FEATURE-BASED TRANSFER LEARNING WITH REAL-WORLD APPLICATIONS

by

JIALIN PAN

A Thesis Submitted to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in Computer Science and Engineering

September 2010, Hong Kong

Authorization

I hereby declare that I am the sole author of the thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_____

JIALIN PAN

FEATURE-BASED TRANSFER LEARNING WITH REAL-WORLD APPLICATIONS

by

JIALIN PAN

This is to certify that I have examined the above Ph.D. thesis
and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by
the thesis examination committee have been made.

PROF. QIANG YANG, THESIS SUPERVISOR

PROF. SIU-WING CHENG, ACTING HEAD OF DEPARTMENT

Department of Computer Science and Engineering
17 September 2010

ACKNOWLEDGMENTS

First of all, I would like to express my thanks to my supervisor Prof. Qiang Yang sincerely and gratefully for his valuable advice and in-depth guidance throughout my Ph.D. study. In the past four years, I learned a lot from him. With his help, I learned how to survey a field of interest, how to discover interesting research topics, how to write research papers and how to do presentation clearly. From him, I also learned that as a researcher, one needs to be always positive and active, and work harder and harder. What I learned from him would be great wealth in my future research career. I also thank his useful advice and information on my job hunting.

I would also like to thank Prof. James Kwok, who I worked with closely at HKUST during the past four years. James deeply impressed me from his passion for doing research to his sense to research ideas. Special thanks also goes to Prof. Carlos Guestrin and Prof. Andreas Krause, who hosted and supervised me during my visit to Carnegie Mellon University and California Institute of Technology, respectively. I thank them for their advice and discussion on my research work during my visit.

I am also very thankful to Prof. Dit-Yan Yeung, Prof. Fugee Tsung, Prof. Jieping Ye, Prof. Chak-Keung Chan, Prof. Brian Mak and Prof. Raymond Wong for serving as committee members of my proposal and thesis defenses. Their valuable comments are of great help in my research work. Furthermore, during my internship at Microsoft Research Asia, I got generous help from my mentor Dr. Jian-Tao Sun. I also got great help from Xiaochuan Ni, Dr. Gang Wang, Dr. Zheng Chen, Weizhu Chen, Dr. Jingdong Wang and Prof. Zhihua Zhang. I thank them very much.

In addition, I am grateful to current and previous members in our research group. They are Dr. Rong Pan, Dr. Jie Yin, Dr. Dou Shen, Dr. Junfeng Pan, Wenchen Zheng, Wei Xiang, Nan Liu, Bin Cao, Hao Hu, Qian Xu, Yin Zhu, Weike Pan, Erheng Zhong, Si Shen and so on. More especially, I would like to express my special thanks to Dr. Rong Pan. When I was still a master student in Sun Yat-Sen University, Dr. Rong Pan brought me to the field of machine learning. His research passion made me become more and more interested in doing research on machine learning. I am also grateful to a number of colleagues at HKUST. Special thanks goes to Guang Dai, Dr. Hong Chang, Dr. Ivor Tsang, Dr. Jian Xia, Dr. Wu-Jun Li, Dr. Bingsheng He, Dr. Yi Wang, Dr. Pingzhong Tang, Yu Zhang, Qi Wang, Yi Zhen and so on.

Last but not least, I would like to give my deepest gratitude to my parents Yunyu Pan and Jinping Feng, who always encourage and support me when I feel depressed. Meanwhile, I would also like to give my great thanks to my wife Zhiyun Zhao. Her kind help and support make everything I have possible.

I dedicate this dissertation to my parents, my wife and my little son Zhuoxuan Pan.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

FEATURE-BASED TRANSFER LEARNING WITH REAL-WORLD APPLICATIONS

by

JIALIN PAN

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

ABSTRACT

Transfer learning is a new machine learning and data mining framework that allows the training and test data to come from different distributions and/or feature spaces. We can find many novel applications of machine learning and data mining where transfer learning is helpful, especially when we have limited labeled data in our domain of interest. In this thesis, we first survey different settings and approaches of transfer learning and give a big picture of the field. We focus on latent space learning for transfer learning, which aims at discovering a "good" common feature space across domain, such that knowledge transfer becomes possible. In our study, we propose a novel dimensionality reduction framework for transfer learning, which tries to reduce the distance between different domains while preserve data properties as much as possible. This framework is general for many transfer learning problems when domain knowledge is unavailable. Based on this framework, we propose three effective solutions to learn the latent space for transfer learning. We apply these methods to two diverse applications: cross-domain WiFi localization and cross-domain text classification, and achieve promising results. Furthermore, for a specific application area, such as sentiment classification, where domain knowledge is available for encoding to transfer learning methods, we propose a spectral feature alignment algorithm for cross-domain learning. In this algorithm, we try to align domain-specific features from different domains by using some domain independent features as a bridge. Experimental results show that this method outperforms a state-of-the-art algorithm in two real-world datasets on cross-domain sentiment classification.

CHAPTER 1

INTRODUCTION

Supervised data mining and machine learning technologies have already been widely studied and applied to many knowledge engineering areas. However, most traditional supervised algorithms work well only under a common assumption: the training and test data are drawn from the same feature space and the same distribution. Furthermore, the performance of these algorithms heavily reply on collecting high quality and sufficient labeled training data to train a statistical or computational model to make predictions on the future data [127, 77, 189]. However, in many real-world scenarios, labeled training data are in short supply or can only be obtained with expensive cost. This problem has become a major bottleneck of making machine learning and data mining methods more applicable in practice.

In the last decade, semi-supervised learning [233, 34, 131, 27, 90] techniques have been proposed to address the problem that the labeled training data may be too few to build a good classifier, by making use of a large amount of unlabeled data to discover a powerful structure together with a small amount of labeled data to train models. Nevertheless, most semi-supervised methods require that the training data, including labeled and unlabeled data, and the test data are both from the same domain of interest, which implicitly assumes the training and test data are still represented in the same feature space and drawn from the same data distribution.

Instead of exploring unlabeled data to train a precise model, active learning, which is another branch in machine learning for reducing annotation effort of supervised learning, tries to design an active learner to pose queries, usually in the form of unlabeled data instances to be labeled by an oracle (e.g., a human annotator). The key idea behind active learning is that a machine learning algorithm can achieve greater accuracy with fewer training labels if it is allowed to choose the data from which it learns [101, 168]. However, most active learning methods assume that there is a budget for the active learner to pose queries in the domain of interest. In some real-world applications, the budget may be quite limited, where active learning methods may not work in learning accurate classifiers in the domain of interest.

