(x_{1i},x_{1j})\Big)\\ &= \frac{n_1}{n_1}\int_{\mathbb{R}^d} L(x_{1i},x_{1j})dp_{x_{1j}}\\ &= \int_{\mathbb{R}^d} L_{x_{1i}}(x_1')dp_{x_1'},\end{aligned}$$

and $\sigma_{1i}^2 < +\infty$, using the central limit theorem, we prove the Formula (A.2). Thus, we have

$$\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{1i},x_{1j}) = \int_{\mathbb{R}^d} L_{x_{1i}}(x_1')dp_{x_1'} + O_{p_1}(\frac{1}{\sqrt{n_1}}), \tag{A.3}$$

where $O_{p_1}(1/\sqrt{n_1}) = 1/\sqrt{n_1}\mathcal{N}(0,\sigma_{1,1i}^2)$. Similarly, we know

$$\frac{1}{n_2}\sum_{j=1}^{n_1} L(x_{1i},x_{2j}) = \int_{\mathbb{R}^d} L_{x_{1i}}(x_2')dq_{x_2'} + O_{p_2}(\frac{1}{\sqrt{n_2}}), \tag{A.4}$$

where $O_{p_2}(1/\sqrt{n_2}) = 1/\sqrt{n_2}\mathcal{N}(0,\sigma^2_{2,1i})$. Substituting Eq. (A.3) and Eq. (A.4) into $\mathrm{D\hat{D}S}_p$, we have

$$\begin{aligned}\mathrm{D\hat{D}S}_p &= \frac{1}{n_1}\sum_{i=1}^{n_1}\left|\frac{1}{n_1}\sum_{j=1}^{n_1}L(x_{1i},x_{1j}) - \frac{1}{n_2}\sum_{j=1}^{n_2}L(x_{1i},x_{2j})\right|,\\ &= \frac{1}{n_1}\sum_{i=1}^{n_1}\left|\int_{\mathbb{R}^d}L_{x_{1i}}(x')d_{p_{x'_1}-q_{x'_2}} + O_p(\frac{1}{\sqrt{n_1}}+\frac{1}{\sqrt{n_2}})\right|,\end{aligned} \tag{A.5}$$

where $O_p(1/\sqrt{n_1}+1/\sqrt{n_2}) = \mathcal{N}(0,\sigma^2_{1,1i}/n_1+\sigma^2_{2,1i}/n_2)$. Using Taylor series, expand Eq. (A.5) at the value $\int_{\mathbb{R}^d}L_{x_{1i}}(x')d_{p_{x'_1}-q_{x'_2}}$, we have

$$\mathrm{D\hat{D}S}_p = \frac{1}{n_1}\sum_{i=1}^{n_1}\left(\left|\int_{\mathbb{R}^d}L_{x_{1i}}(x')d_{p_{x'_1}-q_{x'_2}}\right| + \left|O'_p(\frac{1}{\sqrt{n_1}}+\frac{1}{\sqrt{n_2}})\right|\right),$$

where $O'_p(1/\sqrt{n_1}+1/\sqrt{n_2}) = \mathcal{N}(0,M\sigma^2_{1,1i}/n_1+M\sigma^2_{2,1i}/n_2)$ and M is a positive finite number. Because $\int_{\mathbb{R}^d}L_{x_{1i}}(x')d_{p_{x'_1}-q_{x'_2}}$ is a function related to x_{1i}, So, we have

$$\begin{aligned}\mathbb{E}_{X_1,X_2}(\mathrm{D\hat{D}S}_p - \mathrm{DDS}_p) &= \mathbb{E}_{X_1,X_2}\left(O_{p_1}(1/\sqrt{n_1}) + \frac{1}{n_1}\sum_{i=1}^{n_1}\left|O'_p(\frac{1}{\sqrt{n_1}}+\frac{1}{\sqrt{n_2}})\right|\right)\\ &= \frac{1}{n_1}\sum_{i=1}^{n_1}\mathbb{E}\left(\left|O'_p(\frac{1}{\sqrt{n_1}}+\frac{1}{\sqrt{n_2}})\right|\right).\end{aligned}$$

Because $|O'_p|$ is a random variable obeys the half Normal distribution, we know

$$\mathbb{E}_{X_1,X_2}(\mathrm{D\hat{D}S}_p - \mathrm{DDS}_p) = \frac{1}{n_1}\sum_{i=1}^{n_1}\frac{\sqrt{2}}{\sqrt{\pi}}\sqrt{M\Big(\frac{\sigma^2_{1,1i}}{n_1}+\frac{\sigma^2_{2,1i}}{n_2}\Big)}.$$

Since $\mathrm{D\hat{D}S}_p = \mathrm{DDS}_p + |O'_p| > \mathrm{DDS}_p$, we know

$$\mathbb{E}_{X_1,X_2}(\Delta(p,q,X_1,X_2)) = \sum_{l=1}^{2}\frac{1}{n_l}\sum_{i=1}^{n_l}\frac{\sqrt{2}}{\sqrt{\pi}}\sqrt{M\Big(\frac{\sigma^2_{1,li}}{n_1}+\frac{\sigma^2_{2,li}}{n_2}\Big)} \tag{A.6}$$

Combining (A.1) and (A.6), the proof completes. ■

A.1.2.2 Proof of Lemma 3.2

Proof. It is clear that $L(z_i,z_j)$, as an n-by-n matrix, will not change further if X_1 and X_2 are obtained. The one-time permutation of $\mathrm{D\hat{D}S}(X_1,X_2)$ uniformly

chooses n_1 from z_i as the new X_1 and selects the rest of z_i as the new X_2, and then calculates the new $\hat{\text{DDS}}(X_1, X_2)$. We do not need to calculate this new $\hat{\text{DDS}}(X_1, X_2)$ from the very beginning (*e.g.*, re-calculating $D(x_{1i}, x_{1j})$) but rather only from $L(z_i, z_j)$. This allows us to find the asymptotic null distribution of $\hat{\text{DDS}}(X_1, X_2)$.

Consider following random variable b_j,

$$P(b_j = n_1^{-1} L(x_{1i}, z_j)) = \lambda_1, P(b_j = -n_2^{-1} L(x_{1i}, z_j)) = \lambda_2,$$

where P means the probability and $\lambda_1 + \lambda_2 = 1$. Under the permutation null ($p = q$), we know

$$n_1^{-1} \sum_{j=1}^{n_1} L(x_{1i}, x_{1j}) - n_2^{-1} \sum_{j=1}^{n_2} L(x_{1i}, x_{2j}) = \sum_{j=1}^{n} b_j.$$

Because $L(z_i, z_j)$ is fixed after obtaining X_1 and X_2, 1) each b_j is independent and 2) $n_1^{-1} \sum_{j=1}^{n_1} L(x_{1i}, x_{1j}) - n_2^{-1} \sum_{j=1}^{n_2} L(x_{1i}, x_{2j})$ is the sum of n independent random variables. To apply Lyapunov central limit theorem to $\sum b_j$, we need to verify the following condition.

$$\lim_{n \to \infty} \frac{\sum_{j=1}^{n} \mathbb{E}\Big(\big(b_j - \mathbb{E}(b_j)\big)^3\Big)}{s_n^3} = 0, \; s_n^2 = \sum_{j=1}^{n} \mathbb{E}\Big(\big(b_j - \mathbb{E}(b_j)\big)^2\Big). \tag{A.7}$$

In terms of the definition of b_j, we know $\mathbb{E}(b_j) = 0$ and

$$\mathbb{E}\Big(\big(b_j - \mathbb{E}(b_j)\big)^3\Big) = \lambda_1 n_1^{-3} L(x_{1i}, z_j)^3 - \lambda_2 n_2^{-3} L(x_{1i}, z_j)^3,$$
$$\mathbb{E}\Big(\big(b_j - \mathbb{E}(b_j)\big)^2\Big) = \lambda_1 n_1^{-2} L(x_{1i}, z_j)^2 + \lambda_2 n_2^{-2} L(x_{1i}, z_j)^2$$

It is clear that any non-zero $n_1^{-1} L(x_{1i}, x_{1j})$ or $n_2^{-1} L(x_{1i}, x_{2j})$ has a minimum value, denoted by $L_{min} > 0$, and a maximum value, $L_{max} < +\infty$. If $\mathbb{E}\big(\big(b_j - \mathbb{E}(b_j)\big)^3\big) > 0$, we have

$$\frac{\sum_{j=1}^{n} \mathbb{E}\Big(\big(b_j - \mathbb{E}(b_j)\big)^3\Big)}{s_n^3} < \frac{n \lambda_1 L_{max}^3}{n^{\frac{3}{2}} (\lambda_1 + \lambda_2) L_{min}^2} \propto \mathcal{O}(\frac{1}{\sqrt{n}}).$$

If $\mathbb{E}\big(\big(b_j - \mathbb{E}(b_j)\big)^3\big) < 0$, we have

$$\frac{\sum_{j=1}^{n} \mathbb{E}\Big(\big(b_j - \mathbb{E}(b_j)\big)^3\Big)}{s_n^3} > \frac{-n\lambda_2 L_{max}^3}{n^{\frac{3}{2}}(\lambda_1+\lambda_2)L_{min}^2} \propto \mathcal{O}(\frac{-1}{\sqrt{n}}).$$

Hence, the condition in (A.7) is verified, which means, due to Lyapunov central limit theorem, we have

$$\frac{1}{s_n}\sum_{j=1}^{n}(b_j - \mathbb{E}(b_j)) \xrightarrow{\mathcal{D}} \mathcal{N}(0,1).$$

Based on the definitions of b_j and s_n, this lemma is proved. ■

A.1.2.3 Proof of Theorem 3.2

Proof. When, $p = q$, it is obvious that $\mathrm{DDS}(p,q) = 0$. We now prove the converse. $\forall x, x' \in \mathcal{X}$, according to the definition of $I_x(x')$, we know $I_x(x')$ is an indicator function as $L \to +\infty$ and is bounded by 1. Because $V_{x'}$ is over zero, we define two new measures as

$$\tilde{p}(S) = \int_S \frac{1}{V_{x'}} dp_{x'}, \quad \tilde{q}(S) = \int_S \frac{1}{V_{x'}} dq_{x'}, \tag{A.8}$$

where S is any subset in $\mathcal{X}$. So, $\mathrm{DDS}(p,q) = 0$ means that

$$\int \Big| \int I_x(x') d(\tilde{p}_{x'} - \tilde{q}_{x'}) \Big| d(\tilde{p}_x + \tilde{q}_x) = 0. \tag{A.9}$$

We denote $Supp = \{x | \forall \epsilon > 0, \tilde{p}_x(B_\epsilon(x) \cap \mathcal{X}) + \tilde{q}_x(B_\epsilon(x) \cap \mathcal{X}) \neq 0, x \in \mathcal{X}\}$, where $B_\epsilon(x)$ is the ball in $\mathbb{R}^d$ with radius ϵ, center x. Because $|\int I_x(x') d(\tilde{p}_{x'} - \tilde{q}_{x'})|$ is continuous, we arrive at

$$\begin{aligned} &\Big| \int I_x(x') d(\tilde{p}_{x'} - \tilde{q}_{x'}) \Big| = 0, \quad \forall x \in Supp, \\ \Rightarrow &\int I_x(x') d(\tilde{p}_{x'} - \tilde{q}_{x'}) = 0, \quad \forall x \in Supp, \\ \Rightarrow &\int_{B_{r_x}(x')} d(\tilde{p}_{x'} - \tilde{q}_{x'}) = 0, \quad \forall x \in Supp. \end{aligned} \tag{A.10}$$

Because $V_{x'}$ is over zero, we have $\tilde{p}_x = \tilde{q}_x$. Therefore, $p = q$. ■

A.1.2.4 Proof of Theorem 3.4

Proof. We first define a new random variable $a_{x_{1i}}$ as following.

$$a_{x_{1i}} = \frac{1}{n_1}\Big(\frac{1}{n_1}\sum_{j=1}^{n_1} L(x_{1i}, x_{1j}) - \frac{1}{n_2}\sum_{j=1}^{n_2} L(x_{1i}, x_{2j})\Big).$$

So, we know $\hat{\mathrm{DDS}}_X = \sum_i |a_{x_{1i}}|$. Since $L(x_{1i}, z_j)$ are fixed after obtaining X_1 and X_2 and all of x_{1i} are independent, all of $a_{x_{1i}}$ are independent. Next, to apply Lyapunov central limit theorem to $\sum_i |a_{x_{1i}}|$, we need to verify the following condition.

$$\lim_{n\to\infty} \frac{\sum_{i=1}^{n_1} \mathbb{E}\Big(\big(|a_{x_{1i}}| - \mathbb{E}(|a_{x_{1i}}|)\big)^3\Big)}{s_n^3} = 0,\ s_n^2 = \sum_{i=1}^{n_1} \mathbb{E}\Big(\big(|a_{x_{1i}}| - \mathbb{E}(|a_{x_{1i}}|)\big)^2\Big). \tag{A.11}$$

Because $n_1^{\frac{1}{2}} a_{x_{1i}}$ approaches a normal distribution when $n \to \infty$ (see Lemma 3.2), $n_1^{\frac{1}{2}} |a_{x_{1i}}|$ approaches a half normal distribution which has a first moment, a second moment and a second central moment as follows.

$$\mathbb{E}(|a_{x_{1i}}|) = \frac{\sqrt{2}}{\sqrt{\pi n_1}}\sigma_{x_{1i}},\ \mathbb{E}(|a_{x_{1i}}|^2) = \frac{\sigma_{x_{1i}}^2}{n_1},\ \mathbb{E}\Big((|a_{x_{1i}}| - \mathbb{E}(|a_{x_{1i}}|))^2\Big) = \Big(1 - \frac{2}{\pi}\Big)\frac{\sigma_{x_{1i}}^2}{n_1}. \tag{A.12}$$

Next, we will calculate the third moment of $|a_{x_{1i}}|$.

$$\mathbb{E}(|a_{x_{1i}}|^3) = \int_0^{+\infty} \frac{\sqrt{2n_1}}{\sigma_{x_{1i}}\sqrt{\pi}} \cdot e^{-\frac{n_1|a_{x_{1i}}|^2}{2\sigma_{x_{1i}}^2}} \cdot |a_{x_{1i}}|^3\, d|a_{x_{1i}}| = \frac{\sqrt{2n_1}}{\sigma_{x_{1i}}\sqrt{\pi}} \int_0^{+\infty} e^{-\frac{n_1|a_{x_{1i}}|^2}{2\sigma_i^2}} \cdot |a_{x_{1i}}|^3\, d|a_{x_{1i}}|,$$

using integration by parts, we have following equations

$$\begin{aligned}
\mathbb{E}(|a_{x_{1i}}|^3) &= \frac{\sigma_{x_{1i}}\sqrt{2}}{\sqrt{n_1\pi}} \int_0^{+\infty} e^{-\frac{n_1|a_{x_{1i}}|^2}{2\sigma_{x_{1i}}^2}} \cdot \Big(-\frac{n_1|a_{x_{1i}}|}{\sigma_{x_{1i}}^2}\Big) \cdot (-|a_{x_{1i}}|^2)\, d|a_{x_{1i}}| \\
&= \frac{\sigma_{x_{1i}}\sqrt{2}}{\sqrt{n_1\pi}} \Big(e^{-\frac{n_1|a_{x_{1i}}|^2}{2\sigma_{x_{1i}}^2}} \Big|_0^{+\infty} + 2\int_0^{+\infty} e^{-\frac{n_1|a_{x_{1i}}|^2}{2\sigma_{x_{1i}}^2}} \cdot |a_{x_{1i}}|\, d|a_{x_{1i}}| \Big),
\end{aligned}$$

thus,

$$\mathbb{E}(|a_{x_{1i}}|^3) = \frac{\sigma_{x_{1i}} 2\sqrt{2}}{n_1\sqrt{n_1\pi}}(-\sigma_{x_{1i}}^2) \cdot e^{-\frac{n_1|a_{x_{1i}}|^2}{2\sigma_i^2}} \Big|_0^{+\infty} = \frac{2\sqrt{2}}{n_1\sqrt{n_1\pi}}\sigma_{x_{1i}}^3.$$

Since

$$\mathbb{E}\Big(\big(|a_{x_{1i}}|-\mathbb{E}(|a_{x_{1i}}|)\big)^3\Big)=\mathbb{E}(|a_{x_{1i}}|^3)+4\mathbb{E}(|a_{x_{1i}}|)^3-3\mathbb{E}(|a_{x_{1i}}|)\mathbb{E}(|a_{x_{1i}}|^2),$$

we have

$$\mathbb{E}\Big(\big(|a_{x_{1i}}|-\mathbb{E}(|a_{x_{1i}}|)\big)^3\Big)=\frac{4-\pi}{(n_1\pi)^{\frac{3}{2}}}\sigma_{x_{1i}}^3.$$

Because $\sigma_{x_{1i}}/\sqrt{n_1}$ is finite, it is clear that

$$\frac{\sum_{i=1}^{n_1}\mathbb{E}\Big(\big(|a_{x_{1i}}|-\mathbb{E}(|a_{x_{1i}}|)\big)^3\Big)}{s_n^3}\propto\mathcal{O}(\frac{1}{\sqrt{n_1}}).$$

So, the condition in (A.11) is verified, which means

$$\frac{1}{s_n}\sum_{j=1}^{n}(a_{x_{1i}}-\mathbb{E}(a_{x_{1i}}))\xrightarrow{\mathcal{D}}\mathcal{N}(0,1). \tag{A.13}$$

Combining (A.11), (A.12) and (A.13), we obtain the result. ■

A.2 Appendix of Chapter 4

Appendix A.2.2 gives the main results under some assumptions about the kernel parameterization, using intermediate results about uniform convergence of our estimators in Appendix A.2.3. Appendix A.2.4 then shows that these assumptions hold for different settings of kernel learning. Please note that proofs of all results can be found in our previous paper [Liu et al., 2020a].