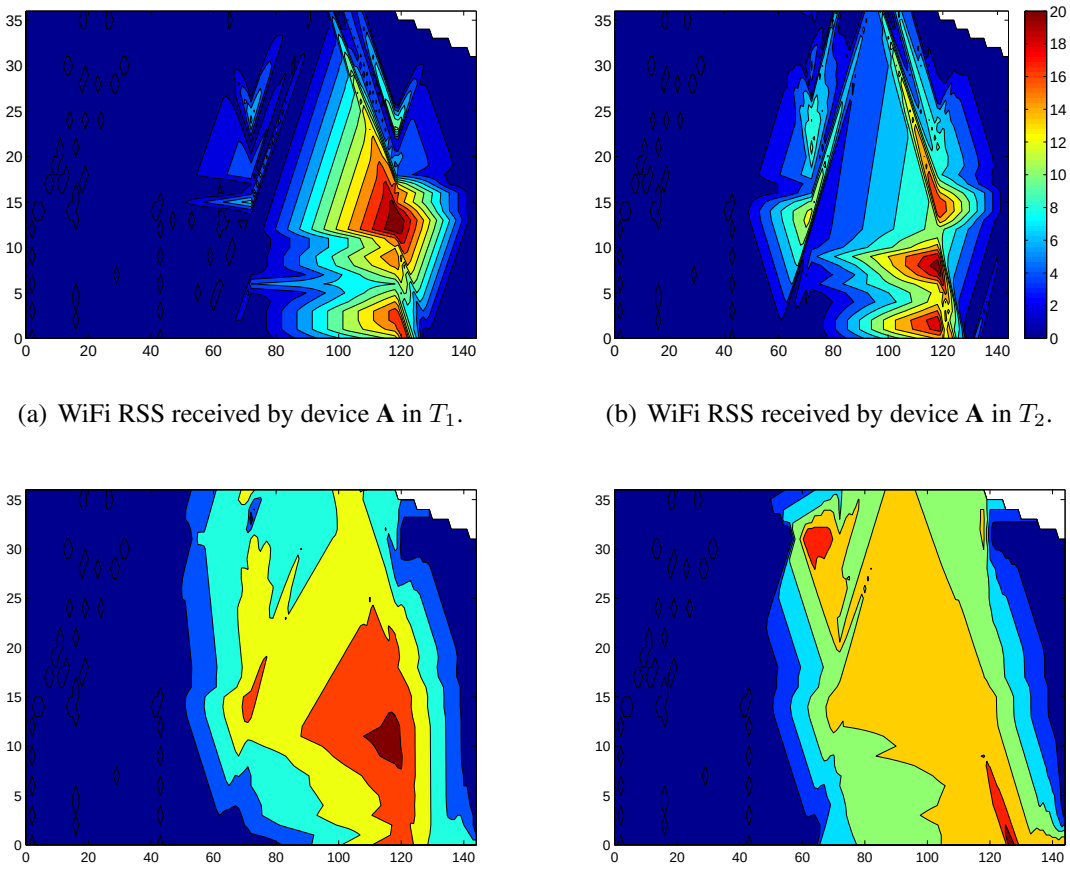
Transfer learning, in contrast, allows the domains, tasks, and distributions used in training and testing to be different. The main idea behind transfer learning is to borrow labeled data or knowledge extracted from some related domains to help a machine learning algorithm to achieve greater performance in the domain of interest [183]. Thus, transfer learning can be referred to as a different strategy for learning model with minimal human supervision, compared to semi-supervised and active learning. In the real world, we can observe many examples of

transfer learning. For example, we may find that learning to recognize apples might help to recognize pears. Similarly, learning to play the electronic organ may help facilitate learning the piano. Furthermore, in many engineering applications, it is expensive or impossible to collect sufficient training data to train a model for use in each domain of interest. It would be nice if one could reuse the training data which have been collected in some related domains/tasks or the knowledge that is already extracted from some related domains/tasks to learn a precise model for use in the domain of interest. In such cases, *knowledge transfer* or *transfer learning* between tasks or domains become more desirable and crucial.

Many examples in knowledge engineering can be found where transfer learning can truly be beneficial. One example is Web document classification, where our goal is to classify a given Web document into several predefined categories. As an example in the area of Web-document classification (see, e.g., [49]), the labeled examples may be the university Web pages that are associated with category information obtained through previous manual-labeling efforts. For a classification task on a newly created Web site where the data features or data distributions may be different, there may be a lack of labeled training data. As a result, we may not be able to directly apply the Web-page classifiers learned on the university Web site to the new Web site. In such cases, it would be helpful if we could transfer the classification knowledge into the new domain.

The need for transfer learning may also arise when the data can be easily outdated. In this case, the labeled data obtained in one time period may not follow the same distribution in a later time period. For example, in indoor WiFi localization problems, which aims to detect a user's current location based on previously collected WiFi data, it is very expensive to calibrate WiFi data for building localization models in a large-scale environment, because a user needs to label a large collection of WiFi signal data at each location. However, the WiFi signal-strength values may be a function of time, device or other dynamic factors. As shown in Figure 4.2, values of received signal strength (RSS) may differ across time periods and mobile devices. As a result, a model trained in one time period or on one device may cause the performance for location estimation in another time period or on another device to be reduced. To reduce the re-calibration effort, we might wish to adapt the localization model trained in one time period (the source domain) for a new time period (the target domain), or to adapt the localization model trained on a mobile device (the source domain) for a new mobile device (the target domain), as introduced in [142].

As a third example, transfer learning is also desirable when the features between domains change. Consider the problem of sentiment classification, where our task is to automatically classify the reviews on a product, such as a brand of camera, into polarity categories (e.g., positive or negative). In literature, supervised learning algorithms [146] have proven to be promising and widely used in sentiment classification. However, these methods are domain



(a) WiFi RSS received by device **A** in T_1.

(b) WiFi RSS received by device **A** in T_2.

(c) WiFi RSS received by device **B** in T_1.

(d) WiFi RSS received by device **B** in T_2.

Figure 1.1: Contours of RSS values over a 2-dimensional environment collected from the same AP but in different time periods and received by different mobile devices. Different colors denote different signal strength values.

dependent. The reason is that users may use domain-specific words to express sentiment in different domains. Table 1.1 shows several user review sentences from two domains: *electronics* and *video games*. In the *electronics* domain, we may use words like "compact", "sharp" to express our positive sentiment and use "blurry" to express our negative sentiment. While in the *video game* domain, words like "hooked", "realistic" indicate positive opinion and the word "boring" indicates negative opinion. Due to the mismatch among domain-specific words, a sentiment classifier trained in one domain may not work well when directly applied to other domains. Thus, cross-domain sentiment classification algorithms are highly desirable to reduce domain dependency and manually labeling cost by transferring knowledge from related domains to the domain of interest [25].

Table 1.1: Cross-domain sentiment classification examples: reviews of *electronics* and *video games* products. Boldfaces are domain-specific words, which are much more frequent in one domain than in the other one. "**+**" denotes positive sentiment, and "**-**" denotes negative sentiment.

	electronics	*video games*
+	**Compact**; easy to operate; very good picture quality; looks **sharp**!	A very good game! It is action packed and full of excitement. I am very much **hooked** on this game.
+	I purchased this unit from Circuit City and I was very excited about the quality of the picture. It is really nice and **sharp**.	Very **realistic** shooting action and good plots. We played this and were **hooked**.
-	It is also quite **blurry** in very dark settings. I will never buy HP again.	The game is so **boring**. I am extremely unhappy and will probably never buy UbiSoft again.

1.1 The Contribution of This Thesis

Generally speaking, transfer learning can be categorized into three settings: *inductive transfer*, *transductive transfer* and *unsupervised transfer*, which is first described in our survey article [141] and will be introduced in detail in Chapter 2. In this thesis, we focus on the *transductive transfer learning* setting, where we are given a lot of labeled data in a source domain and some unlabeled data in a target domain, our goal is to learn an accurate model for use in the target domain. Note that in this setting, no labeled data in the target domain are available for training.

Furthermore, in transfer learning, we have the following three main research issues: (1) What to transfer; (2) How to transfer; (3) When to transfer [141], which will be introduced in detail in Chapter 2 as well.

"What to transfer" asks which part of knowledge can be transferred across domains or tasks. Some knowledge is specific for individual domains or tasks, and some knowledge may be common between different domains such that they may help improve performance for the target domain or task. After discovering which knowledge can be transferred, learning algorithms need to be developed to transfer the knowledge, which corresponds to the "how to transfer" issue.

"When to transfer" asks in which situations, transferring skills should be done. Likewise, we are interested in knowing in which situations, knowledge should **not** be transferred. In some situations, when the source domain and target domain are not related to each other, brute-force transfer may be unsuccessful. In the worst case, it may even hurt the performance of learning in the target domain, a situation which is often referred to as *negative transfer*.

In this thesis, we focus on "What to transfer" and "How to transfer" by implicitly assuming that the source and target domains are related to each other. We leave the issue on how to avoid

negative transfer to our future work. For "How to transfer", we propose to discover a latent feature space for transfer learning, where the distance between domains can then be reduced and the important information of the original data can be preserved simultaneously. Standard machine learning and data mining methods can be applied directly in the latent space to train models for making predictions on the target domain data. Thus, the latent space can be treated as a bridge across domains to make knowledge transfer possible and successful.

For "How to transfer", we propose two embedding learning frameworks to learn the latent space based on two different situations: (1) domain knowledge is hidden or hard to capture, and (2) domain knowledge can be observed or easy to encode in embedding learning. In most application areas, such as text classification or WiFi localization, the domain knowledge is hidden. For example, text data may be controlled by some latent topics, and WiFi data may be controlled by some hidden factors, such as the structure of a building, etc. In this case, we propose a novel and general dimensionality reduction framework for transfer learning. Our framework aims to learn a latent space underlying across domains, such that the distance between data distributions can be dramatically reduced and the original data properties, such as variance and local geometric structure, can be preserved as much as possible, when data from different domains are projected onto the latent space. Based on this framework, we propose three different algorithms to learn the latent space, Maximum Mean Discrepancy Embedding (MMDE) [135], Transfer Component Analysis (TCA) [139] and Semi-supervised Transfer Component Analysis (SSTCA) [140]. More specifically, in MMDE we translate the latent space learning for transfer learning to a non-parametric kernel matrix learning problem. The resultant kernel in a free form may be more precise for transfer learning, but suffers from expensive computational cost. Thus, in TCA and SSTCA, we propose to learn parametric kernel based embeddings for transfer learning instead. The main difference between TCA and SSTCA is that TCA is an unsupervised feature extraction method while SSTCA is semi-supervised feature extrantion method. We apply these three algorithms to two diverse application areas: wireless sensor networks and Web mining.

In contrast to the general framework to transfer learning without domain knowledge, in some application areas, such as sentiment classification, some domain knowledge can be observed and used for learning the latent space across domains. For example, in sentiment classification, though users may use some domain-specific words as shown in Table 1.1, they may use some domain-independent sentiment words, such as "good", "never buy", etc. In addition, some domain-specific and domain-independent words may co-occur in reviews frequently, which means there may be a correlation between these words. This observation motivates us to propose a spectral feature clustering framework [137] to align domain-specific words from different domains in a latent space by modeling the correlation between the domain-independent and domain-specific words in a bipartite graph and using the domain-specific features as a

bridge for cross-domain sentiment classification.

In this thesis, we study the problem of feature-based transfer learning and its real-world applications, such as WiFi localization, text classification and sentiment classification. Note that there has been a large amount of work on transfer learning for reinforcement learning in the machine learning literature (e.g., a current survey article [182]). However, in this thesis, we only focus on transfer learning for classification and regression tasks that are related more closely to machine learning and data mining tasks. The main contributions of this thesis can be summarized as follows,

- We give a comprehensive survey on transfer learning, where we summarize different transfer learning settings and approaches and discuss the relationship between transfer learning and other related areas. Other researchers may get a big picture of transfer learning by reading the survey.
- We propose a general dimensionality reduction framework for transfer learning without any domain knowledge. Based on the framework, we propose three solutions to learn the latent space for transfer learning. Furthermore, we apply them to solve the WiFi localization and text classification problems and achieve promising results.
- We propose a specific latent space learning for sentiment classification, which encode the domain knowledge in a spectral feature alignment framework. The proposed method outperforms a sate-of-the-art cross-domain methods in the field of sentiment classification.