A.2.1 Preliminaries

Given a kernel k_ω and sample sets $\{X_i\}_{i=1}^n\sim\mathbb{P}^n$, $\{Y_i\}_{i=1}^n\sim\mathbb{Q}^n$, define the $n\times n$ matrix

$$H_{ij}^{(\omega)}=k_\omega(X_i,X_j)+k_\omega(Y_i,Y_j)-k_\omega(X_i,Y_j)-k_\omega(X_j,Y_i);$$

we will often omit ω when it is clear from context. The U-statistic estimator of the squared MMD (4.2) is

$$\hat{\eta}_\omega = \frac{1}{n(n-1)} \sum_{i \neq j} H_{ij}.$$

The squared MMD is $\eta_\omega = \mathbb{E}[H_{12}]$. The variance of $\hat{\eta}_\omega$ is given by Lemma A.1.

Lemma A.1 ([Liu et al., 2020a]). *For a fixed kernel k_ω and random sample sets $\{X_i\}_{i=1}^n$, $\{Y_i\}_{i=1}^n$, we have*

$$\text{Var}[\hat{\eta}_\omega] = \frac{4(n-2)}{n(n-1)} \xi_1^{(\omega)} + \frac{2}{n(n-1)} \xi_2^{(\omega)} = \frac{4}{n} \xi_1^{(\omega)} + \frac{2\xi_2^{(\omega)} - 4\xi_1^{(\omega)}}{n(n-1)}, \tag{A.14}$$

where

$$\xi_1^{(\omega)} = \mathbb{E}\left[H_{12}^{(\omega)} H_{13}^{(\omega)}\right] - \mathbb{E}\left[H_{12}^{(\omega)}\right]^2, \qquad \xi_2^{(\omega)} = \mathbb{E}\left[\left(H_{12}^{(\omega)}\right)^2\right] - \mathbb{E}\left[H_{12}^{(\omega)}\right]^2.$$

Thus as $n \to \infty$,

$$n \,\text{Var}[\hat{\eta}_\omega] \to 4\xi_1^{(\omega)} =: \sigma_\omega^2.$$

We use a V-statistic estimator (4.5) for σ_ω^2:

$$\hat{\sigma}_\omega^2 = 4\left(\frac{1}{n}\sum_{i=1}^n \left(\frac{1}{n}\sum_{j=1}^n H_{ij}^{(\omega)}\right)^2 - \left(\frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n H_{ij}^{(\omega)}\right)^2\right).$$

As a V-statistic, $\hat{\sigma}_\omega^2$ is biased. In fact, Sutherland et al. [2017] and Sutherland [2019] provide an unbiased estimator of $\text{Var}[\hat{\eta}_\omega]$ – including the terms of order $\frac{1}{n(n-1)}$. Although this estimator takes the same quadratic time to compute as (4.5), it contains many more terms, which are cumbersome both for implementation and for analysis. (4.5) is also marginally more convenient in that it is always at least nonnegative. As we show in Lemma A.3, the amount of bias is negligible as n increases. In practice, we expect the difference to be unimportant – or

the V-statistic may in fact be beneficial, since underestimating σ^2 harms the estimate of η/σ^2 more than overestimating it does.

Similarly, although we use the U-statistic estimator (4.2), it would be very similar to use the biased estimator $n^{-2}\sum_{ij}H_{ij}$, or the minimum variance unbiased estimator $n^{-1}(n-1)^{-1}\sum_{i\neq j}(k(X_i,X_j)+k(Y_i,Y_j))-2n^{-2}\sum_{ij}k(X_i,Y_J)$ Showing comparable concentration behavior to Proposition A.1 is trivially different, and in fact it is also not difficult to show σ_ω^2 is the same for all three estimators (up to lower-order terms).

A.2.2 Main Results

We will require the following assumptions. These are fairly agnostic as to the kernel form; Appendix A.2.4.2 shows that these assumptions hold (and gives the constants) for the kernels (4.1) we use in the paper.

(**A**) The kernels k_ω are uniformly bounded:

$$\sup_{\omega\in\Omega}\sup_{x\in\mathcal{X}} k_\omega(x,x)\leq \nu.$$

For the kernels we use in practice, $\nu=1$.

(**B**) The possible kernel parameters ω lie in a Banach space of dimension D. Furthermore, the set of possible kernel parameters Ω is bounded by R_ω, $\Omega\subseteq\{\omega \mid \|\omega\|\leq R_\Omega\}$.

Appendix A.2.4.2 builds this space and its norm for the kernels we use in the paper.

(**C**) The kernel parameterization is Lipschitz: for all $x,y\in\mathcal{X}$ and $\omega,\omega'\in\Omega$,

$$|k_\omega(x,y)-k_{\omega'}(x,y)|\leq L_k\left\|\omega-\omega'\right\|.$$

Proposition A.4 in Appendix A.2.4.2 gives an expression for L_k for the kernels we use in the paper.

We will first show the main results under these general assumptions, using uniform convergence results shown in Appendix A.2.3, then show Assumptions (**B**) and (**C**) for particular kernels in Appendix A.2.4.2.

Theorem A.2 ([Liu et al., 2020a]). *Under Assumptions (**A**) to (**C**), let $\bar{\Omega}_s \subseteq \Omega$ be the set of kernel parameters for which $\sigma_\omega^2 \geq s^2$, and assume $\nu \geq 1$. Take $\lambda = n^{-1/3}$. Then, with probability at least $1-\delta$,*

$$\sup_{\omega \in \bar{\Omega}_s} \left| \frac{\hat{\eta}_\omega}{\hat{\sigma}_{\omega,\lambda}} - \frac{\eta_\omega}{\sigma_\omega} \right| \leq \frac{2\nu}{s^2 n^{1/3}} \left(\frac{1}{s} + \frac{2304\nu^2}{\sqrt{n}} + \left[\frac{4s}{n^{1/6}} + 1024\nu \right] \left[L_k + \sqrt{2\log\frac{2}{\delta} + 2D\log\left(4R_\Omega\sqrt{n}\right)} \right] \right),$$

and thus, treating ν as a constant,

$$\sup_{\omega \in \bar{\Omega}_s} \left| \frac{\hat{\eta}_\omega}{\hat{\sigma}_{\omega,\lambda}} - \frac{\eta_\omega}{\sigma_\omega} \right| = \tilde{O}_P\left(\frac{1}{s^2 n^{1/3}} \left[\frac{1}{s} + L_k + \sqrt{D} \right] \right).$$

It is worth noting that, if we are particularly concerned about the s dependence, we can make some slightly different choices in the decomposition to improve the dependence on s while worsening the rate with n.

Corollary A.2 ([Liu et al., 2020a]). *In the setup of Theorem A.2, additionally assume that there is a unique population maximizer ω^* of J from* (4.3)*, i.e. for each $t > 0$ we have*

$$\sup_{\omega \in \bar{\Omega}_s : \|\omega - \omega^*\| \geq t} J(\mathbb{P}, \mathbb{Q}; k_\omega) < J(\mathbb{P}, \mathbb{Q}; k_{\omega^*}).$$

For each n, let $S_{\mathbb{P}}^{(n)}$ and $S_{\mathbb{Q}}^{(n)}$ be sequences of sample sets of size n, let $\hat{J}_n(\omega)$ denote $J_{\lambda=n^{-1/3}}(S_{\mathbb{P}}^{(n)}, S_{\mathbb{Q}}^{(n)}; k_\omega)$, and take $\hat{\omega}_n^*$ to be a maximizer of $\hat{J}_n(\omega)$.[1] *Then $\hat{\omega}_n^*$ converges in probability to ω^*.*

Corollary A.3 ([Liu et al., 2020a])**.** *In the setup of Theorem A.2, suppose we use n sample points to select a kernel $\hat{\omega}_n \in \arg\max_{\omega \in \bar{\Omega}_s} \hat{J}_\lambda(\omega)$ and m sample points to run a test of level α. Let $r_{\hat{\omega}_n}^{(m)}$ denote the rejection threshold for a test with that kernel of size m. Define $J^* := \sup_{\omega \in \bar{\Omega}_s} J(\omega)$, and constants C, C', C'', N_0 depending on ν, L_k, D, R_Ω and s. For any $n \geq N_0$, with probability at least $1-\delta$, this test procedure has power*

$$\Pr\left(m\hat{\eta}_{\hat{\omega}_n} > r_{\hat{\omega}_n}^{(m)}\right) \geq \Phi\left(\sqrt{m}J^* - C\frac{\sqrt{m}}{n^{\frac{1}{3}}}\sqrt{\log\frac{n}{\delta}} - C'\sqrt{\log\frac{1}{\alpha}}\right) - \frac{C''}{\sqrt{m}}.$$

Corollary A.4 ([Liu et al., 2020a])**.** *In the setup of Corollary A.3, suppose we are given N data points to divide between n training points and $m = N - n$ testing points. Ignoring the Berry-Esseen convergence term outside of Φ, the asymptotic power upper bound*

$$\Phi\left(\sqrt{m}J^* - C\frac{\sqrt{m}}{n^{\frac{1}{3}}}\sqrt{\log\frac{n}{\delta}} - C'\sqrt{\log\frac{1}{\alpha}}\right)$$

is maximized by choosing, as $N \to \infty$ and other quantities remain constant,

$$n \sim \left(\frac{C}{J^*}N\sqrt{\frac{1}{3}\log N}\right)^{\frac{3}{4}}.$$

A.2.3 Uniform Convergence Results

These results, on the uniform convergence of $\hat{\eta}$ and $\hat{\sigma}^2$, were used in the proof of Theorem A.2 [Liu et al., 2020a].

[1] In fact, it suffices for the $\hat{\omega}_n^*$ to only approximately maximize $\hat{J}_n$, as long as their suboptimality is $o_P(1)$.

Proposition A.1 ([Liu et al., 2020a]). *Under Assumptions (**A**) to (**C**), we have that with probability at least* $1-\delta$,

$$\sup_{\omega} \left|\hat{\eta}_\omega - \eta_\omega\right| \leq \frac{8}{\sqrt{n}} \left[\nu \sqrt{2\log\frac{2}{\delta} + 2D\log\left(4R_\Omega\sqrt{n}\right)} + L_k \right].$$

Proposition A.2 ([Liu et al., 2020a]). *Under Assumptions (**A**) to (**C**), with probability at least* $1-\delta$,

$$\sup_{\omega\in\Omega} \left|\hat{\sigma}^2_\omega - \sigma^2_\omega\right| \leq \frac{64}{\sqrt{n}} \left[7\sqrt{2\log\frac{2}{\delta} + 2D\log\left(4R_\Omega\sqrt{n}\right)} + \frac{18\nu^2}{\sqrt{n}} + 8L_k\nu \right].$$

Lemma A.2 ([Liu et al., 2020a]). *For any kernel k bounded by ν (Assumption (**A**)), with probability at least* $1-\delta$,

$$\left|\hat{\sigma}^2_k - \mathbb{E}\hat{\sigma}^2_k\right| \leq 448\sqrt{\frac{2}{n}\log\frac{2}{\delta}}.$$

Lemma A.3 ([Liu et al., 2020a]). *For any kernel k bounded by ν (Assumption (**A**)), the estimator $\hat{\sigma}^2_k$ satisfies*

$$\left|\mathbb{E}\hat{\sigma}^2_k - \sigma^2_k\right| \leq \frac{1152\nu^2}{n}.$$

Lemma A.4 ([Liu et al., 2020a]). *Under Assumptions (**A**) and (**C**), we have*

$$\sup_{\omega,\omega'\in\Omega} \frac{\left|\hat{\sigma}^2_\omega - \hat{\sigma}^2_{\omega'}\right|}{\|\omega-\omega'\|} \leq 256L_k\nu \qquad \textit{and} \qquad \sup_{\omega,\omega'\in\Omega} \frac{\left|\sigma^2_\omega - \sigma^2_{\omega'}\right|}{\|\omega-\omega'\|} \leq 256L_k\nu.$$

A.2.4 Constructing Appropriate Kernels

We now show that Assumption (**C**) is satisfied by various choices of kernel: Gaussian bandwidth selection (Appendix A.2.4.1), deep kernels (Appendix A.2.4.2), and classic multiple kernel learning (Appendix A.2.4.3). The following assumption will be useful for different kernel schemes.

(I) The domain $\mathcal{X}$ is Euclidean and bounded, $\mathcal{X} \subseteq \{x \in \mathbb{R}^d : \|x\| \leq R_X\}$ for some constant $R_X < \infty$.

We begin by recalling a well-known property of the Gaussian kernel, useful for both Gaussian bandwidth selection and deep kernels. A proof is in our previous paper [Liu et al., 2020a].

Lemma A.5. *The Gaussian kernel* $\kappa(a,b) = \exp\left(-\frac{\|a-b\|^2}{2\sigma^2}\right)$ *satisfies*

$$\left|\kappa(a,b) - \kappa(a',b')\right| \leq \frac{1}{\sigma\sqrt{e}}\left(\|a-b\| + \left\|a'-b'\right\|\right) \leq \frac{1}{\sigma\sqrt{e}}\left(\left\|a-a'\right\| + \left\|b-b'\right\|\right).$$

A.2.4.1 Gaussian bandwidth selection

Lemma A.5 immediately gives us Assumption (**C**) when we chose among Gaussian kernels:

Proposition A.3 ([Liu et al., 2020a])**.** *Define a one-dimensional Banach space for inverse lengthscales of Gaussian kernels* $\gamma > 0$*, so that* $k_\gamma(x,y) = \kappa_{1/\gamma}(x,y)$*, with standard addition and multiplication and norms defined by the absolute value, and* k_0 *taken to be the constant* 1 *function. Let* Ω *be any subset of this space. Under Assumption (**I**), Assumption (**C**) holds: for any* $x, y \in \mathcal{X}$ *and* $\gamma, \gamma' \in \Gamma$*,*

$$\left|k_\gamma(x,y) - k_{\gamma'}(x,y)\right| \leq \frac{2R_X}{\sqrt{e}}\left|\gamma - \gamma'\right|.$$

A.2.4.2 Deep kernels

To handle the deep kernel case, we will need some more assumptions on the form of the kernel.