1.2 The Organization of This Thesis

The organization of this thesis is shown in Figure 1.2. In Chapter 2, we survey the field of transfer learning, where we give some definitions of transfer learning, summarize transfer learning into three settings, categorize transfer learning approaches into four contexts, analyze the relationship between transfer learning and other related areas, discuss some interesting research issues in transfer learning and introduce some applications of transfer learning. Based on different different situations that whether domain knowledge is available, we propose two feature space learning frameworks for transfer learning. The first framework is focused on the situation that domain knowledge is hidden and hard to encode in embedding learning. In Chapter 3, we present our proposed general dimensionality reduction framework and three proposed algorithms, Maximum Mean Discrepancy Embedding (MMDE) (Chapter 3.4), Transfer Component Analysis (TCA) (Chapter 3.5) and Semi-supervised Transfer Component Analysis (SSTCA) (Chapter 3.6). Then we apply these three methods to two diverse applications: WiFi localization (Chapter 4) and text classification (Chapter 5). The other framework is focused on the situation that domain knowledge is explicit and easy to be encoded in feature space learning. In

Chapter 6, we present a spectral feature alignment (SFA) algorithm for sentiment classification across domains. Finally, we conclude this thesis and discuss some thoughts on future work in Chapter 7.

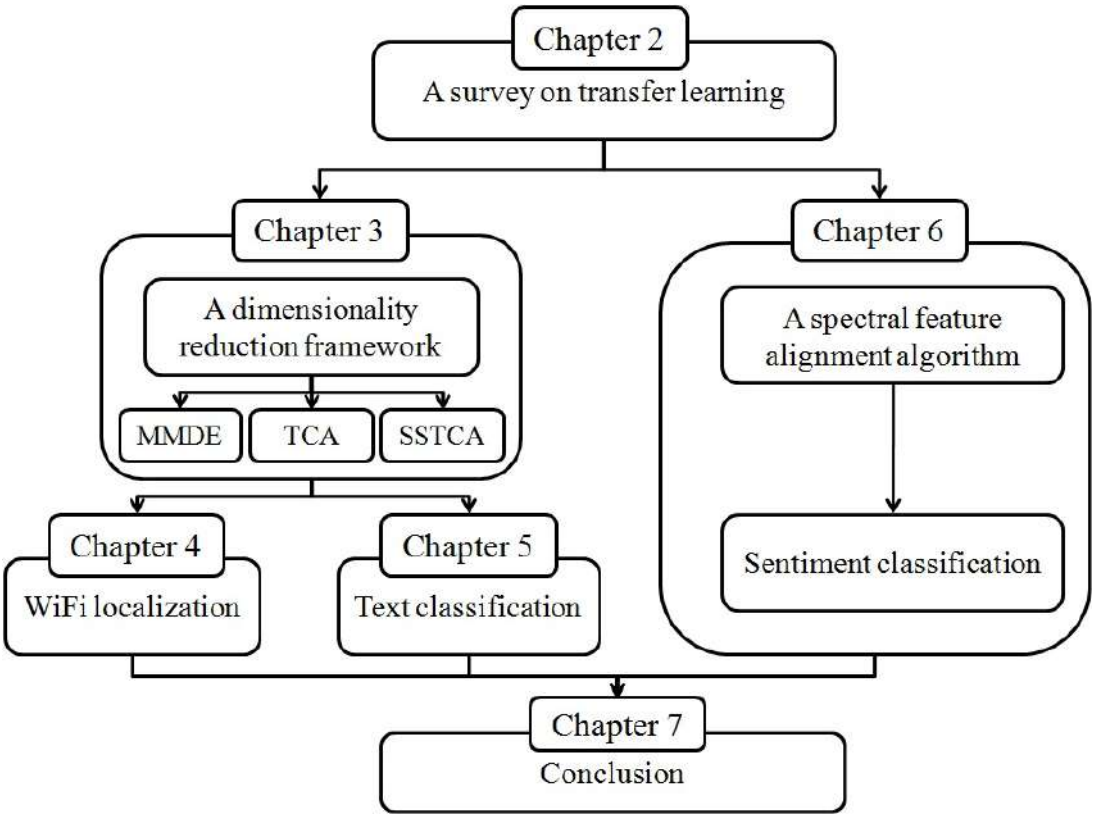


Figure 1.2: The organization of thesis.

CHAPTER 2

A SURVEY ON TRANSFER LEARNING

In this chapter, we give a comprehensive survey of transfer learning for classification, regression and clustering developed in machine learning and data mining areas, and their real-world applications. In fact, this chapter originated as our survey article [141], which is the first survey in the field of transfer learning. Compared to the previous survey article, in this chapter, we add some up-to-date materials on transfer learning, including both methodologies and applications.

2.1 Overview

2.1.1 A Brief History of Transfer Learning

The study of transfer learning is motivated by the fact that people can intelligently apply knowledge learned previously to solve new problems faster or with better solutions [61]. The fundamental motivation for transfer learning in the field of machine learning was discussed in a NIPS-95 workshop on "Learning to Learn"[1], which focused on the need for lifelong machine-learning methods that retain and reuse previously learned knowledge.

Research on transfer learning has attracted more and more attention since 1995 in different names: learning to learn, life-long learning, knowledge transfer, inductive transfer, multi-task learning, knowledge consolidation, context-sensitive learning, knowledge-based inductive bias, meta learning, and incremental/cumulative learning [183]. Among these, a closely related learning technique to transfer learning is the multi-task learning framework [31], which tries to learn multiple tasks simultaneously even when they are different. A typical approach for multi-task learning is to uncover the common (latent) features that can benefit each individual task.

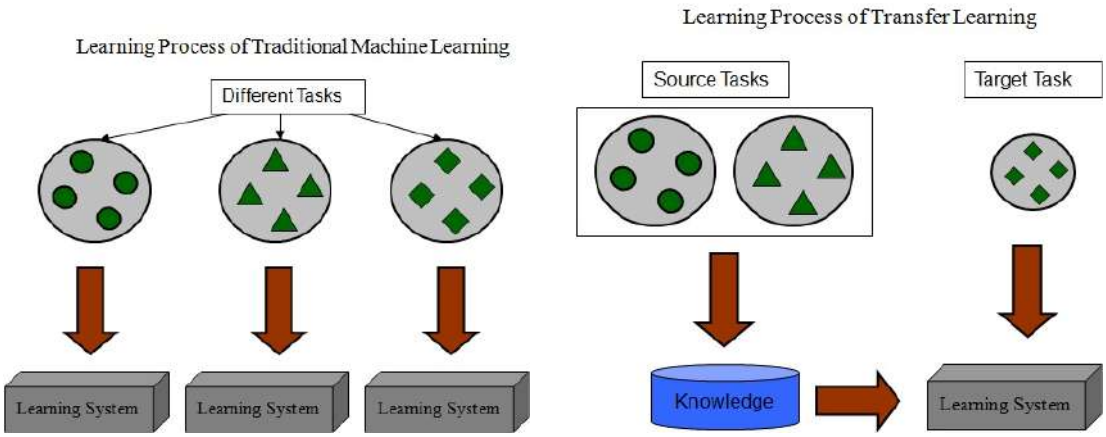
In 2005, the years after the NIPS-05 workshop, the Broad Agency Announcement (BAA) 05-29 of Defense Advanced Research Projects Agency (DARPA)'s Information Processing Technology Office (IPTO) [2] gave a new mission of transfer learning: the ability of a system to recognize and apply knowledge and skills learned in previous tasks to novel tasks. In this definition, transfer learning aims to extract the knowledge from one or more source tasks and applies the knowledge to a target task. In contrast to multi-task learning, rather than learning all

[1] http:/socrates.acadiau.ca/courses/comp/dsilver/NIPS95_LTL/transfer.workshop.1995.html

[2] http://www.darpa.mil/ipto/programs/tl/tl.asp

of the source and target tasks simultaneously, transfer learning cares most about the target task. The roles of the source and target tasks are no longer symmetric in transfer learning.

Figure 2.1 shows the difference between the learning processes of traditional and transfer learning techniques. As we can see, traditional machine learning techniques try to learn each task from scratch, while transfer learning techniques try to transfer the knowledge from some previous tasks to a target task when the latter has fewer high-quality training data.



(a) Traditional Machine Learning (b) Transfer Learning

Figure 2.1: Different learning processes between traditional machine learning and transfer learning

Today, transfer learning methods appear in several top venues, most notably in data mining (ACM International Conference on Knowledge Discovery and Data Mining (KDD), IEEE International Conference on Data Mining (ICDM) and European Conference on Knowledge Discovery in Databases (PKDD), for example), machine learning (International Conference on Machine Learning (ICML), Annual Conference on Neural Information Processing Systems (NIPS) and European Conference on Machine Learning (ECML) for example), artificial intelligence (AAAI Conference on Artificial Intelligence (AAAI) and International Joint Conference on Artificial Intelligence (IJCAI), for example) and applications (Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), International World Wide Web Conference (WWW), Joint Conference of the 47th Annual Meeting of the Association of Computational Linguistics (ACL), International Conference on Computational Linguistics (COLING), Conference on Empirical Methods in Natural Language Processing (EMNLP), IEEE International Conference on Computer Vision (ICCV), European Conference on Computer Vision (ECCV), IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), ACM international conference on Multimedia (MM), ACM International Conference on Ubiquitous Computing (UBICOMP), Annual International Conference on Research in Computational Molecular Biology (RECOMB) and Annual International

Conference on Intelligent Systems for Molecular Biology (ISMB) for example) [3]. Before we give different categorizations of transfer learning, we first describe the notations and definitions used in this chapter.

2.1.2 Notations and Definitions

First of all, we give the definitions of a "domain" and a "task", respectively.

A *domain* $\mathcal{D}$ consists of two components: a feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$, where $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$. For example, if our learning task is document classification, and each term is taken as a binary feature, then $\mathcal{X}$ is the space of all term vectors, x_i is the i^{th} term vector corresponding to some documents, and X is a particular learning sample. In general, if two domains are different, then they may have different feature spaces or different marginal probability distributions.

Given a specific domain, $\mathcal{D} = \{\mathcal{X}, P(X)\}$, a *task* $\mathcal{T}$ consists of two components: a label space $\mathcal{Y}$ and an objective predictive function $f(\cdot)$ (denoted by $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$), which is not observed but can be learned from the training data, which consist of pairs $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in \mathcal{Y}$. The function $f(\cdot)$ can be used to predict the corresponding label, $f(x)$, of a new instance x. From a probabilistic viewpoint, $f(x)$ can be written as $P(y|x)$. In our document classification example, $\mathcal{Y}$ is the set of all labels, which is True, False for a binary classification task, and y_i is "True" or "False".

Here, we only consider the case where there is one source domain $\mathcal{D}_S$, and one target domain, $\mathcal{D}_T$, as this is by far the most popular of the research works in the literature. The issue of transfer learning from multiple source domains will be discussed in 2.6. More specifically, we denote $D_S = \{(x_{S_1}, y_{S_1}), \ldots, (x_{S_{n_S}}, y_{S_{n_S}})\}$ the *source domain data*, where $x_{S_i} \in \mathcal{X}_S$ is the data instance and $y_{S_i} \in \mathcal{Y}_S$ is the corresponding class label. In our document classification example, D_S can be a set of term vectors together with their associated true or false class labels. Similarly, we denote the target domain data as $D_T = \{(x_{T_1}, y_{T_1}), \ldots, (x_{T_{n_T}}, y_{T_{n_T}})\}$, where the input x_{T_i} is in $\mathcal{X}_T$ and $y_{T_i} \in \mathcal{Y}_T$ is the corresponding output. In most cases, $0 \leq n_T \ll n_S$.

We now give a unified definition of transfer learning.

Definition 1. *Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$,* transfer learning *aims to help improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$.*

In the above definition, a domain is a pair $\mathcal{D} = \{\mathcal{X}, P(X)\}$. Thus the condition $\mathcal{D}_S \neq \mathcal{D}_T$ implies that either $\mathcal{X}_S \neq \mathcal{X}_T$ or $P_S(X) \neq P_T(X)$. For example, in our document classification

[3] We summarize a list of transfer learning papers published in these few years and a list of workshops that are related to transfer learning at the following url for reference, `http://www.cse.ust.hk/~sinnopan/conferenceTL.htm`

example, this means that between a source document set and a target document set, either the term features are different between the two sets (e.g., they use different languages), or their marginal distributions are different.

Similarly, a task is defined as a pair $\mathcal{T} = \{\mathcal{Y}, P(Y|X)\}$. Thus the condition $\mathcal{T}_S \neq \mathcal{T}_T$ implies that either $\mathcal{Y}_S \neq \mathcal{Y}_T$ or $P(Y_S|X_S) \neq P(Y_T|X_T)$. When the target and source domains are the same, i.e. $\mathcal{D}_S = \mathcal{D}_T$, and their learning tasks are the same, i.e., $\mathcal{T}_S = \mathcal{T}_T$, the learning problem becomes a traditional machine learning problem. When the domains are different, then either (1) the feature spaces between the domains are different, i.e. $\mathcal{X}_S \neq \mathcal{X}_T$, or (2) the feature spaces between the domains are the same but the marginal probability distributions between domain data are different; i.e. $P(X_S) \neq P(X_T)$, where $X_{S_i} \in \mathcal{X}_S$ and $X_{T_i} \in \mathcal{X}_T$. As an example, in our document classification example, case (1) corresponds to when the two sets of documents are described in different languages, and case (2) may correspond to when the source domain documents and the target domain documents focus on different topics.