(II) $\phi_\omega(x) = \phi_\omega^{(\Lambda)}$ is a feedforward neural network with Λ layers given by

$$\phi_\omega^{(0)}(x) = x \qquad \phi_\omega^{(\ell)}(x) = \sigma^{(\ell)}\left(W_\omega^{(\ell)}\phi_\omega^{(\ell-1)}(x) + b_\omega^{(\ell)}\right),$$

where the network parameter ω consists of all the weight matrices $W_\omega^{(\ell)}$ and biases $b_\omega^{(\ell)}$, and the activation functions $\sigma^{(\ell)}$ are each 1-Lipschitz, $\left\|\sigma^{(\ell)}(x) - \sigma^{(\ell)}(y)\right\| \le \|x - y\|$, with $\sigma^{(\ell)}(0) = 0$ so that $\left\|\sigma^{(\ell)}(x)\right\| \le \|x\|$. Define a Banach space on ω, with addition and scalar multiplication componentwise, and

$$\|\omega\| = \max_{\ell \in \{1,\dots,\Lambda\}} \max\left(\left\|W_\omega^{(\ell)}\right\|, \left\|b_\omega^{(\ell)}\right\|\right),$$

where the matrix norm denotes operator norm $\|W\| = \sup_x \|Wx\|/\|x\|$. (For convolutional networks, see Remark A.2.)

(III) k_ω is a kernel of the form (4.1),

$$k_\omega(x,y) = \left[(1-\epsilon)\kappa(\phi_\omega(x), \phi_\omega(y)) + \epsilon\right] q(x,y),$$

with $0 \le \epsilon \le 1$, κ a kernel function, and $q(x,y)$ a kernel with $\sup_x q(x,x) \le Q$. Note that this includes kernels of the form $k_\omega(x,y) = \kappa(\phi_\omega(x), \phi_\omega(y))$: take $\epsilon = 0$ and $q(x,y) = 1$.

(IV) κ in Assumption (**III**) is a kernel function satisfying

$$\left|\kappa(a,b) - \kappa(a',b')\right| \le L_\kappa\left(\left\|a - a'\right\| + \left\|b - b'\right\|\right).$$

This holds for a Gaussian κ via Lemma A.5.

We now turn to proving Assumption (**C**) for deep kernels. First, we will need some smoothness properties of the network ϕ.

Lemma A.6. *Under Assumption (**II**), suppose ω, ω' have $\|\omega\| \le R$, $\|\omega'\| \le R$, with $R \neq 1$. Then, for any x,*

$$\|\phi_\omega(x)\| \le R^\Lambda \|x\| + \frac{R}{R-1}(R^\Lambda - 1) \tag{A.15}$$

$$\|\phi_\omega(x) - \phi_{\omega'}(x)\| \le \left(\Lambda R^{\Lambda-1}\left(\|x\| + \frac{R}{R-1}\right) - \frac{R^\Lambda - 1}{(R-1)^2}\right)\|\omega - \omega'\|. \tag{A.16}$$

If $R \ge 2$, we furthermore have

$$\|\phi_\omega(x)\| \le R^\Lambda(\|x\| + 2) \tag{A.17}$$

$$\|\phi_\omega(x) - \phi_{\omega'}(x)\| \le \Lambda R^{\Lambda-1}(\|x\| + 2)\|\omega - \omega'\|. \tag{A.18}$$

The proof, by recursion, is given in our previous paper [Liu et al., 2020a]. We are now ready to prove Assumption (**C**) for deep kernels.

Proposition A.4 ([Liu et al., 2020a])**.** *Make Assumptions (**I**) to (**IV**) and Assumption (**B**), with $R \ge 2$.*[2] *Then Assumption (**C**) holds: for any $x, y \in \mathcal{X}$ and $\omega, \omega' \in \Omega$,*

$$|k_\omega(x,y) - k_{\omega'}(x,y)| \le 2Q(1-\epsilon)L_\kappa \Lambda R_\Omega^{\Lambda-1}(R_X + 2)\|\omega - \omega'\|.$$

Remark A.1. *For the deep kernels we use in the paper (Assumptions (**II**) to (**IV**)) on bounded domains (Assumption (**I**)), we know L_k via Proposition A.4; Theorem 4.1 combines Theorem A.2, Corollary A.2, and Proposition A.4. If we further use a Gaussian kernel q of bandwidth σ_ϕ, the last bracketed term in the error bound of Theorem A.2 becomes*

$$\frac{2(1-\epsilon)}{\sigma_\phi\sqrt{e}}\Lambda R_\Omega^{\Lambda-1}(R_X+2) + \sqrt{2\log\frac{2}{\delta} + 2D\log\left(4R_\Omega\sqrt{n}\right)}.$$

[2] Of course, if we know a bound of $R < 2$, the result will still hold using $R = 2$. It is also possible to show a tighter result, via (A.15) and (A.16) or their analogue for $R = 1$; the expression is simply less compact.

The component $R_\Omega^{\Lambda-1}(R_X+2)$, from (A.17)*, is approximately the largest that ϕ_ω could make its outputs' norms; σ_ϕ will generally be on a comparable scale to the norm of the actual outputs of the network, so their ratio is something like the "unused capacity" of the network to blow up its inputs. This term is weighted about equally in the convergence bound with the square root of the total number of parameters in the network.*

Remark A.2. *We can handle convolutional networks as follows. We define Ω in essentially the same way, letting $W_\omega^{(\ell)}$ denote the convolutional kernel (the set of parameters being optimized), but define $\|\omega\|$ in terms of the operator norm of the linear transform corresponding to the convolution operator. This is given in terms of the operator norm of various discrete Fourier transforms of the kernel matrix by Lemma 2 of Bibi et al. [2019]; see also Theorem 6 of Sedghi et al. [2019]. The number of parameters D is then the actual number of parameters optimized in gradient descent, but the radius R_Ω is computed differently.*

A.2.4.3 Multiple kernel learning

Multiple kernel learning [Gönen and Alpaydın, 2011] also falls into our setting. A special case of this family of kernels was studied for the (easier to analyze) "streaming" MMD estimator by Gretton et al. [2012].

(V) Let $\{k_i\}_{i=1}^D$ be a set of base kernels, each satisfying $\sup_{x\in\mathcal{X}} k_i(x,x) \le K$ for some finite K. Define k_ω as

$$k_\omega(x,y) = \sum_{i=1}^{D} \omega_i k_i(x,y).$$

Define the norm of a kernel parameter by the norm of the corresponding vector $\omega \in \mathbb{R}^D$. Let Ω be a set of possible parameters such that for each $\omega \in \Omega$, k_ω is positive semi-definite, and $\|\omega\| \leq R_\Omega$ for some $R_\Omega < \infty$.

Not only does learning in this setting work (Proposition A.5), it is also – unlike the deep setting – efficient to find an exact maximizer of $\hat{J}_\lambda$ (Proposition A.6).

Proposition A.5 ([Liu et al., 2020a])**.** *Assumption (**V**) implies Assumptions (**A**) to (**C**). In particular,*

$$\sup_{\omega\in\Omega}\sup_{x\in\mathcal{X}} k_\omega(x,x) \leq KR_\Omega\sqrt{D}$$
$$|k_\omega(x,y) - k_{\omega'}(x,y)| \leq K\sqrt{D}\left\|\omega-\omega'\right\|.$$

Proposition A.6 ([Liu et al., 2020a])**.** *Take Assumption (**V**), and additionally assume that $\Omega = \{\omega \mid \forall i.\, \omega_i \geq 0, \sum_i \omega_i = Q\}$ for some $Q < \infty$. A maximizer of $\hat{J}_\lambda(\omega)$ can then be found by scaling the solution to a convex quadratic program,*

$$\tilde{\omega} = \operatorname*{arg\,min}_{\omega\in[0,\infty)^D\,:\,\omega^\top \mathbf{b}=1} \omega^\top(\mathbf{A}+\lambda I)\omega, \qquad \hat{\omega} = \frac{Q}{\sum_i \tilde{\omega}_i}\tilde{\omega} \in \operatorname*{arg\,max}_{\omega\in\Omega} \hat{J}_\lambda(\omega),$$

where

$$\left(\mathbf{H}_{ij}\right)_\ell = k_\ell(X_i,X_j) + k_\ell(Y_i,Y_j) - k_\ell(X_i,Y_j) - k_\ell(X_j,Y_i)$$
$$\mathbf{b} = \frac{1}{n(n-1)}\sum_{i\neq j}\mathbf{H}_{ij} \in \mathbb{R}^D$$
$$\mathbf{A} = \frac{4}{n^3}\sum_i\left(\sum_j \mathbf{H}_{ij}\right)\left(\sum_j \mathbf{H}_{ij}\right)^\top - \frac{4}{n^4}\left(\sum_{ij}\mathbf{H}_{ij}\right)\left(\sum_{ij}\mathbf{H}_{ij}\right)^\top \in \mathbb{R}^{D\times D},$$

as long as $\mathbf{b}$ *has at least one positive entry.*

A.2.5 Miscellaneous Results

The following lemma [Liu et al., 2020a] was used for Propositions A.3 and A.4.

Lemma A.5. *The Gaussian kernel* $\kappa(a,b) = \exp\left(-\frac{\|a-b\|^2}{2\sigma^2}\right)$ *satisfies*

$$\left|\kappa(a,b) - \kappa(a',b')\right| \leq \frac{1}{\sigma\sqrt{e}}\left(\|a-b\| + \left\|a'-b'\right\|\right) \leq \frac{1}{\sigma\sqrt{e}}\left(\left\|a-a'\right\| + \left\|b-b'\right\|\right).$$

This next lemma [Liu et al., 2020a] was used in Proposition A.4.

Lemma A.6. *Under Assumption (**II**), suppose* ω, ω' *have* $\|\omega\| \leq R$, $\left\|\omega'\right\| \leq R$, *with* $R \neq 1$. *Then, for any x,*

$$\left\|\phi_\omega(x)\right\| \leq R^\Lambda \|x\| + \frac{R}{R-1}(R^\Lambda - 1) \tag{A.15}$$

$$\left\|\phi_\omega(x) - \phi_{\omega'}(x)\right\| \leq \left(\Lambda R^{\Lambda-1}\left(\|x\| + \frac{R}{R-1}\right) - \frac{R^\Lambda - 1}{(R-1)^2}\right)\left\|\omega - \omega'\right\|. \tag{A.16}$$

If $R \geq 2$, *we furthermore have*

$$\left\|\phi_\omega(x)\right\| \leq R^\Lambda(\|x\| + 2) \tag{A.17}$$

$$\left\|\phi_\omega(x) - \phi_{\omega'}(x)\right\| \leq \Lambda R^{\Lambda-1}(\|x\| + 2)\left\|\omega - \omega'\right\|. \tag{A.18}$$

A.3 Appendix of Chapter 5

A.3.1 Review of Generation of Noisy Labels

This section presents a review on two label corruption processes.

A.3.1.1 Transition matrix

We assume that there is a clean *multivariate random variable* (m.r.v.) (X_s, Y_s) defined on $\mathcal{X} \times \mathcal{Y}$ with a probability density $p_s(x_s, y_s)$, where $\mathcal{Y} = \{1, ..., K\}$ is a label set with K labels. However, samples of (X_s, Y_s) cannot be directly obtained and we only can observe noisy source data from the m.r.v. $(X_s, \tilde{Y}_s)$ defined on $\mathcal{X} \times \mathcal{Y}$ with a probability density $\tilde{p}_s(x_s, \tilde{y}_s)$. $\tilde{p}_s(x_s, \tilde{y}_s)$ is generated by a transition

probability $\Pr(\tilde{Y}_s = j|Y_s = i)$, i.e., the flip rate from a clean label i to a noisy label j. When we generate $\tilde{p}_s(x_s, \tilde{y}_s)$ using Q, we often assume that $\sum_{y_s=1}^{K} p_s(x_s, y_s) = \sum_{\tilde{y}_s=1}^{K} \tilde{p}_s(x_s, \tilde{y}_s)$, i.e., the class conditional noise Liu and Tao [2016]; Van Rooyen et al. [2015]; Xia et al. [2019]. All these transition probabilities are summarized into a transition matrix Q, where $Q_{ij} = \Pr(\tilde{Y}_s = j|Y_s = i)$.

The transition matrix Q is easily estimated in certain situations Liu and Tao [2016]. However, in more complex situations, such as clothing1M dataset Xiao et al. [2015], noisy data is directly generated by selecting data from a pool that mixes correct data (data with correct labels) and incorrect data (data with incorrect labels). Namely, how a correct label i is corrupted to j ($i \neq j$) is unclear.

A.3.1.2 Sample selection

Formally, there is a m.r.v. (X_s, Y_s, V_s) defined on $\mathcal{X} \times \mathcal{Y} \times \mathcal{V}$ with a probability density $p_s^{\text{po}}(x_s, y_s, v_s)$, where $\mathcal{V} = \{0, 1\}$ and $V_s = 1$ means "correct" and $V_s = 0$ means "incorrect". Nonetheless, samples from (X_s, Y_s, V_s) cannot be obtained and we can only observe $(X_s, \tilde{Y}_s)$ from a distribution with the following probability density.

$$\tilde{p}_s(x_s, \tilde{y}_s) = \sum_{v_s=0}^{1} p_{X_s, Y_s|V_s}^{\text{po}}(x_s, y_s|v_s) p_{V_s}^{\text{po}}(v_s), \tag{A.19}$$

where $p_{V_s}^{\text{po}}(v_s) = \int_{\mathcal{X}} \sum_{y_s=1}^{K} p_s^{\text{po}}(x_s, y_s, v_s) dx_s$. The density in Eq. (A.19) means that we lost the information from V_s. If we uniformly select samples drawn from $\tilde{p}_s(x_s, \tilde{y}_s)$, the noise rate of these samples is $p_{V_s}^{\text{po}}(0)$. It is clear that the m.r.v. $(X_s, Y_s|V_s = 1)$ is the clean m.r.v. (X_s, Y_s) defined in Appendix A.3.1.1. Then, $q_s(x_s, y_s)$ is used to describe the probability density of incorrect m.r.v. $(X_s, Y_s|V_s = 0)$.

Using $p_s(x_s, y_s)$ and $q_s(x_s, y_s)$, $\tilde{p}_s(x_s, \tilde{y}_s)$ can be expressed by the following equation.

$$\tilde{p}_s(x_s, \tilde{y}_s) = (1-\rho)p_s(x_s, y_s) + \rho q_s(x_s, y_s), \tag{A.20}$$

where $\rho = p^{\text{po}}_{V_s}(v_s = 0)$. Here, we do not assume $\sum_{y_s=1}^{K} p_s(x_s, y_s) = \sum_{y_s=1}^{K} q_s(x_s, y_s)$. To reduce noise effects from incorrect data, scholars aim to recover the information of V_s, i.e., to select correct data Han et al. [2018]; Jiang et al. [2018]; Malach and Shalev-Shwartz [2017].

A.3.2 Proofs

A.3.2.1 Proof of Theorem 5.1

Proof. We will fist prove Eq. (5.2) (Case 1) and then prove Eq. (5.3) (Case 2).

Case 1. According to definition of $\tilde{R}_s(h)$, we have

$$\begin{aligned}
\tilde{R}_s(h) &= \mathbb{E}_{\tilde{p}_s(x_s, \tilde{y}_s)}[\ell(h(x_s), \tilde{y}_s)] \\
&= \int_{\mathcal{X}} \sum_{\tilde{y}_s=1}^{K} \ell(h(x_s), \tilde{y}_s)\tilde{p}_s(x_s, \tilde{y}_s)dx_s \\
&= \int_{\mathcal{X}} \sum_{\tilde{y}_s=1}^{K} \ell(h(x_s), \tilde{y}_s)\tilde{p}_{\tilde{Y}_s|X_s}(\tilde{y}_s|x_s)p_{x_s}(x_s)dx_s \\
&= \int_{\mathcal{X}} \tilde{\eta}^T(x_s)\ell(h(x_s))p_{x_s}(x_s)dx_s, \qquad \text{(A.21)}
\end{aligned}$$

where $\ell(h(x_s)) = [\ell(h(x_s), 1), ..., \ell(h(x_s), K)]^T$ and $\tilde{\eta}(x_s) = [\tilde{p}_{\tilde{Y}_s|X_s}(1|x_s), \ldots, \tilde{p}_{\tilde{Y}_s|X_s}(K|x_s)]^T$. According to definition of the transition matrix Q, we know that

$$\tilde{\eta}^T(x_s) = \eta^T(x_s)Q, \tag{A.22}$$

where $\eta(x_s) = [p_{Y_s|X_s}(1|x_s), \dots, p_{Y_s|X_s}(K|x_s)]^T$. Substituting Eq. (A.22) into Eq. (A.21), we have

$$\begin{aligned}
\tilde{R}_s(h) &= \int_{\mathcal{X}} \eta^T(x_s) \boldsymbol{Q} \ell(h(x_s)) p_{x_s}(x_s) dx_s \\
&= \int_{\mathcal{X}} \eta^T(x_s) I \ell(h(x_s)) p_{x_s}(x_s) dx_s \\
&\quad + \int_{\mathcal{X}} \eta^T(x_s) (\boldsymbol{Q} - I) \ell(h(x_s)) p_{x_s}(x_s) dx_s \\
&= R_s(h) + \mathbb{E}_{p_{x_s}(x_s)}[\eta^T(x_s)(\boldsymbol{Q} - I)\ell(h(x_s))].
\end{aligned}$$

Hence, Case 1 is proved.