Given specific domains $\mathcal{D}_S$ and $\mathcal{D}_T$, when the learning tasks $\mathcal{T}_S$ and $\mathcal{T}_T$ are different, then either (1) the label spaces between the domains are different, i.e. $\mathcal{Y}_S \neq \mathcal{Y}_T$, or (2) the conditional probability distributions between the domains are different; i.e. $P(Y_S|X_S) \neq P(Y_T|X_T)$, where $Y_{S_i} \in \mathcal{Y}_S$ and $Y_{T_i} \in \mathcal{Y}_T$. In our document classification example, case (1) corresponds to the situation where source domain has binary document classes, whereas the target domain has ten classes to classify the documents to. Case (2) corresponds to the situation where the documents classes are defined subjectively, as such tagging. Different users may define different different tags for a same document, resulting in $P(Y|X)$ changes across different users.

In addition, when there exists some relationship, explicit or implicit, between the two domains or tasks, we say that the source and target domains or tasks are *related*. For example, the task classifying documents into the categories {*book, desktop*} may be related the task classifying documents into the categories{*book, laptop*}. This because from a semantic point of view, the terms "laptop" and "desktop" are close to each other. As a result, the learning tasks may be related to each other. Note that it is hard to define the term "relationship" mathematically. Thus, in most transfer learning methods introduced in the following sections assume that the source and target domains or tasks are related. How to measure the relatedness between domains or tasks is an important research issue in transfer learning, which will be discussed in Chapter 2.5.

2.1.3 A Categorization of Transfer Learning Techniques

In transfer learning, we have the following three main research issues: (1) What to transfer; (2) How to transfer; (3) When to transfer.

"What to transfer" asks which part of knowledge can be transferred across domains or tasks. Some knowledge is specific for individual domains or tasks, and some knowledge may be com-

mon between different domains such that they may help improve performance for the target domain or task. After discovering which knowledge can be transferred, learning algorithms need to be developed to transfer the knowledge, which corresponds to the "how to transfer" issue.

"When to transfer" asks in which situations, transferring skills should be done. Likewise, we are interested in knowing in which situations, knowledge should **not** be transferred. In some situations, when the source domain and target domain are not related to each other, brute-force transfer may be unsuccessful. In the worst case, it may even hurt the performance of learning in the target domain, a situation which is often referred to as *negative transfer*. Most current work on transfer learning focuses on "What to transfer" and "How to transfer", by implicitly assuming that the source and target domains be related to each other. However, how to avoid negative transfer is an important open issue that is attracting more and more attention in the future.

Based on the definition of transfer learning, we summarize the relationship between traditional machine learning and various transfer learning settings in Table 2.1, where we categorize transfer learning under three settings, *inductive transfer learning*, *transductive transfer learning* and *unsupervised transfer learning*, based on different situations between the source and target domains and tasks.

Table 2.1: Relationship between traditional machine learning and transfer learning settings

Learning Settings		Source & Target Domains	Source & Target Tasks
Traditional Machine Learning		the same	the same
Transfer Learning	*Inductive Transfer Learning* / *Unsupervised Transfer Learning*	the same	different but related
		different but related	different but related
	Transductive Transfer Learning	different but related	the same

1. In the *inductive transfer learning* setting, the target task is different from the source task, no matter when the source and target domains are the same or not.

   In this case, some labeled data in the target domain are required to *induce* an objective predictive model $f_T(\cdot)$ for use in the target domain. In addition, according to different situations of labeled and unlabeled data in the source domain, we can further categorize the *inductive transfer learning* setting into two cases:

   (1.1) A lot of labeled data in the source domain are available. In this case, the *inductive transfer learning* setting is similar to the multi-task learning setting [31]. However, the *inductive transfer learning* setting only aims at achieving high performance in the target task by transferring knowledge from the source task while multi-task learning tries to learn the target and source task simultaneously.

(1.2) No labeled data in the source domain are available. In this case, the *inductive transfer learning* setting is similar to the *self-taught learning* setting, which is first proposed by Raina *et al.* [150]. In the self-taught learning setting, the label spaces between the source and target domains may be different, which implies the side information of the source domain cannot be used directly. Thus, it's similar to the inductive transfer learning setting where the labeled data in the source domain are unavailable.

2. In the *transductive transfer learning* setting, the source and target tasks are the same, while the source and target domains are different.

   In this situation, no labeled data in the target domain are available while a lot of labeled data in the source domain are available. In addition, according to different situations between the source and target domains, we can further categorize the *transductive transfer learning* setting into two cases.

   (2.1) The feature spaces between the source and target domains are different, $\mathcal{X}_S \neq \mathcal{X}_T$.

   (2.2) The feature spaces between domains are the same, $\mathcal{X}_S = \mathcal{X}_T$, but the marginal probability distributions of the input data are different, $P(X_S) \neq P(X_T)$.

   The latter case of the *transductive transfer learning* setting is related to domain adaptation for knowledge transfer in Natural Language Processing (NLP) [23, 85] and sample selection bias [219] or co-variate shift [170], whose assumptions are similar.

3. Finally, in the *unsupervised transfer learning* setting, similar to *inductive transfer learning* setting, the target task is different from but related to the source task. However, the *unsupervised transfer learning* focus on solving unsupervised learning tasks in the target domain, such as clustering, dimensionality reduction and density estimation [50, 197]. In this case, there are no labeled data available in both source and target domains in training.

The relationship between the different settings of transfer learning and the related areas are summarized in Table 2.2 and Figure 2.2.

Approaches to transfer learning in the above three different settings can be summarized into four contexts based on "What to transfer". Table 2.3 shows these four cases and brief description. The first context can be referred to as instance-based transfer learning(or instance-transfer) approach (see, e.g., [219, 107, 49, 48, 87, 82, 21, 180, 67, 20, 148, 22, 147] for example), which assumes that certain parts of the data in the source domain can be reused for learning in the target domain by *re-weighting*. Instance re-weighting and importance sampling are two major techniques in this context.

A second case can be referred to as feature-representation-transfer approach (see, e.g., [31, 84, 4, 26, 6, 150, 47, 51, 8, 99, 50, 83, 37, 46, 149, 112, 231] for example). The intuitive idea behind this case is to learn a "good" feature representation for the target domain. In this case, the

Table 2.2: Different settings of transfer learning

Transfer Learning Settings	Related Areas	Source Domain Labels	Target Domain Labels	Tasks
Inductive Transfer Learning	Multi-task Learning	Available	Available	Regression, Classification
	Self-taught Learning	Unavailable	Available	Regression, Classification
Transductive Transfer Learning	Domain Adaptation, Sample Selection Bias, Co-variate Shift	Available	Unavailable	Regression, Classification
Unsupervised Transfer Learning		Unavailable	Unavailable	Dimensionality Reduction, Clustering

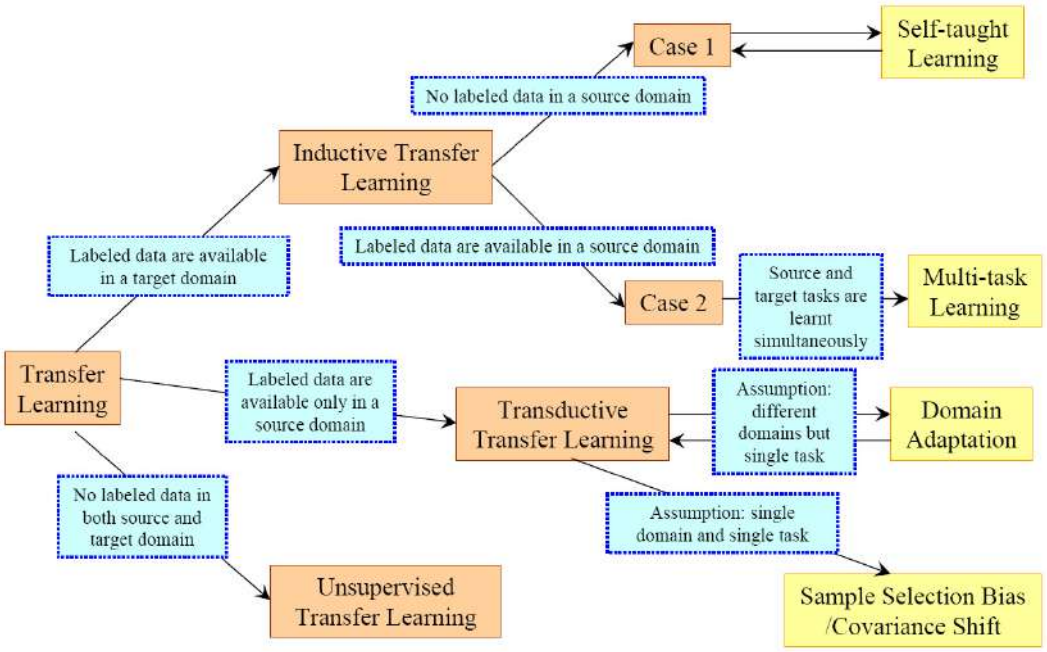


Figure 2.2: An overview of different settings of transfer

knowledge used to transfer across domains is encoded into the learned feature representation. With the new feature representation, the performance of the target task is expected to improve significantly.

A third case can be referred to as parameter-transfer approach (see, e.g., [97, 63, 165, 62, 28, 30] for example), which assumes that the source tasks and the target tasks share some parameters or prior distributions of the hyper-parameters of the models. The transferred knowledge is encoded into the shared parameters or priors. Thus, by discovering the shared parameters or priors, knowledge can be transferred across tasks.

Finally, the last case can be referred to as the relational-knowledge-transfer problem [124], which deals with transfer learning for relational domains. The basic assumption behind this context is that some relationship among the data in the source and target domains are similar.

Table 2.3: Different approaches to transfer learning

Approaches	Brief Description
Instance-transfer	To re-weight some labeled data in the source domain for use in the target domain (see, e.g., [219, 107, 49, 48, 87, 82, 21, 180, 20, 148, 22, 147] for example).
Feature-representation-transfer	Find a "good" feature representation that reduces difference between the source and the target domains and the error of classification and regression models (see, e.g., [31, 84, 4, 26, 6, 150, 47, 51, 8, 99, 50, 83, 37, 46, 149, 112, 231] for example).
Parameter-transfer	Discover shared parameters or priors between the source domain and target domain models, which can benefit for transfer learning (see, e.g., [97, 63, 165, 62, 28, 30] for example).
Relational-knowledge-transfer	Build mapping of relational knowledge between the source domain and the target domains. Both domains are relational domains and i.i.d assumption is relaxed in each domain (see, e.g., [124, 125, 52] for example).

Thus, the knowledge to be transferred is the relationship among the data. Recently, statistical relational learning techniques dominate this context [125, 52].

Table 2.4 shows the cases where the different approaches are used for each transfer learning setting. We can see that the *inductive transfer learning* setting has been studied in many research works, while the *unsupervised transfer learning* setting is a relatively new research topic and only studied in the context of the *feature-representation-transfer* case. In addition, the *feature-representation-transfer* problem has been proposed to all three settings of transfer learning. However, the *parameter-transfer* and the *relational-knowledge-transfer* approach are only studied in the *inductive transfer learning* setting, which we discuss in detail in the following chapters.

Table 2.4: Different approaches in different settings

	Inductive Transfer Learning	Transductive Transfer Learning	Unsupervised Transfer Learning
Instance-transfer	$\surd$	$\surd$	
Feature-representation-transfer	$\surd$	$\surd$	$\surd$
Parameter-transfer	$\surd$		
Relational-knowledge-transfer	$\surd$		

2.2 Inductive Transfer Learning

Definition 2. (Inductive Transfer Learning) *Given a source domain $\mathcal{D}_S$ and a learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and a learning task $\mathcal{T}_T$,* inductive transfer learning *aims to help improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where*

$\mathcal{T}_S \neq \mathcal{T}_T$.

Based on the above definition of the inductive transfer learning setting, a few labeled data in the target domain are required as the training data to *induce* the target predictive function. As mentioned in Chapter 2.1.3, this setting has two cases: (1) Labeled data in the source domain are available; (2) Labeled data in the source domain are unavailable while unlabeled data in the source domain are available. Most transfer learning approaches in this setting focus on the former case.

2.2.1 Transferring Knowledge of Instances

The *instance-transfer approach* to the inductive transfer learning setting is intuitively appealing: although the source domain data cannot be reused directly, there are certain parts of the data that can still be reused together with a few labeled data in the target domain.