Case 2. According to definition of $\tilde{R}_s(h)$ and Eq. (A.20), we have

$$\begin{aligned}
\tilde{R}_s(h) &= \mathbb{E}_{\tilde{p}_s(x_s, \tilde{y}_s)}[\ell(h(x_s), \tilde{y}_s)] \\
&= \int_{\mathcal{X}} \sum_{\tilde{y}_s=1}^{K} \ell(h(x_s), \tilde{y}_s) \tilde{p}_s(x_s, \tilde{y}_s) dx_s \\
&= \int_{\mathcal{X}} \sum_{y_s=1}^{K} \ell(h(x_s), y_s) \big((1-\rho) p_s(x_s, y_s) + \rho q_s(x_s, y_s)\big) dx_s \\
&= (1-\rho) \int_{\mathcal{X}} \sum_{y_s=1}^{K} \ell(h(x_s), y_s) p_s(x_s, y_s) dx_s \\
&\quad + \rho \int_{\mathcal{X}} \sum_{y_s=1}^{K} \ell(h(x_s), y_s) q_s(x_s, y_s) dx_s \\
&= (1-\rho) R_s(h) \\
&\quad + \rho \int_{\mathcal{X}} \sum_{y_s=1}^{K} \ell(h(x_s), y_s) q_{Y_s|X_s}(y_s|x_s) q_{x_s}(x_s) dx_s. \qquad \text{(A.23)}
\end{aligned}$$

Let $\eta_{\mathbf{q}}(x_s) = [q_{Y_s|X_s}(1|x_s), \dots, q_{Y_s|X_s}(K|x_s)]^T$, we have

$$\tilde{R}_s(h) = (1-\rho) R_s(h) + \rho \mathbb{E}_{q_{x_s}(x_s)}[{\eta_{\mathbf{q}}}^T(x_s) \ell(h(x_s))].$$

Hence, Case 2 is proved. ■

A.3.2.2 Proof of Theorem 5.2

Proof. For any $h \in \mathcal{H}$, we have

$$\begin{aligned} R_t(h,f_t) &= R_t(h,f_t) + \tilde{R}_s(h) - \tilde{R}_s(h) + R_s(h,f_t) - R_s(h,f_t) \\ &= \tilde{R}_s(h) + R_t(h,f_t) - \tilde{R}_s(h,f_t) + R_s(h,f_t) - R_s(h) \\ &\quad + R_s(h) - \tilde{R}_s(h) + \tilde{R}_s(h,f_t) - R_s(h,f_t). \end{aligned} \tag{A.24}$$

Since we do not know f_t, we substitute following equations into Eq. (A.24),

$$\begin{aligned} R_t(h,f_t) &= R_t(h,\tilde{f}_t) + R_t(h,f_t) - R_t(h,\tilde{f}_t), \\ \tilde{R}_s(h,f_t) &= \tilde{R}_s(h,\tilde{f}_t) + \tilde{R}_s(h,f_t) - \tilde{R}_s(h,\tilde{f}_t), \\ R_s(h,f_t) &= R_s(h,\tilde{f}_t) + R_s(h,f_t) - R_s(h,\tilde{f}_t). \end{aligned}$$

Then, we have

$$\begin{aligned} R_t(h,f_t) &= \tilde{R}_s(h) + R_t(h,\tilde{f}_t) - \tilde{R}_s(h,\tilde{f}_t) + R_s(h,\tilde{f}_t) - R_s(h) \\ &\quad + R_s(h) - \tilde{R}_s(h) + \tilde{R}_s(h,\tilde{f}_t) - R_s(h,\tilde{f}_t) \\ &\quad + R_t(h,f_t) - R_t(h,\tilde{f}_t) \\ &\leq \tilde{R}_s(h) + |R_t(h,\tilde{f}_t) - \tilde{R}_s(h,\tilde{f}_t)| + |R_s(h,\tilde{f}_t) - R_s(h)| \\ &\quad + |\tilde{R}_s(h) - R_s(h)| + |\tilde{R}_s(h,\tilde{f}_t) - R_s(h,\tilde{f}_t)| \\ &\quad + |R_t(h,f_t) - R_t(h,\tilde{f}_t)|. \end{aligned}$$

Hence, this theorem is proved. ■

A.3.2.3 Proof of Theorem 5.3

Proof of Lemma 5.1 According to definition of $\tilde{R}_s^{\mathrm{po}}(h, u_s)$ in Section 5.5.3, we have

$$
\begin{aligned}
&\tilde{R}_s^{\mathrm{po}}(h, u_s) \\
&= \frac{\int_{\mathcal{X}} \sum_{u_s=0}^{1} \sum_{y_s=1}^{K} u_s \ell(h(x_s), y_s) \tilde{p}_s^{\mathrm{po}}(x_s, y_s, u_s) dx_s}{1 - \rho_{u_s}} \\
&= \frac{\int_{\mathcal{X}} \sum_{y_s=1}^{K} \ell(h(x_s), y_s) \tilde{p}_{X_s, Y_s|U_s}^{\mathrm{po}}(x_s, y_s|1) \tilde{p}_{U_s}^{\mathrm{po}}(1) dx_s}{1 - \rho_{u_s}} \\
&\overset{(a)}{=} \frac{1 - \rho_{u_s}}{1 - \rho_{u_s}} \int_{\mathcal{X}} \sum_{y_s=1}^{K} \ell(h(x_s), y_s) \big(\rho_{01}^s q_s(x_s, y_s) + \rho_{11}^s p_s(x_s, y_s) \big) dx_s \\
&= \rho_{01}^s \mathbb{E}_{q_s(x_s, y_s)}[\ell(h(x_s), y_s)] + \rho_{11}^s R_s(h),
\end{aligned}
$$

where (a) is based on the definition of ρ_{u_s} and Eq. (5.6). Thus, we have

$$
\begin{aligned}
&|\tilde{R}_s^{\mathrm{po}}(h, u_s) - R_s(h)| \\
&= |\rho_{01}^s \mathbb{E}_{q_s(x_s, y_s)}[\ell(h(x_s), y_s)] - (1 - \rho_{11}^s) R_s(h)| \\
&\leq \rho_{01}^s \max\{\mathbb{E}_{q_s(x_s, y_s)}[\ell(h(x_s), y_s)], R_s(h)\}.
\end{aligned}
$$

This lemma is proved.

Proof of Lemma 5.2 According to definition of $\tilde{R}_t^{\text{po}}(h, \tilde{f}_t, u_t)$ in Section 5.5.3, we have

$$
\begin{aligned}
&\tilde{R}_t^{\text{po}}(h, \tilde{f}_t, u_t) \\
&= (1-\rho_{u_t})^{-1} \int_{\mathcal{X}} \sum_{u_t=0}^{1} u_t \ell(h(x_t), \tilde{f}_t(x_t)) \tilde{p}_t^{\text{po}}(x_t, u_t) dx_t \\
&= (1-\rho_{u_t})^{-1} \int_{\mathcal{X}} \ell(h(x_t), \tilde{f}_t(x_t)) \tilde{p}_{X_t|U_t}^{\text{po}}(x_t|1) \tilde{p}_{U_t}^{\text{po}}(1) dx_t \\
&\overset{(a)}{=} \frac{1-\rho_{u_t}}{1-\rho_{u_t}} \int_{\mathcal{X}} \ell(h(x_s), \tilde{f}_t(x_t)) \big(\rho_{01}^t q_{x_t}(x_t) + \rho_{11}^t p_{X_t|V_t}^{\text{po}}(x_t|1)\big) dx_t \\
&= \rho_{01}^t \mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t), \tilde{f}_t(x_t))] \\
&\quad + \rho_{11}^t \int_{\mathcal{X}} \ell(h(x_t), \tilde{f}_t(x_t)) p_{X_t|V_t}^{\text{po}}(x_t|V_t=1) dx_t \\
&\overset{(b)}{=} \rho_{01}^t \mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t), \tilde{f}_t(x_t))] \\
&\quad + \rho_{11}^t \int_{\mathcal{X}} \ell(h(x_t), f_t(x_t)) p_{X_t|V_t}^{\text{po}}(x_t|V_t=1) dx_t \\
&= \rho_{01}^t \mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t), \tilde{f}_t(x_t))] + \rho_{11}^t \int_{\mathcal{X}} \ell(h(x_t), f_t(x_t)) p'_{x_t}(x_t) dx_t \\
&= \rho_{01}^t \mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t), \tilde{f}_t(x_t))] + \rho_{11}^t \mathbb{E}_{p'_{x_t}(x_t)}[\ell(h(x_t), f_t(x_t))], \qquad \text{(A.25)}
\end{aligned}
$$

where (a) is based on the definition of ρ_{u_s} and Eq. (5.6) and (b) is based on the definition of V_t ($f_t(x_t) = \tilde{f}_t(x_t)$ when $V_t = 1$). Since $\mathbb{E}_{p'_{x_t}(x_t)}[\ell(h(x_t), f_t(x_t))] \leq R_t(h, f_t) + \rho_{01}^s M_t$, we have

$$
\begin{aligned}
&\tilde{R}_t^{\text{po}}(h, \tilde{f}_t, u_t) \\
&\leq \rho_{01}^t \mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t), \tilde{f}_t(x_t))] + \rho_{11}^t (R_t(h, f_t) + \rho_{01}^s M_t). \qquad \text{(A.26)}
\end{aligned}
$$

Thus, we have

$$
\begin{aligned}
&|\tilde{R}_t^{\mathrm{po}}(h,\tilde{f}_t,u_t)-R_t(h,f_t)| \\
&=|\rho_{01}^t\mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t),\tilde{f}_t(x_t))]+\rho_{11}^t\mathbb{E}_{p'_{x_t}(x_t)}[\ell(h(x_t),f_t(x_t))] \\
&\quad -R_t(h,f_t)| \\
&\le|\rho_{01}^t\mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t),\tilde{f}_t(x_t))]+\rho_{11}^t(R_t(h,f_t)+\rho_{01}^sM_t) \\
&\quad -R_t(h,f_t)| \\
&=|\rho_{01}^t(\mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t),\tilde{f}_t(x_t))]-R_t(h,f_t))+\rho_{11}^t\rho_{01}^sM_t| \\
&\le\rho_{01}^t\max\{\mathbb{E}_{q_{x_t}(x_t)}[\ell(h(x_t),\tilde{f}_t(x_t))],R_t(h,f_t)\}+\rho_{11}^t\rho_{01}^sM_t.
\end{aligned}
$$

This lemma is proved.

Proof of Theorem 5.3. Now, we prove Theorem 5.3 as follows.

Proof. We first prove upper bounds of $|\tilde{R}_s^{\mathrm{po}}(h,u_s)-R_s(h)|$, $|\tilde{R}_s^{\mathrm{po}}(h,\tilde{f}_t,u_t)-R_s(h,\tilde{f}_t)|$ and $|\tilde{R}_t^{\mathrm{po}}(h,\tilde{f}_t,u_t)-R_t(h,f_t)|$ under assumptions in Theorem 5.3.

Based on Lemma 5.1, we have

$$
\begin{aligned}
&|\tilde{R}_s^{\mathrm{po}}(h,u_s)-R_s(h)| \\
&=|\rho_{01}^s\mathbb{E}_{q_s(x_s,y_s)}[\ell(h(x_s),y_s)]-(1-\rho_{11}^s)R_s(h)| \\
&\le|\rho_{01}^s(R_s(h)+M_s)-\rho_{01}^sR_s(h)| \\
&=\rho_{01}^sM_s. \qquad \text{(A.27)}
\end{aligned}
$$

Similar, we have

$$
|\tilde{R}_s^{\mathrm{po}}(h,\tilde{f}_t,u_s)-R_s(h,\tilde{f}_t)|\le\rho_{01}^tM_t, \qquad \text{(A.28)}
$$

$$
|\tilde{R}_t^{\mathrm{po}}(h,\tilde{f}_t,u_t)-R_t(h,f_t)|\le\rho_{01}^tM_t+\rho_{11}^t\rho_{01}^sM_t. \qquad \text{(A.29)}
$$

Since M_s and M_t are positive constants, it is clear that $\tilde{R}_s^{\text{po}}(h,u_s) \to R_s(h)$, $\tilde{R}_s^{\text{po}}(h,\tilde{f}_t,u_s) \to R_s(h,\tilde{f}_t)$ and $\tilde{R}_t^{\text{po}}(h,\tilde{f}_t,u_t) \to R_t(h,f_t)$ when $\rho_{01}^s \to 0$ and $\rho_{01}^t \to 0$.

Specifically, $\forall \epsilon \in (0,1)$, let $\delta_t = \epsilon/M_t$ and $\delta_s = \epsilon/\max\{M_s, \rho_{11}^t M_t\}$. When $\rho_{01}^s < \delta_s$ and $\rho_{01}^t < \delta_t$, we have

$$|\tilde{R}_s^{\text{po}}(h,u_s) - R_s(h)| + |\tilde{R}_s^{\text{po}}(h,\tilde{f}_t,u_s) - R_s(h,\tilde{f}_t)| < 2\epsilon \tag{A.30}$$

$$|\tilde{R}_t^{\text{po}}(h,\tilde{f}_t,u_t) - R_t(h,f_t)| < 2\epsilon. \tag{A.31}$$

Hence, we prove the Eq. (5.11). In following, we give a new upper bound of $R_t(h,f_t)$. Call back to Theorem 5.2, we replace 1) $\tilde{R}_s(h)$ with $\tilde{R}_s^{\text{po}}(h,u_s)$, 2) $\tilde{R}_s(h,\tilde{f}_t)$ with $\tilde{R}_s^{\text{po}}(h,\tilde{f}_t,u_s)$, 3) $R_t(h,\tilde{f}_t)$ with $\tilde{R}_t^{\text{po}}(h,\tilde{f}_t,u_t)$. Then, we have

$$\begin{aligned} R_t(h,f_t) \le \tilde{R}_s^{\text{po}}(h,u_s) &+ |\tilde{R}_t^{\text{po}}(h,\tilde{f}_t,u_t) - \tilde{R}_s^{\text{po}}(h,\tilde{f}_t,u_t))| \\ &+ |R_s(h,\tilde{f}_t) - R_s(h)| + |\tilde{R}_s^{\text{po}}(h,u_s) - R_s(h)| \\ &+ |\tilde{R}_s^{\text{po}}(h,\tilde{f}_t,u_s) - R_s(h,\tilde{f}_t)| \\ &+ |R_t(h,f_t) - \tilde{R}_t^{\text{po}}(h,\tilde{f}_t,u_t)|. \end{aligned} \tag{A.32}$$

Let $\rho_{01}^s \le \delta_s$ and $\rho_{01}^t \le \delta_t$, based on Eqs. (A.30) and (A.31), we have

$$\begin{aligned} R_t(h,f_t) \le & \underbrace{\tilde{R}_s^{\text{po}}(h,u_s)}_{(i)\ \textbf{noisy-data risk}} + \underbrace{|\tilde{R}_t^{\text{po}}(h,\tilde{f}_t,u_t) - \tilde{R}_s^{\text{po}}(h,\tilde{f}_t,u_s)|}_{(ii)\ \textbf{discrepancy between distributions}} \\ & + \underbrace{|R_s(h,\tilde{f}_t) - R_s(h)|}_{(iii)\ \textbf{domain dissimilarity}} + \underbrace{2\epsilon}_{(iv)\ \textbf{noise effects}\ \Delta_s} \\ & + \underbrace{2\epsilon}_{(iv)\ \textbf{noise effects}\ \Delta_t}. \end{aligned}$$

Hence, we prove this theorem. ■

A.3.2.4 Proof of Lemma 5.3

For simplicity, in this proof, we let $\mathcal{L}_{S_s}(\ell,h) = \mathcal{L}(\theta,h;\mathbf{u}_s,D_s^{xy})$, $\tilde{R}_s^{\mathrm{po}}(\ell,h) = \tilde{R}_s^{\mathrm{po}}(h,u_s)$, and $\mathbb{E}_{S_s}[\cdot] = \mathbb{E}_{S_s\sim(\tilde{P}_s^{\mathrm{po}})^n}[\cdot]$, where $\tilde{P}_s^{\mathrm{po}}$ is the probability measure corresponding to the density $\tilde{p}_s^{\mathrm{po}}$. We first show that $\mathcal{L}_{S_s}(\ell,h)$ is an unbiased estimator of $\tilde{R}_s^{\mathrm{po}}(\ell,h)$ based on the definition of $\tilde{R}_s^{\mathrm{po}}(h,u_s)$ in Section 5.5.3. Since $S_s = \{(x_{si},y_{si},u_{si})\}_{i=1}^n$ are i.i.d samples from $\tilde{P}_s^{\mathrm{po}}$, $\mathbb{E}_{S_s}[\mathcal{L}_{S_s}(\ell,h)]$ can be expressed as follows.

$$
\begin{aligned}
&\mathbb{E}_{S_s}\left[\frac{1}{\sum_{i=1}^n u_{si}}\sum_{i=1}^n u_{si}\ell(h(x_{si}),y_{si})\right]\\
&= \int_{\mathcal{X}}\frac{1}{\sum_{i=1}^n u_{si}}\sum_{i=1}^n\sum_{u_{si}=0}^{1}\sum_{y_{si}=1}^{K} u_{si}\ell(h(x_{si}),y_{si})d\tilde{P}_s^{\mathrm{po}}\\
&= \frac{1}{n}\int_{\mathcal{X}}\frac{n}{\sum_{i=1}^n U_i}\sum_{i=1}^n\sum_{u_{si}=0}^{1}\sum_{y_{si}=1}^{K} u_{si}\ell(h(x_{si}),y_{si})d\tilde{P}_s^{\mathrm{po}}\\
&= \frac{1}{n}\int_{\mathcal{X}}(1-\rho_{u_s})^{-1}\sum_{i=1}^n\sum_{u_{si}=0}^{1}\sum_{y_{si}=1}^{K} u_{si}\ell(h(x_{si}),y_{si})d\tilde{P}_s^{\mathrm{po}}\\
&= (1-\rho_{u_s})^{-1}\frac{1}{n}\sum_{i=1}^n\int_{\mathcal{X}}\sum_{u_{si}=0}^{1}\sum_{y_{si}=1}^{K} u_{si}\ell(h(x_{si}),y_{si})d\tilde{P}_s^{\mathrm{po}}\\
&= (1-\rho_{u_s})^{-1}\frac{1}{n}\sum_{i=1}^n\mathbb{E}_{\tilde{p}_s^{\mathrm{po}}(x_s,y_s,u_s)}[u_s\ell(h(x_s),y_s)]\\
&= \tilde{R}_s^{\mathrm{po}}(h,u_s) = \tilde{R}_s^{\mathrm{po}}(\ell,h), \qquad \text{(A.33)}
\end{aligned}
$$

which means that $\mathcal{L}_{S_s}(\ell,h)$ is an unbiased estimator of $\tilde{R}_s^{\mathrm{po}}(\ell,h)$. Then, let $\Phi(S_s) = \sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\left(\tilde{R}_s^{\mathrm{po}}(\ell,h) - \mathcal{L}_{S_s}(\ell,h)\right)$. Changing a point of S_s affects $\Phi(S_s)$ at most $C_L/(n(1-\tau_s))$. Thus, by McDiarmid's inequality applied to $\Phi(S_s)$, for any $\delta > 0$, with probability of at least $1-\delta/2$, the following inequality holds.

$$
\Phi(S_s) \le \mathbb{E}_{S_s}[\Phi(S_s)] + \frac{C_L}{1-\tau_s}\sqrt{\frac{\ln(\delta/2)}{2n}}. \qquad \text{(A.34)}
$$

Then, we have

$$\mathbb{E}_{S_s}[\Phi(S_s)] = \mathbb{E}_{S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\big(\tilde{R}_s^{\mathrm{po}}(\ell,h) - \mathcal{L}_{S_s}(h)\big)\Big]$$

$$= \mathbb{E}_{S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\big(\mathbb{E}_{S'_s}[\mathcal{L}_{S'_s}(\ell,h)] - \mathcal{L}_{S_s}(h)\big)\Big] \tag{A.35}$$

$$= \mathbb{E}_{S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\big(\mathbb{E}_{S'_s}[\mathcal{L}_{S'_s}(\ell,h) - \mathcal{L}_{S_s}(h)]\big)\Big]$$

$$\leq \mathbb{E}_{S_s,S'_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\big(\mathcal{L}_{S'_s}(\ell,h) - \mathcal{L}_{S_s}(h)\big)\Big] \tag{A.36}$$

$$= \frac{1}{n}\mathbb{E}_{S_s,S'_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}\Big(\frac{u'_{si}\ell(h(x'_{si}),y'_{si}) - u_{si}\ell(h(x_{si}),y_{si}))}{1-\tau_s}\Big)\Big]$$

$$= \frac{1}{n}\mathbb{E}_{\sigma,S_s,S'_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}\sigma_i\Big(\frac{u'_{si}\ell(h(x'_{si}),y'_{si}) - u_{si}\ell(h(x_{si}),y_{si}))}{1-\tau_s}\Big)\Big]$$

$$\leq \frac{1}{n(1-\tau_s)}\mathbb{E}_{\sigma,S'_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}\sigma_i u'_{si}\ell(h(x'_{si}),y'_{si})\Big]$$

$$+ \frac{1}{n(1-\tau_s)}\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}-\sigma_i u_{si}\ell(h(x_{si}),y_{si})\Big] \tag{A.37}$$

$$= \frac{2}{n(1-\tau_s)}\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si})\Big], \tag{A.38}$$

where Eq. (A.35) is based on Eq. (A.33), Inequalities (A.36) and (A.37) are based on Jensen's Inequality. Because of existence of u_{si}, Eq. (A.38) is not the Rademacher complexity of $\mathbb{L}_{\mathcal{H}}$ (i.e., $\Re(\mathbb{L}_{\mathcal{H}})$). However, in following, we prove that Eq. (A.38) can be bounded by $\Re(\mathbb{L}_{\mathcal{H}})/(1-\tau_s)$.

$$\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si})\Big]$$

$$= \mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\Big(\sigma_1 u_{s1}\ell(h(x_{s1}),y_{s1}) + \sum_{i=2}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si})\Big)\Big]$$

$$
\begin{aligned}
&= \frac{1}{2}\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell,\ell'\in\mathbb{L}_{\mathcal{H}}}\Big(u_{s1}\ell(h(x_{s1}),y_{s1})+\sum_{i=2}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si}) \\
&\qquad +(-u_{s1})\ell'(h(x_{s1}),y_{s1})+\sum_{i=2}^{n}\sigma_i u_{si}\ell'(h(x_{si}),y_{si})\Big)\Big] \\
&= \frac{1}{2}\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell,\ell'\in\mathbb{L}_{\mathcal{H}}}\Big(u_{s1}(\ell(h(x_{s1}),y_{s1})-\ell'(h(x_{s1}),y_{s1})) \\
&\qquad +\sum_{i=2}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si})+\sum_{i=2}^{n}\sigma_i u_{si}\ell'(h(x_{si}),y_{si})\Big)\Big] \\
&\leq \frac{1}{2}\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell,\ell'\in\mathbb{L}_{\mathcal{H}}}\Big(\ell(h(x_{s1}),y_{s1})-\ell'(h(x_{s1}),y_{s1}) \\
&\qquad +\sum_{i=2}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si})+\sum_{i=2}^{n}\sigma_i u_{si}\ell'(h(x_{si}),y_{si})\Big)\Big] \\
&= \mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\Big(\sigma_1\ell(h(x_{s1}),y_{s1})+\sum_{i=2}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si})\Big)\Big],
\end{aligned}
\tag{A.39}
$$

where Inequality (A.39) is based on the fact that there are always $\ell,\ell'\in\mathbb{L}_{\mathcal{H}}$ such that $\ell(h(x_{s1}),y_{s1})-\ell'(h(x_{s1}),y_{s1})>0$. Repeat above procedures $n-1$ times, we have

$$
\begin{aligned}
&\frac{2}{n}\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}\sigma_i u_{si}\ell(h(x_{si}),y_{si})\Big] \\
&\leq \frac{2}{n}\mathbb{E}_{\sigma,S_s}\Big[\sup_{\ell\in\mathbb{L}_{\mathcal{H}}}\sum_{i=1}^{n}\sigma_i\ell(h(x_{si}),y_{si})\Big] := \Re_n(\mathbb{L}_{\mathcal{H}}).
\end{aligned}
\tag{A.40}
$$

Changing a point of S_s affects $\Re_n(\mathbb{L}_{\mathcal{H}})$ at most $2C_L/n$. Thus, by McDiarmid's inequality, for any $\delta>0$, with probability of at least $1-\delta/2$, the following inequality holds.

$$
\Re_n(\mathbb{L}_{\mathcal{H}})\leq\hat{\Re}_{S_s}(\mathbb{L}_{\mathcal{H}})+2C_L\sqrt{\frac{\ln(\delta/2)}{2n}}.
\tag{A.41}
$$

Since ℓ is Lipschitz continuous, according to Maurer [2016], we have

$$
\hat{\Re}_{S_s}(\mathbb{L}_{\mathcal{H}})\leq\sqrt{2}L_\ell\hat{\Re}_{D_s^x}(\mathcal{H}).
\tag{A.42}
$$

Combining (A.34), (A.38), (A.40), (A.41) and (A.42) we prove this Lemma.

A.3.2.5 Proof of Theorem 5.4

We prove this theorem (i.e., Inequality (5.14)) according to Inequality (A.32), where (5.14) has 8 terms in the right side and (A.32) have 6 terms in the right side.

1) For last 3 terms in (A.32), since $\rho_{01}^{s} < 1/\sqrt{n_s T}$ and $\rho_{01}^{t} < 1/\sqrt{n_t T}$, according to (A.27), (A.28) and (A.29), we know the sum of last three terms of (A.32) is less than or equal to $(M_s + M_t)/\sqrt{n_s T} + 2M_t/\sqrt{n_t T}$ (i.e., the last 2 terms in (5.14)).

2) For first 3 terms in (A.32), we have shown that (in Section 5.5.4) the sum of the first 3 terms in (A.32) is less than or equal to $(*)$:

$$2\tilde{R}_s^{\mathrm{po}}(h, u_s) + 2\tilde{R}_s^{\mathrm{po}}(h, \tilde{f}_t, \mathbf{u}_s) + \tilde{R}_t^{\mathrm{po}}(h, \tilde{f}_t, \mathbf{u}_t) + \frac{M_s}{\sqrt{n_s T}} + \frac{M_t}{\sqrt{n_t T}}.$$

Then, we can prove that (similar with Lemma 5.3), with probability of at least $1-\delta$, for any $h \in \mathcal{H}$,

$$\begin{aligned} \tilde{R}_s^{\mathrm{po}}(h, \tilde{f}_t, \mathbf{u}_s) \leq \mathcal{L}(\theta, h; \mathbf{u}_s, D_{\tilde{s}}^{xy}) + \frac{\sqrt{2} L_\ell \hat{\Re}_{D_s^x}(\mathcal{H})}{1-\tau_s} \\ + \frac{2\sqrt{2} K C_h L_\ell + C_L}{1-\tau_s} \sqrt{\frac{\ln \frac{\delta}{2}}{2n_s}}, \end{aligned} \tag{A.43}$$

$$\begin{aligned} \tilde{R}_t^{\mathrm{po}}(h, \tilde{f}_t, \mathbf{u}_t) \leq \mathcal{L}(\theta, h; \mathbf{u}_t, D_{\tilde{t}}^{xy}) + \frac{\sqrt{2} L_\ell \hat{\Re}_{D_t^x}(\mathcal{H})}{1-\tau_s} \\ + \frac{2\sqrt{2} K C_h L_\ell + C_L}{1-\tau_t} \sqrt{\frac{\ln \frac{\delta}{2}}{2n_t}}. \end{aligned} \tag{A.44}$$

Combining (5.13), (A.43), (A.44) with $(*)$, we get the first 6 terms in (5.14). Hence we obtain all 6 terms in (5.14) and prove this theorem.

A.4 Appendix of Chapter 6

A.4.1 Proof of Theorem 6.1

Proof. For simplicity, we let

$$\rho(\mathbf{Y}=1,\mathbf{X}_s,\mathbf{X}_t)=\frac{\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega))P(\mathbf{X}_s(\omega))}{\beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega))P(\mathbf{X}_t(\omega))},$$

and $\rho_{\mathbf{Y},\mathbf{X}_s,\mathbf{X}_t}$ for short. Based on the Eq. (2), we have

$$\frac{P_{\mathbf{Y},\mathbf{X}_s}(\mathbf{Y}=1,\mathbf{X}_s(\omega))}{P_{\mathbf{X}_s}(\mathbf{X}_s(\omega))}=\rho_{\mathbf{Y},\mathbf{X}_s,\mathbf{X}_t}\frac{P_{\mathbf{Y},\mathbf{X}_t}(\mathbf{Y}=1,\mathbf{X}_t(\omega))}{P_{\mathbf{X}_t}(\mathbf{X}_t(\omega))}$$

Let $\mathbf{Z}_s=f_s(\mathbf{X}_s)$ and $\mathbf{Z}_t=f_t(\mathbf{X}_t)$. Since $f_t^{-1}(f_t(\mathbf{X}_t))=\mathbf{X}_t$ and $f_s^{-1}(f_s(\mathbf{X}_s))=\mathbf{X}_s$, we have

$$P_{\mathbf{Y}=1,\mathbf{Z}_s}(\mathbf{Y}=1,\mathbf{Z}_s)=P_{\mathbf{Y}=1,\mathbf{X}_s}(\mathbf{Y}=1,f_s^{-1}(\mathbf{Z}_s)),$$

$$P_{\mathbf{Y}=1,\mathbf{Z}_t}(\mathbf{Y}=1,\mathbf{Z}_t)=P_{\mathbf{Y}=1,\mathbf{X}_t}(\mathbf{Y}=1,f_t^{-1}(\mathbf{Z}_t)),$$

and

$$P_{\mathbf{Z}_s}(\mathbf{Z}_s)=P_{\mathbf{X}_s}(f_s^{-1}(\mathbf{Z}_s)),\ \ P_{\mathbf{Z}_t}(\mathbf{Z}_t)=P_{\mathbf{X}_t}(f_t^{-1}(\mathbf{Z}_t)),$$

Because $f_s(\mathbf{X}_s)$ is a monotonic map, there must be a 1-1 map between $\mathbf{X}_s$ and $\mathbf{Z}_s$, that is,

$$P_{\mathbf{Y}=1,\mathbf{X}_s}(\mathbf{Y}=1,f_s^{-1}(\mathbf{Z}_s))=P_{\mathbf{Y}=1,\mathbf{X}_s}(\mathbf{Y}=1,\mathbf{X}_s).$$

Hence, we arrive at the following equation.

$$\frac{P_{\mathbf{Y},\mathbf{X}_s}(\mathbf{Y}=1,f_s^{-1}(\mathbf{Z}_s))}{P_{\mathbf{X}_s}(f_s^{-1}(\mathbf{Z}_s))}=\rho_{\mathbf{Y},\mathbf{X}_s,\mathbf{X}_t}\frac{P_{\mathbf{Y},\mathbf{X}_t}(\mathbf{Y}=1,f_t^{-1}(\mathbf{Z}_t))}{P_{\mathbf{X}_t}(f_t^{-1}(\mathbf{Z}_t))}.$$

That is,

$$\frac{P_{\mathbf{Y},\mathbf{Z}_s}(\mathbf{Y}=1,\mathbf{Z}_s)}{P_{\mathbf{Z}_s}(\mathbf{Z}_s)}=\rho_{\mathbf{Y},\mathbf{X}_s,\mathbf{X}_t}\frac{P_{\mathbf{Y},\mathbf{Z}_t}(\mathbf{Y}=1,\mathbf{Z}_t)}{P_{\mathbf{Z}_t}(\mathbf{Z}_t)}.$$

Thus, we have

$$\frac{P(\mathbf{Y}=1|f_s(\mathbf{X}_s(\omega)))}{\beta_s(\mathbf{Y}=1,\mathbf{X}_s(\omega))}=\frac{P(\mathbf{Y}=1|f_t(\mathbf{X}_t(\omega)))}{\beta_t(\mathbf{Y}=1,\mathbf{X}_t(\omega))}=c(\omega),$$

and this theorem is proven. ■

A.4.2 Proof of Lemma 6.1

Proof. $\forall x_1, x_2 \in \mathbb{R}^m$, without loss of generality, we assume $x_1 < x_2$ ($x_{1i} < x_{2i}, i = 1,...,m$). Because $f(x) = xU^T$, we have

$$(f(x_1))_j = \sum_{i=1}^{m} x_{1i}u_{ji}, \;\; (f(x_2))_j = \sum_{i=1}^{m} x_{2i}u_{ji}, j = 1,...,r.$$

So,

$$(f(x_1))_j - (f(x_2))_j = \sum_{i=1}^{m}(x_{1i} - x_{2i})u_{ji}, j = 1,...,r.$$

Because $x_{1i} - x_{2i} < 0$ and x_1 and x_2 are any vector in $\mathbb{R}^m$ satisfying $x_1 < x_2$, $(f(x_1))_j < (f(x_2))_j$ if and only if $u_{ji} > 0$. We can simply prove the $f(x)$ is a decreasing monotonic map if and only if $u_{ji} < 0$. ■

A.4.3 Proof of Theorem 6.2

Proof. Because $f_s(X_s)$ and $f_t(X_t)$ are LMMs, they satisfy the first condition of Theorem 6.1. So we only need to prove $f^{-1}(f(X_s)) = X_s$. According to the Moore-Penrose pseudoinverse of U_s in $f(X_s)$, it is clear that the second condition of Theorem 6.1 can be satisfied. Hence, this theorem is proved. ■

A.4.4 Proof of Theorem 6.3

Proof. Let A, B, C, D, E and F be subspaces in $\mathbb{R}^N$. We need to prove following conditions.

1) $\mathcal{D}((A,B),(C,D)) \geq 0$;

2) $\mathcal{D}((A,B),(C,D)) = \mathcal{D}((C,D),(A,B))$;

3) $\mathcal{D}((A,B),(C,D)) = 0 \Leftrightarrow A^TB = C^TD$;

4) $\mathcal{D}((A,B),(C,D)) \leq \mathcal{D}((A,B),(E,F)) + \mathcal{D}((E,F),(C,D))$.

From Definition 4, it is easy to prove 1) and 2). Based on Definition 3 (principal angles for heterogeneous feature spaces), we know $\sigma_i(A^TB) = \sigma_i(C^TD) \Leftrightarrow A^TB = C^TD$, which means that $\sigma_i(A^TB) - \sigma_i(C^TD) = 0 \Leftrightarrow A^TB = C^TD$. Therefore, 3) is also proven. For 4), we have

$$\begin{aligned}
&\mathcal{D}((A,B),(C,D)) \\
&= \sum_{i=1}^{r} \left| \sigma_i(A^TB) - \sigma_i(E^TF) + \sigma_i(E^TF) - \sigma_i(C^TD) \right| \\
&\leq \sum_{i=1}^{r} \left| \sigma_i(A^TB) - \sigma_i(E^TF) \right| + \sum_{i=1}^{r} \left| \sigma_i(E^TF) - \sigma_i(C^TD) \right| \\
&= \mathcal{D}((A,B),(E,F)) + \mathcal{D}((E,F),(C,D)).
\end{aligned}$$

Thus, condition 4) is proven and $(\mathcal{D}, G_{N,*}^T \times G_{N,*})$ is a metric space. ■

A.4.5 Proof of Theorem 6.4

Proof. Proving this theorem only requires proving that the optimized U_s^* and U_t^* in the GLG model are identical matrixes when $m = n$. In terms of Theorem 3.3, it is evident that $\mathcal{D}((S_{X_s^\delta}, S_{X_t^\delta}),(S_{X_s^\delta}, S_{X_t^\delta})) = 0$. So, if $f_s(X_s) = X_s$ and $f_t(X_t) = X_t$, then we have $J_1 = 0$ (when $m = n, \lambda_s = \lambda_t = 0$), which results in the optimal GLG model.

Because $f_s(X_s) = X_s \Leftrightarrow U_s = I_s$ and $f_t(X_t) = X_t \Leftrightarrow U_t = I_t$, the GLG model degenerates into an ordinary GFK model. ■

A.4.6 Proof of Lemma 6.2

Proof. Let 1) Λ represent the diagonal matrix constructed by λ_i; 2) $(XX^T - \lambda_i I)^- = A(\Lambda - \lambda_i I)^+ A^T$; and 3) $e_i = A^{-1} y_i$.

Hence, we derive the following equations:

$$XX^T A = A\Lambda, \ (\Lambda - \lambda_i I)^- e_i = (\Lambda - \lambda_i I)^- e_i = 0,$$
$$(XX^T - \lambda_i I)^- (XX^T - \lambda_i I)(XX^T - \lambda_i I)^- = (XX^T - \lambda_i I)^-,$$
$$(XX^T - \lambda_i I)(XX^T - \lambda_i I)^- (XX^T - \lambda_i I) = (XX^T - \lambda_i I).$$

Based on these equations and $(XX^T - \lambda_i I)^- = A(\Lambda - \lambda_i I)^+ A^T$, we obtain

$$(XX^T - \lambda_i I)^- (XX^T - \lambda_i I) = I - y_i y_i^T, \tag{A.45}$$

$$(XX^T - \lambda_i I)^- y_i = 0. \tag{A.46}$$

Next, we calculate the first-order derivatives of the EDS. First, we transform Eq. (10) in the paper into the following term.

$$(XX^T - \lambda_i I)\frac{\partial y_i}{\partial X} = y_i \frac{\partial \lambda_i}{\partial X} - \frac{\partial XX^T}{\partial X} y_i. \tag{A.47}$$

Then, we pre-multiply both sides of $(XX^T - \lambda_i I)^-$ and arrive at the following equation based on (A.45).

$$(I - y_i y_i^T)\frac{\partial y_i}{\partial X} = (XX^T - \lambda_i I)^- y_i \frac{\partial \lambda_i}{\partial X} - (XX^T - \lambda_i I)^- \frac{\partial XX^T}{\partial X} y_i.$$

Due to (A.46), we have

$$\frac{\partial y_i}{\partial X} - y_i y_i^T \frac{\partial y_i}{\partial X} = -(XX^T - \lambda_i I)^- \frac{\partial XX^T}{\partial X} y_i. \tag{A.48}$$

Since $y_i^T y_i = 1$, we arrive at

$$\frac{\partial y_i^T}{\partial X} y_i + \frac{\partial y_i}{\partial X} y_i^T = 0 \Rightarrow y_i^T \frac{\partial y_i}{\partial X} = 0. \tag{A.49}$$

Hence, we arrive at the derivatives of the eigenvector.

$$\frac{\partial y_i}{\partial X} = -(XX^T - \lambda_i I)^+ \frac{\partial XX^T}{\partial X} y_i.$$

We only need to pre-multiply both sides of (A.47) with y_i^T to calculate the derivatives of the eigenvalue,

$$\frac{\partial \lambda_i}{\partial X} = y_i^T \frac{\partial XX^T}{\partial X} y_i.$$

This lemma is proven. ■

BIBLIOGRAPHY

A. C. Bianchi, R., Celiberto Jr., L. A., Santos, P. E., Matsuura, J. P. & López de Mántaras, R., 2015, 'Transferring knowledge as heuristics in reinforcement learning: A case-based approach', *Artificial Intelligence*, vol. 226, pp. 102–121.

Alba Fernández, V., Jiménez Gamero, M. & Muñoz García, J., 2008, 'A test for the two-sample problem based on empirical characteristic functions', *Computational Statistics & Data Analysis*, vol. 52, no. 7, pp. 3730–3748.

Arbel, M., Sutherland, D. J., Binkowski, M. & Gretton, A., 2018, 'On gradient regularizers for MMD GANs', *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, Montréal, Canada, 3-8 December, 2018, pp. 6701–6711.

Arjovsky, M., Chintala, S. & Bottou, L., 2017, 'Wasserstein generative adversarial networks', *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, 6-11 August, 2017, pp. 214–223.

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M., Maharaj, T., Fischer, A., Courville, A. & Bengio, Y., 2017, 'A closer look at memorization in deep networks', *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, 6-11 August, 2017, pp. 233–242.

Baktashmotlagh, M., Harandi, M. T., Lovell, B. C. & Salzmann, M., 2013, 'Unsupervised domain adaptation by domain invariant projection', *Proceedings of the 2013 IEEE International Conference on Computer Vision*, Sydney, Australia, 1-8 December, 2013, pp. 769–776.

Baldi, P., Sadowski, P. & Whiteson, D., 2014, 'Searching for exotic particles in high-energy physics with deep learning', *Nature communications*, vol. 5, pp. 4308–4316.

Bartlett, P. L. & Mendelson, S., 2002, 'Rademacher and gaussian complexities: Risk bounds and structural results', *Journal of Machine Learning Research*, vol. 3, pp. 463–482.

Behbood, V., Lu, J., Zhang, G. & Pedrycz, W., 2015, 'Multistep fuzzy bridged refinement domain adaptation algorithm and its application to bank failure prediction', *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 6, pp. 1917–1935.

Behbood, V., Member, S., Lu, J. & Zhang, G., 2014, 'Fuzzy refinement domain adaptation for long term prediction in banking ecosystem', *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 1637–1646.

Ben-David, S. & Blitzer, J., 2006, 'Analysis of representations for domain adaptation', *Proceedings of the 20th Annual Conference on Neural Information Processing Systems*, , vol. 19Vancouver, Canada, 4-7 December, 2006, pp. 137–144.

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

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

Bengio, Y., 2014, 'Evolving culture versus local minima', *Growing Adaptive Machines*, Springer, pp. 109–138.

Bergamo, A. & Torresani, L., 2010, 'Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach', *Proceedings of the 24th Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 6-9 December, 2010, pp. 181–189.

Berlinet, A. & Thomas-Agnan, C., 2004, *Reproducing Kernel Hilbert Spaces in Probability and Statistics*, Kluwer.

Biau, G. & Gyorfi, L., 2005, 'On the asymptotic properties of a nonparametric l1-test statistic of homogeneity', *IEEE Transactions on Information Theory*, vol. 51, no. 11, pp. 3965–3973.

Bibi, A., Ghanem, B., Koltun, V. & Ranftl, R., 2019, 'Deep layers as stochastic solvers', *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, LA, USA, 6-9 May, 2019.

Binkowski, M., Sutherland, D. J., Arbel, M. & Gretton, A., 2018, 'Demystifying MMD GANs', *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 30 April - 3 May, 2018.

Buckley, J. & Eslami, E., 1997a, 'Fuzzy plane geometry I : Points and lines', *Fuzzy Sets and Systems*, vol. 86, pp. 179–187.

Buckley, J. & Eslami, E., 1997b, 'Fuzzy plane geometry II : Circles and polygons', *Fuzzy Sets and Systems*, vol. 87, pp. 79–85.

Callaert, H. & Janssen, P., 1978, 'The Berry-Esseen theorem for u-statistics', *The Annals of Statistics*, vol. 6, no. 2, pp. 417–421.

Cao, Y., Long, M. & Wang, J., 2018, 'Unsupervised domain adaptation with distribution matching machines', *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, 2-7 February, 2018, pp. 2795–2802.

Chakraborty, D. & Ghosh, D., 2014, 'Analytical fuzzy plane geometry II', *Fuzzy Sets and Systems*, vol. 243, pp. 84–109.

Chalmers, E., Contreras, E. B., Robertson, B., Luczak, A. & Gruber, A. J., 2018, 'Learning to predict consequences as a method of knowledge transfer in reinforcement learning', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2259–2270.

Chen, H. & Friedman, J. H., 2017, 'A new graph-based two-sample test for multivariate and object data', *Journal of the American Statistical Association*, vol. 112, no. 517, pp. 397–409.

Cheng, X. & Cloninger, A., 2019, 'Classification logit two-sample testing by neural networks', *CoRR*, vol. abs/1909.11298.

Chwialkowski, K., Ramdas, A., Sejdinovic, D. & Gretton, A., 2015, 'Fast two-sample testing with analytic representations of probability measures', *Proceed-*

ings of the 29th Annual Conference on Neural Information Processing Systems, Montréal, Canada, 7-12 December, 2015, pp. 1981–1989.

Cohn, D. L., 2013, *Measure Theory*, 2nd edn., Springer, New York.

Cortes, C., Mansour, Y. & Mohri, M., 2010, 'Learning bounds for importance weighting', *Proceedings of the 24th Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 6-9 December, 2010, pp. 442–450.

Courty, N., Flamary, R., Tuia, D., Member, S. & Rakotomamonjy, A., 2017, 'Optimal transport for domain adaptation', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1853 – 1865.

Cucker, F. & Smale, S., 2001, 'On the mathematical foundations of learning', *Bulletin of the American Mathematical Society*, vol. 39, no. 1, pp. 1–49.

Deng, J., Dong, W., Socher, R., Li, L., Li, K. & Li, F., 2009, 'Imagenet: A large-scale hierarchical image database', *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, Florida, USA, 20-25 June, 2009, pp. 248–255.

Deng, Z., Choi, K., Jiang, Y. & Wang, S., 2014, 'Generalized hidden-mapping ridge regression, knowledge-leveraged inductive transfer learning for neural networks, fuzzy systems and kernel methods', *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2585–2599.

Deng, Z., Jiang, Y., Choi, K., Chung, F. & Wang, S., 2013a, 'Knowledge-leverage-based TSK fuzzy system modeling', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 8, pp. 1200–1212.

Deng, Z., Jiang, Y., Chung, F., Ishibuchi, H., Choi, K. & Wang, S., 2016a, 'Transfer prototype-based fuzzy clustering', *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 5, pp. 1210–1232.

Deng, Z., Jiang, Y., Chung, F., Ishibuchi, H. & Wang, S., 2013b, 'Knowledge-leverage-based fuzzy system and its modeling', *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 4, pp. 597–609.

Deng, Z., Jiang, Y., Ishibuchi, H., Choi, K. & Wang, S., 2016b, 'Enhanced knowledge-leverage-based TSK fuzzy system modeling for inductive transfer learning', *ACM Transactions on Intelligent Systems and Technology*, vol. 8, no. 1, pp. 1–21.

Duan, L., Tsang, I. W. & Xu, D., 2012a, 'Domain transfer multiple kernel learning', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 465–479.

Duan, L., Xu, D. & Tsang, I., 2012b, 'Learning with augmented features for heterogeneous domain adaptation', *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, UK, 26 June - 1 July, 2012, pp. 711–718.

Duan, L., Xu, D. & Tsang, I. W., 2012c, 'Domain adaptation from multiple sources: A domain-dependent regularization approach', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 3, pp. 504–518.

Dudley, R. M., 2002, *Real analysis and probability*, Cambridge University Press.

Dwass, M., 1957, 'Modified randomization tests for nonparametric hypotheses', *The Annals of Mathematical Statistics*, vol. 28, no. 1, pp. 181–187.

Fernando, B., Habrard, A., Sebban, M. & Tuytelaars, T., 2013, 'Unsupervised visual domain adaptation using subspace alignment', *Proceedings of the 2013 IEEE International Conference on Computer Vision*, Sydney, Australia, 1-8 December, 2013, pp. 2960–2967.

Gamrian, S. & Goldberg, Y., 2019, 'Transfer learning for related reinforcement learning tasks via image-to-image translation', *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, CA, USA, 9-15 June, 2019, pp. 2063–2072.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M. & Lempitsky, V. S., 2016a, 'Domain-adversarial training of neural networks', *Journal of Machine Learning Research*, vol. 17, pp. 59:1–59:35.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M. & Lempitsky, V. S., 2016b, 'Domain-adversarial training of neural networks', *Journal of Machine Learning Research*, vol. 17, pp. 59:1–59:35.

Ghifary, M., Balduzzi, D., Kleijn, W. B. & Zhang, M., 2017, 'Scatter component analysis : A unified framework for domain adaptation and domain generalization', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1414–1430.

Ghosh, D. & Chakraborty, D., 2012, 'Analytical fuzzy plane geometry I', *Fuzzy Sets and Systems*, vol. 209, pp. 66–83.

Ghosh, D. & Chakraborty, D., 2016, 'Analytical fuzzy plane geometry III', *Fuzzy Sets and Systems*, vol. 283, pp. 83–107.

Ghoshdastidar, D., Gutzeit, M., Carpentier, A. & von Luxburg, U., 2017, 'Two-sample tests for large random graphs using network statistics', *Proceedings of the 30th Conference on Learning Theory*, Amsterdam, The Netherlands, 7-10 July, 2017, pp. 954–977.

Gibbons, J. D. & Chakraborti, S., 2011, *Nonparametric statistical inference*, Springer, New York.

Goetschel, R. & Voxman, W., 1983, 'Topological properties of fuzzy numbers', *Fuzzy Sets and Systems*, vol. 10, pp. 87–99.

Gönen, M. & Alpaydın, E., 2011, 'Multiple kernel learning algorithms', *Journal of Machine Learning Research*, vol. 12, pp. 2211–2268.

Gong, B., Grauman, K. & Sha, F., 2014, 'Learning kernels for unsupervised domain adaptation with applications to visual object recognition', *International Journal of Computer Vision*, vol. 109, no. 1-2, pp. 3–27.

Gong, B., Shi, Y., Sha, F. & Grauman, K., 2012, 'Geodesic flow kernel for unsupervised domain adaptation', *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, USA, 16-21 June, 2012, pp. 2066–2073.

Gong, M., Zhang, K., Huang, B., Glymour, C., Tao, D. & Batmanghelich, K., 2018, 'Causal generative domain adaptation networks', *CoRR*, vol. abs/1804.04333.

Gong, M., Zhang, K., Liu, T., Tao, D., Glymour, C. & Systems, I., 2016, 'Domain adaptation with conditional transferable components', *Proceedings of the 33rd International Conference on Machine Learning*, New York City, NY, USA, 19-24 June, 2016, pp. 2839–2848.

Gong, R., Li, W., Chen, Y. & Gool, L. V., 2019, 'DLOW: domain flow for adaptation and generalization', *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 16-20 June, 2019, pp. 2477–2486.

Gopalan, R., Li, R. & Chellappa, R., 2014, 'Unsupervised adaptation across domain shifts by generating intermediate data representations', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2288–2302.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B. & Smola, A. J., 2012, 'A kernel two-sample test', *Journal of Machine Learning Research*, vol. 13, pp. 723–773.

Gretton, A., Fukumizu, K., Harchaoui, Z. & Sriperumbudur, B. K., 2009, 'A fast, consistent kernel two-sample test', *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 7-10 December 2009, pp. 673–681.

Griffin, G., Holub, A. & Perona, P., 2007, 'Caltech-256 object category dataset', Tech. rep., California Institute of Technology.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I. W. & Sugiyama, M., 2018, 'Co-teaching: Robust training of deep neural networks with extremely noisy labels', *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, Montréal, Canada, 3-8 December, 2018, pp. 8536–8546.

Harchaoui, Z., Bach, F. R. & Moulines, E., 2007, 'Testing for homogeneity with kernel fisher discriminant analysis', *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 3-6 December, 2007, pp. 609–616.

Heller, R. & Heller, Y., 2016, 'Multivariate tests of association based on univariate tests', *Proceedings of the 30th Annual Conference on Neural Information Processing Systems*, Barcelona, Spain, 5-10 December, 2016, pp. 208–216.

Henze, N., 1988, 'A multivariate two-sample test based on the number of nearest neighbor type coincidences', *The Annals of Statistics*, vol. 16, no. 2, pp. 772–783.

Hoffman, J., Mohri, M. & Zhang, N., 2018a, 'Algorithms and theory for multiple-source adaptation', *Proceedings of the 32nd Neural Information Processing Systems*, Montréal, Canada, 3-8 December, 2018, pp. 8256–8266.

Hoffman, J., Rodner, E., Donahue, J., Saenko, K. & Darrell, T., 2013, 'Efficient learning of domain-invariant image representations', *Proceedings of the 1st International Conference on Learning Representations*, Scottsdale, AZ, USA, 2-4 May, 2013.

Hoffman, J., Tzeng, E., Park, T., Zhu, J., Isola, P., Saenko, K., Efros, A. A. & Darrell, T., 2018b, 'CyCADA: Cycle-consistent adversarial domain adapta-

tion', *Proceedings of the 35th International Conference on Machine Learning*, Stockholmsmässan, Stockholm, Sweden, 10-15 July, 2018, pp. 1994–2003.

Huang, X. & Paul, M. J., 2019, 'Neural temporality adaptation for document classification: Diachronic word embeddings and domain adaptation models', *Proceedings of the 57th Conference of the Association for Computational Linguistics*, Florence, Italy, 28 July - 2 August, 2019, pp. 4113–4123.

Jean, N., Xie, S. M. & Ermon, S., 2018, 'Semi-supervised deep kernel learning: Regression with unlabeled data by minimizing predictive variance', *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, Montréal, Canada, 3-8 December, 2018, pp. 5327–5338.

Jiang, L., Zhou, Z., Leung, T., Li, L. & Fei-Fei, L., 2018, 'Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels', *Proceedings of the 35th International Conference on Machine Learning*, Stockholmsmässan, Stockholm, Sweden, 10-15 July, 2018, pp. 2309–2318.

Jitkrittum, W., Szabó, Z., Chwialkowski, K. P. & Gretton, A., 2016, 'Interpretable distribution features with maximum testing power', *Proceedings of the 30th Annual Conference on Neural Information Processing Systems*, Barcelona, Spain, 5-10 December, 2016, pp. 181–189.

Jitkrittum, W., Xu, W., Szabo, Z., Fukumizu, K. & Gretton, A., 2017, 'A linear-time kernel goodness-of-fit test', *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, Long Beach, CA, USA, 4-9 December, 2017, pp. 262–271.

Kaleva, O. & Seikkala, S., 1984, 'On fuzzy metric spaces', *Fuzzy Sets and Systems*, vol. 12, pp. 215–229.

Kanamori, T., Hido, S. & Sugiyama, M., 2009, 'A least-squares approach to direct importance estimation', *Journal of Machine Learning Research*, vol. 10, pp. 1391–1445.

Kang, G., Jiang, L., Yang, Y. & Hauptmann, A. G., 2019, 'Contrastive adaptation network for unsupervised domain adaptation', *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 16-20 June, 2019, pp. 4893–4902.

Kingma, D. P. & Ba, J., 2015, 'Adam: A method for stochastic optimization', *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, CA, USA, 7-9 May, 2015.

Kirchler, M., Khorasani, S., Kloft, M. & Lippert, C., 2019, 'Two-sample testing using deep learning', *CoRR*, vol. abs/1910.06239.

Kiryo, R., Niu, G., du Plessis, M. C. & Sugiyama, M., 2017, 'Positive-unlabeled learning with non-negative risk estimator', *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, Long Beach, CA, USA, 4-9 December, 2017, pp. 1675–1685.

Klir, G. J. & Yklsit, B. Y., 1995, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, 1st edn., Prentice Hall PTR.

Korolyuk, V. S. & Borovskikh, Y. V., 1988, 'Asymptotic theory of U-statistics', *Ukrainian Mathematical Journal*, vol. 40, no. 2, pp. 142–154.

Krizhevsky, A., 2009, 'Learning multiple layers of features from tiny images', .

Kulis, B., Saenko, K. & Darrell, T., 2011, 'What you saw is not what you get: Domain adaptation using asymmetric kernel transforms', *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, USA, pp. 1785–1792.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. et al., 1998, 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324.

Lee, J. & Raginsky, M., 2018, 'Minimax statistical learning with wasserstein distances', *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, Montréal, Canada, 3-8 December, 2018, pp. 2692–2701.

Lee, K., He, X., Zhang, L. & Yang, L., 2018, 'Cleannet: Transfer learning for scalable image classifier training with label noise', *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 18-22 June, 2018, pp. 5447–5456.

Lhéritier, A. & Cazals, F., 2018, 'A sequential non-parametric multivariate two-sample test', *IEEE Transactions on Information Theory*, vol. 64, no. 5, pp. 3361–3370.

Li, H., Li, W., Cao, H., Wang, S., Huang, F. & Kot, A. C., 2018a, 'Unsupervised domain adaptation for face anti-spoofing', *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1794–1809.

Li, J., Liu, Y., Yin, R. & Wang, W., 2019a, 'Multi-class learning using unlabeled samples: Theory and algorithm', *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, China, 10-16 August, 2019, pp. 2880–2886.

Li, J., Liu, Y., Yin, R., Zhang, H., Ding, L. & Wang, W., 2018b, 'Multi-class learning: From theory to algorithm', *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, Montréal, Canada, 3-8 December, 2018, pp. 1593–1602.

Li, J., Lu, K., Huang, Z., Zhu, L. & Shen, H. T., 2019b, 'Heterogeneous domain adaptation through progressive alignment', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 5, pp. 1381–1391.

Li, J., Lu, K., Huang, Z., Zhu, L. & Shen, H. T., 2019c, 'Transfer independently together: A generalized framework for domain adaptation', *IEEE Transactions on Cybernetics*, vol. 49, no. 6, pp. 2144–2155.

Li, W., Chen, L., Xu, D. & Van Gool, L., 2017, 'Visual recognition in rgb images and videos by learning from rgb-d data', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 8, pp. 2030–2036.

Li, W., Duan, L., Xu, D. & Tsang, I. W., 2014, 'Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 6, pp. 1134–1148.

Li, Y., Huang, Q., Xie, W. & Li, X., 2015, 'A novel visual codebook model based on fuzzy geometry for large-scale image classification', *Pattern Recognition*, vol. 48, no. 10, pp. 3125–3134.

Li, Y., Tian, X., Gong, M., Liu, Y., Liu, T., Zhang, K. & Tao, D., 2018c, 'Deep domain generalization via conditional invariant adversarial networks', *Proceedings of the 15th European Conference on Computer Vision*, Munich, Germany, 8-14 September, 2018, pp. 647–663.

Liu, A., Lu, J., Liu, F. & Zhang, G., 2018a, 'Accumulating regional density dissimilarity for concept drift detection in data streams', *Pattern Recognition*, vol. 76, pp. 256–272.

Liu, F., Lu, J., Han, B., Niu, G., Zhang, G. & Sugiyama, M., 2019, 'Butterfly: A panacea for all difficulties in wildly unsupervised domain adaptation', *NeurIPS'19 Workshop on Learning Transferable Skills*, Vancouver, Canada, 8-14 December 2019, pp. 1–8.

Liu, F., Lu, J. & Zhang, G., 2018b, 'Unsupervised heterogeneous domain adaptation via shared fuzzy relations', *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 6, pp. 3555–3568.

Liu, F., Xu, W., Lu, J., Zhang, G., Gretton, A. & Sutherland, D. J., 2020a, 'Learning deep kernels for non-parametric two-sample tests', *Proceedings of the 37th Conference of the Internationan Conference on Machine Learning*, Online, 13 July - 18 July, 2020.

Liu, F., Zhang, G. & Lu, J., 2017, 'Heterogeneous unsupervised domain adaptation based on fuzzy feature fusion', *Proceedings of the 2017 IEEE International Conference on Fuzzy Systems*, Naples, Italy, 9-12 July, 2017, pp. 1–6.

Liu, F., Zhang, G. & Lu, J., 2018c, 'Unconstrained fuzzy feature fusion for heterogeneous unsupervised domain adaptation', *Proceedings of the 2018 IEEE International Conference on Fuzzy Systems*, Rio de Janeiro, Brazil, 8-13 July, 2018, pp. 1–8.

Liu, F., Zhang, G. & Lu, J., 2020b, 'Heterogeneous domain adaptation: An unsupervised approach', *IEEE Transactions on Neural Networks and Learning Systems*, vol. Early Access, pp. 1–15.

Liu, T. & Tao, D., 2016, 'Classification with noisy labels by importance reweighting', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 3, pp. 447–461.

Long, M., Cao, Y., Cao, Z., Wang, J. & Jordan, M. I., 2019, 'Transferable representation learning with deep adaptation networks', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 12, pp. 3071–3085.

Long, M., Cao, Y., Wang, J. & Jordan, M. I., 2015, 'Learning transferable features with deep adaptation networks', *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, 6-11 July 2015, pp. 97–105.

Long, M., Cao, Z., Wang, J. & Jordan, M. I., 2018, 'Conditional adversarial domain adaptation', *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, Long Beach, CA, USA, 4-9 December, 2017, pp. 1640–1650.

Long, M., Wang, J., Cao, Y., Sun, J. & Yu, P. S., 2016a, 'Deep learning of transferable representation for scalable domain adaptation', *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2027–2040.

Long, M., Wang, J., Ding, G., Sun, J. & Yu, P. S., 2013, 'Transfer feature learning with joint distribution adaptation', *Proceedings of the 2013 IEEE International Conference on Computer Vision*, Sydney, Australia, 1-8 December, 2013, pp. 2200–2207.

Long, M., Zhu, H., Wang, J. & Jordan, M. I., 2016b, 'Unsupervised domain adaptation with residual transfer networks', *Proceedings of the 30th Annual Conference on Neural Information Processing Systems*, Barcelona, Spain, 5-10 December, 2016, pp. 136–144.

Long, M., Zhu, H., Wang, J. & Jordan, M. I., 2017, 'Deep transfer learning with joint adaptation networks', *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, 6-11 August, 2017, pp. 2208–2217.

Lopez-Paz, D. & Oquab, M., 2017, 'Revisiting classifier two-sample tests', *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 24-26 April, 2017.

Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S. & Zhang, G., 2015, 'Transfer learning using computational intelligence: A survey', *Knowledge-Based Systems*, vol. 80, pp. 14–23.

Lu, N., Lu, J., Zhang, G. & De Mantaras, R. L., 2016, 'A concept drift-tolerant case-base editing technique', *Artificial Intelligence*, vol. 230, pp. 108–133.

Lu, N., Zhang, G. & Lu, J., 2014, 'Concept drift detection via competence models', *Artificial Intelligence*, vol. 209, pp. 11–28.

Luo, Y., Liu, T., Wen, Y. & Tao, D., 2018, 'Online heterogeneous transfer metric learning', *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, 13-19 July , 2018, pp. 2525–2531.

Luo, Y., Wen, Y., Liu, T. & Tao, D., 2017, 'General heterogeneous transfer distance metric learning via knowledge fragments transfer', *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, Melbourne, Australia, 19-25 August, 2017, pp. 2450–2456.

Ma, X., Zhang, T. & Xu, C., 2019, 'GCAN: graph convolutional adversarial network for unsupervised domain adaptation', *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 16-20 June, 2019, pp. 8266–8276.

Ma, Z., Yang, Y., Nie, F., Sebe, N., Yan, S. & Hauptmann, A. G., 2014, 'Harnessing lab knowledge for real-world action recognition', *International Journal of Computer Vision*, vol. 109, no. 1-2, pp. 60–73.

Malach, E. & Shalev-Shwartz, S., 2017, 'Decoupling "when to update" from "how to update"', *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, Long Beach, CA, USA, 4-9 December, 2017, pp. 961–971.

Mansour, Y., Mohri, M. & Rostamizadeh, A., 2009, 'Domain adaptation: Learning bounds and algorithms', *Proceedings of the 22nd Conference on Learning Theory*, Montréal, Canada, 18-21 June, 2009, pp. 3:1–3:11.

Maurer, A., 2016, 'A vector-contraction inequality for rademacher complexities', *Proceedings of the 27th International Conference on Algorithmic Learning Theory*, Bari, Italy, 19-21 October, 2016,, pp. 3–17.

McDiarmid, C., 1989, *On the method of bounded differences*, Surveys in Combinatorics, London Mathematical Society Lecture Note Series, Cambridge University Press, pp. 148–188.

Mirzaei, A. & Rahmati, M., 2010, 'A novel hierarchical-clustering-combination scheme based on fuzzy-similarity relations', *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 27–39.

Mohri, M., Rostamizadeh, A. & Talwalkar, A., 2018, *Foundations of machine learning*, MIT press.

Muandet, K., Fukumizu, K., Sriperumbudur, B. & Schölkopf, B., 2017, 'Kernel mean embedding of distributions: A review and beyond', *Foundations and Trends® in Machine Learning*, vol. 10, no. 1-2, pp. 1–141.

Müller, A., 1997, 'Integral probability metrics and their generating classes of functions', *Advances in Applied Probability*, vol. 29, no. 2, pp. 429–443.

Nguyen, H. V., Ho, H. T., Member, S. & Patel, V. M., 2015, 'DASH-N : Joint hierarchical domain adaptation and feature learning', *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5479–5491.

Nguyen, T. T., Silander, T., Li, Z. & Leong, T. Y., 2017, 'Scalable transfer learning in heterogeneous, dynamic environments', *Artificial Intelligence*, vol. 247, pp. 70–94.

Nguyen, X., Wainwright, M. J. & Jordan, M. I., 2010, 'Estimating divergence functionals and the likelihood ratio by convex risk minimization', *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5847–5861.

Pan, S. J., Tsang, I. W., Kwok, J. T. & Yang, Q., 2011, 'Domain adaptation via transfer component analysis', *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210.

Pan, S. J. & Yang, Q., 2010, 'A survey on transfer learning', *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359.

Pan, W. & Yang, Q., 2013, 'Transfer learning in heterogeneous collaborative filtering domains', *Artificial Intelligence*, vol. 197, pp. 39–55.

Patrini, G., Rozza, A., Menon, A. K., Nock, R. & Qu, L., 2017, 'Making deep neural networks robust to label noise: A loss correction approach', *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 21-26 July, 2017, pp. 2233–2241.

Radford, A., Metz, L. & Chintala, S., 2016, 'Unsupervised representation learning with deep convolutional generative adversarial networks', *Proceedings of the 4th International Conference on Learning Representations*, San Juan, Puerto Rico, 2-4 May, 2016.

Ramdas, A., Singh, A. & Wasserman, L. A., 2016, 'Classification accuracy as a proxy for two sample testing', *CoRR*, vol. abs/1602.02210.

Recht, B., Roelofs, R., Schmidt, L. & Shankar, V., 2019, 'Do ImageNet classifiers

generalize to ImageNet?', *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, CA, USA, 9-15 June, 2019, pp. 5389–5400.

Reis, D. d., Flach, P., Matwin, S. & Batista, G., 2016, 'Fast unsupervised online drift detection using incremental Kolmogorov-Smirnov test', *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 13-17 August, 2016, pp. 1545–1554.

Rodriguez, A. & Laio, A., 2014, 'Clustering by fast search and find of density peaks', *Science*, vol. 344, no. 6191, pp. 1492–1496.

Rosenbaum, P. R., 2005, 'An exact distribution-free test comparing two multivariate distributions based on adjacency', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 4, pp. 515–530.

Rozantsev, A., Salzmann, M. & Fua, P., 2019, 'Beyond sharing weights for deep domain adaptation', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 4, pp. 801–814.

Saito, K., Ushiku, Y. & Harada, T., 2017, 'Asymmetric tri-training for unsupervised domain adaptation', *Proceedings of the 34th International Conference on Machine Learning*, Sydney, Australia, 6-11 August, 2017, pp. 2988–2997.

Saito, K., Watanabe, K., Ushiku, Y. & Harada, T., 2018, 'Maximum classifier discrepancy for unsupervised domain adaptation', *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 18-22 June, 2018, pp. 3723–3732.

Schilling, M. F., 1986, 'Multivariate two-sample nearest tests based on nearest neighbours', *Journal of the American Statistical Association*, vol. 81, no. 395, pp. 799–806.

Schroff, F., Criminisi, A. & Zisserman, A., 2011, 'Harvesting image databases from the web', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 4, pp. 754–766.

Sedghi, H., Gupta, V. & Long, P. M., 2019, 'The singular values of convolutional layers', *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, LA, USA, 6-9 May, 2019.

Serfling, R. J., 1980, *Approximation Theorems of Mathematical Statistics*, John Wiley & Sons.

Shao, L., Zhu, F. & Li, X., 2015, 'Transfer learning for visual categorization: A survey', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 5, pp. 1019–1034.

Shen, J., Qu, Y., Zhang, W. & Yu, Y., 2018, 'Wasserstein distance guided representation learning for domain adaptation', *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, 2-7 February, 2018, pp. 4058–4065.

Shi, X., Liu, Q., Fan, W. & Yu, P. S., 2013, 'Transfer across completely different feature spaces via spectral embedding', *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 906–918.

Shu, R., Bui, H. H., Narui, H. & Ermon, S., 2018, 'A DIRT-T approach to unsupervised domain adaptation', *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 30 April - 3 May, 2018.

Shu, Y., Cao, Z., Long, M. & Wang, J., 2019, 'Transferable curriculum for weakly-supervised domain adaptation', *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Honolulu, HI, USA, 27 January - 1 February, 2019, pp. 4951–4958.

Smola, A. J. & Schölkopf, B., 2001, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press.

Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Lanckriet, G. R. & Schölkopf, B., 2009, 'Kernel choice and classifiability for rkhs embeddings of probability distributions', *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 7-10 December, 2009.

Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Schölkopf, B. & Lanckriet, G. R., 2010, 'Non-parametric estimation of integral probability metrics', *Proceedings of the 2010 IEEE International Symposium on Information Theory*, Austin, Texas, USA, 13-18 June, 2010, pp. 1428–1432.

Sugiyama, M., Suzuki, T., Nakajima, S., Kashima, H., von Bünau, P. & Kawanabe, M., 2008, 'Direct importance estimation for covariate shift adaptation', *Annals of the Institute of Statistical Mathematics*, vol. 60, no. 4, pp. 699–746.

Sun, B., Feng, J. & Saenko, K., 2016, 'Return of frustratingly easy domain adaptation', *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, Phoenix, AZ, USA, 12-17 February, 2016, pp. 2058–2065.

Sun, B. & Saenko, K., 2015, 'Subspace distribution alignment for unsupervised domain adaptation', *Proceedings of the 26th British Machine Vision Conference*, Swansea, UK, 7-10 September, 2015, pp. 24.1–24.10.

Sutherland, D. J., 2019, 'Unbiased estimators for the variance of MMD estimators', *CoRR*, vol. abs/1906.02104.

Sutherland, D. J., Tung, H., Strathmann, H., De, S., Ramdas, A., Smola, A. J. & Gretton, A., 2017, 'Generative models and model criticism via optimized maximum mean discrepancy', *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 24-26 April, 2017.

Tan, M., Tsang, I. W. & Wang, L., 2014, 'Towards ultrahigh dimensional feature selection for big data', *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1371–1429.

Tommasi, T. & Tuytelaars, T., 2014, 'A testbed for cross-dataset analysis', *ECCV TASK-CV Workshops*, pp. 18–31.

Torralba, A., Fergus, R. & Freeman, W. T., 2008, '80 million tiny images: A large data set for nonparametric object and scene recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 1958–1970.

Tzeng, E., Hoffman, J., Saenko, K. & Darrell, T., 2017, 'Adversarial discriminative domain adaptation', *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 21-26 July, 2017, pp. 2962–2971.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K. & Darrell, T., 2014, 'Deep domain confusion: Maximizing for domain invariance', *CoRR*, vol. abs/1412.3474.

Van der Vaart, A. W., 2000, *Asymptotic Statistics*, Cambridge University Press.

Van Rooyen, B., Menon, A. & Williamson, R., 2015, 'Learning with symmetric label noise: The importance of being unhinged', *Proceedings of the 29th Annual Conference on Neural Information Processing Systems*, Montréal, Canada, 7-12 December, 2015, pp. 10–18.

Wang, C. & Mahadevan, S., 2011, 'Heterogeneous domain adaptation using manifold alignment', *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, Barcelona, Spain, 16-22 July, 2011, pp. 1541–1546.

Wang, J., Chen, Y., Yu, H., Huang, M. & Yang, Q., 2019, 'Easy transfer learning by exploiting intra-domain structures', *Proceedings of the 2019 IEEE International Conference on Multimedia and Expo*, Shanghai, China, 8-12 July, 2019, pp. 1210–1215.

Wang, J., Feng, W., Chen, Y., Yu, H., Huang, M. & Yu, P. S., 2018, 'Visual domain adaptation with manifold embedded distribution alignment', *Proceedings of the 26th ACM international conference on Multimedia*, Seoul, Republic of Korea, 22-26 October, 2018, pp. 402–410.

Wei, P., Ke, Y. & Goh, C. K., 2019, 'A general domain specific feature transfer framework for hybrid domain adaptation', *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 8, pp. 1440–1451.

Wenliang, L., Sutherland, D. J., Strathmann, H. & Gretton, A., 2019, 'Learning deep kernels for exponential family densities', *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, California, USA, 9-15 June, 2019, pp. 6737–6746.

Wilson, A. G., Hu, Z., Salakhutdinov, R. & Xing, E. P., 2016, 'Deep kernel learning', *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, Cadiz, Spain, 9-11 May, 2016, pp. 370–378.

Wong, Y., 1967, 'Differential geometry of grassmann manifolds', *Proceedings of the National Academy of Sciences*, vol. 57, no. 3, pp. 589–594.

Xia, X., Liu, T., Wang, N., Han, B., Gong, C., Niu, G. & Sugiyama, M., 2019, 'Are anchor points really indispensable in label-noise learning?', *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 8-14 December 2019, pp. 6835–6846.

Xiao, J., Hays, J., Ehinger, K. A., Oliva, A. & Torralba, A., 2010, 'SUN database: Large-scale scene recognition from abbey to zoo', *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, 13-18 June, 2010, pp. 3485–3492.

Xiao, M. & Guo, Y., 2015, 'Feature space independent semi-supervised domain adaptation via kernel matching', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 1, pp. 54–66.

Xiao, T., Xia, T., Yang, Y., Huang, C. & Wang, X., 2015, 'Learning from massive noisy labeled data for image classification', *Proceedings of the 2015 IEEE*

Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7-12 June, 2015, pp. 2691–2699.

Xu, J., Ramos, S., Vazquez, D. & Lopez, A. M., 2014, 'Domain adaptation of deformable part-based models', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2367–2380.

Yamada, M., Suzuki, T., Kanamori, T., Hachiya, H. & Sugiyama, M., 2011, 'Relative density-ratio estimation for robust distribution comparison', *Proceedings of the 25th Annual Conference on Neural Information Processing Systems*, Granada, Spain, 12-14 December, 2011, pp. 594–602.

Yamada, M., Suzuki, T., Kanamori, T., Hachiya, H. & Sugiyama, M., 2013, 'Relative density-ratio estimation for robust distribution comparison', *Neural Computation*, vol. 25, no. 5, pp. 1324–1370.

Yan, Y., Li, W., Ng, M. K. P., Tan, M., Wu, H., Min, H. & Wu, Q., 2017, 'Learning discriminative correlation subspace for heterogeneous domain adaptation', *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, Melbourne, Australia, 19-25 August, 2017, pp. 3252–3258.

Yan, Y., Li, W., Wu, H., Min, H., Tan, M. & Wu, Q., 2018, 'Semi-supervised optimal transport for heterogeneous domain adaptation', *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, 13-19 July , 2018, pp. 2969–2975.

Yang, L., Jing, L., Yu, J. & Ng, M. K., 2016, 'Learning transferred weights from co-occurrence data for heterogeneous transfer learning', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 11, pp. 2187–2200.

Yang, X.-S. & Deb, S., 2010, 'Engineering optimisation by cuckoo search', *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343.

Yao, Y., Zhang, Y., Li, X. & Ye, Y., 2019, 'Heterogeneous domain adaptation via soft transfer network', *Proceedings of the 27th ACM International Conference on Multimedia, MM*, ACM, Nice, France, 21-25 October, 2019, pp. 1578–1586.

Ye, K. & Lim, L.-H., 2016, 'Schubert varieties and distances between subspaces of different dimensions', *SIAM Journal on Matrix Analysis and Applications*, vol. 37, no. 3, pp. 1176–1197.

Yeh, Y. R., Huang, C. H. & Wang, Y. C. F., 2014, 'Heterogeneous domain adaptation and classification by exploiting the correlation subspace', *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2009–2018.

Yu, X., Liu, T., Gong, M., Zhang, K., Batmanghelich, K. & Tao, D., 2017, 'Transfer learning with label noise', *CoRR*, vol. abs/1707.09724.

Zadeh, L. A., 1971, 'Similarity relations and fuzzy orderings', *Information Sciences*, vol. 3, pp. 177–200.

Zaremba, W., Gretton, A. & Blaschko, M. B., 2013, 'B-test: A non-parametric, low variance kernel two-sample test', *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, Lake Tahoe, NV, United States, 5-8 December, 2013, pp. 755–763.

Zhang, K., Gong, M. & Schölkopf, B., 2015, 'Multi-source domain adaptation: A

causal view', *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, Austin, TX, USA, 25-30 January, 2015, pp. 3150–3157.

Zhang, Q., Wu, D., Lu, J., Liu, F. & Zhang, G., 2017, 'A cross-domain recommender system with consistent information transfer', *Decision Support Systtems*, vol. 104, pp. 49–63.

Zhang, W., Ouyang, W., Li, W. & Xu, D., 2018, 'Collaborative and adversarial network for unsupervised domain adaptation', *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3801–3809.

Zhang, Y., Liu, T., Long, M. & Jordan, M. I., 2019, 'Bridging theory and algorithm for domain adaptation', *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, CA, USA, 9-15 June, 2019, pp. 7404–7413.

Zhao, L., Pan, S. J. & Yang, Q., 2017, 'A unified framework of active transfer learning for cross-system recommendation', *Artificial Intelligence*, vol. 245, pp. 38–55.

Zhao, P., Hoi, S. C. H., Wang, J. & Li, B., 2014, 'Online transfer learning', *Artificial Intelligence*, vol. 216, pp. 76–102.

Zhong, Z., Zheng, L., Zheng, Z., Li, S. & Yang, Y., 2019, 'Camstyle: A novel data augmentation method for person re-identification', *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1176–1190.

Zhou, J. T., Pan, S. J., Tsang, I. W. & Yan, Y., 2014, 'Hybrid heterogeneous transfer learning through deep learning', *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, Québec City, Canada, 27 -31 July, 2014, pp. 2213–2219.

Zhuo, H. H. & Yang, Q., 2014, 'Action-model acquisition for planning via transfer learning', *Artificial Intelligence*, vol. 212, pp. 80–103.

Ziser, Y. & Reichart, R., 2019, 'Task refinement learning for improved accuracy and stability of unsupervised domain adaptation', *Proceedings of the 57th Conference of the Association for Computational Linguistics*, Florence, Italy, 28 July - 2 August, 2019, pp. 5895–5906.

Zuo, H., Lu, J., Zhang, G. & Liu, F., 2019, 'Fuzzy transfer learning using an infinite gaussian mixture model and active learning', *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 2, pp. 291–303.

Zuo, H., Zhang, G., Pedrycz, W., Behbood, V. & Lu, J., 2017a, 'Fuzzy regression transfer learning in Takagi-Sugeno fuzzy models', *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1795 – 1807.

Zuo, H., Zhang, G., Pedrycz, W., Behbood, V. & Lu, J., 2017b, 'Granular fuzzy regression domain adaptation in Takagi-Sugeno fuzzy models', *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 2, pp. 847–858.