Dai *et al.* [49] proposed a boosting algorithm, TrAdaBoost, which is an extension of the AdaBoost [66] algorithm, to address classification problems in the inductive transfer learning setting. TrAdaBoost assumes that the source and target domain data use exactly the same set of features and labels, but the distributions of the data in the two domains are different. In addition, TrAdaBoost also assumes that, due to the difference in distributions between the source and the target domains, some of the source domain data may be useful in learning for the target domain but some of them may not and could even be harmful. In detail, TrAdaBoost attempts to iteratively re-weight the source domain data to reduce the effect of the "bad" source data while encourage the "good" source data to contribute more for the target domain. For each round of iteration, TrAdaBoost trains the base classifier on the weighted source and target data. The error is only calculated on the target data. Furthermore, TrAdaBoost uses the same strategy as AdaBoost to update the incorrectly classified examples in the target domain while using a different strategy from AdaBoost to update the incorrectly classified source examples in the source domain. Theoretical analysis of TrAdaBoost is also presented in [49]. More recently, Pardoe and Stone [147] presented a two-stage algorithm TrAdaBoost.R2 to extend the TrAdaBoost algorithm for regression problems. The idea is to apply the techniques which have been proposed for modifying AdaBoost for regression [56] on TrAdaBoost. Furthermore, to avoid the overfitting problem in TrAdaBoost, Pardoe and Stone proposed to adjust the weights of the data in two stages. In the first stage, only the weights of the source domain data are adjusted downwards gradually until reaching a certain point. Then in the second stage, the weights of source domain data are fixed while the weights of the target domain data are updated as normal in the TrAdaBoost algorithm.

Besides modifying boosting algorithms for transfer learning, Jiang and Zhai [87] proposed a heuristic method to remove "misleading" training examples from the source domain based

on the difference between conditional probabilities $P(y_T|x_T)$ and $P(y_S|x_S)$. Wu and Dietterich [202] integrated the source domain (auxiliary) data in a Support Vector Machine (SVM) [189] framework for improving the classification performance in the target domain. Gao *et al.* [67] proposed a graph-based locally weighted ensemble framework to combine multiple models for transfer learning. The idea is to assign weights to various models dynamically based on local structures in a graph, then weighted models are used to make predictions on text data.

2.2.2 Transferring Knowledge of Feature Representations

The feature-representation-transfer approach to the inductive transfer learning problem aims at finding "good" feature representations to minimize domain divergence and classification or regression model error. Strategies to find "good" feature representations are different for different types of the source domain data. If a lot of labeled data in the source domain are available, supervised learning methods can be used to construct a feature representation. This is similar to *common feature learning* in the field of multi-task learning [31]. If no labeled data in the source domain are available, unsupervised learning methods are proposed to construct the feature representation.

Supervised Feature Construction

Supervised feature construction methods for the inductive transfer learning setting are similar to those used in multi-task learning. The basic idea is to learn a low-dimensional representation that is shared across related tasks. In addition, the learned new representation can reduce the classification or regression model error of each task as well. Argyriou *et al.* [6] proposed a sparse feature learning method for multi-task learning. In the inductive transfer learning setting, the common features can be learned by solving an optimization problem, given as follows.

$$\begin{aligned} \arg\min_{A,U} \quad & \sum_{t\in\{T,S\}} \sum_{i=1}^{n_t} L(y_{t_i}, \langle a_t, U^T x_{t_i}\rangle) + \gamma \|A\|_{2,1}^2 \\ s.t. \quad & U \in \mathbf{O}^d \end{aligned} \tag{2.1}$$

In this equation, S and T denote the tasks in the source domain and target domain, respectively. $A = [a_S, a_T] \in R^{d\times 2}$ is a matrix of parameters. U is a $d \times d$ orthogonal matrix (mapping function) for mapping the original high-dimensional data to low-dimensional representations. The (r,p)-norm of A is defined as $\|A\|_{r,p} := (\sum_{i=1}^{d} \|a^i\|_r^p)^{\frac{1}{p}}$. The optimization problem (2.1) estimates the low-dimensional representations $U^T X_T$, $U^T X_S$ and the parameters, A, of the model at the same time. The optimization problem (2.1) can be further transformed into an equivalent convex optimization formulation and be solved efficiently. In a follow-up work,

Argyriou *et al.* [8] proposed a spectral regularization framework on matrices for multi-task structure learning.

Another famous common feature learning method for multi-task learning is the alternating structure optimization ASO algorithm, proposed by Ando and Zhang [4]. In ASO, a linear classifier is trained for each of the multiple tasks. Then weight vectors of the multiple classifiers are used to construct a predictor space. Finally, Singular Vector Decomposition (SVD) is applied on the space to recover a low-dimensional predictive space as a common feature space underlying multiple tasks. The ASO algorithm has been applied successfully to several applications [25, 5]. However, it is non-convex and does not guarantee to find a global optimum. More recently, Chen *et al.* [37] presented an improved formulation (*i*ASO) based on ASO by proposing a novel regularizer. Furthermore, in order to convert the new formulation into a convex formulation, in [37], Chen *et al.* proposed a convex alternating structure optimization (*c*ASO) algorithm to the optimization problem.

Lee *et al.* [99] proposed a convex optimization algorithm for simultaneously learning meta-priors and feature weights from an ensemble of related prediction tasks. The meta-priors can be transferred among different tasks. Jebara [84] proposed to select features for multi-task learning with SVMs. Ruckert *et al.* [160] designed a kernel-based approach to inductive transfer, which aims at finding a suitable kernel for the target data.

Unsupervised Feature Construction

In [150], Raina *et al.* proposed to apply sparse coding [98], which is an unsupervised feature construction method, for learning *higher level* features for transfer learning. The basic idea of this approach consists of two steps. At the first step, higher-level basis vectors $b = \{b_1, b_2, \ldots, b_s\}$ are learned on the source domain data by solving the optimization problem (2.2) as shown as follows,

$$\begin{aligned} \min_{a,b} \sum_i \quad & \|x_{S_i} - \textstyle\sum_j a^j_{S_i} b_j\|_2^2 + \beta \|a_{S_i}\|_1 \\ s.t. \qquad & \|b_j\|_2 \leq 1, \forall j \in 1, \ldots, s \end{aligned} \tag{2.2}$$

In this equation, $a^j_{S_i}$ is a new representation of basis b_j for input x_{S_i} and β is a coefficient to balance the feature construction term and the regularization term. After learning the basis vectors b, in the second step, an optimization algorithm (2.3) is applied on the target domain data to learn *higher level* features based on the basis vectors b.

$$a^*_{T_i} = \arg\min_{a_{T_i}} \|x_{T_i} - \sum_j a^j_{T_i} b_j\|_2^2 + \beta \|a_{T_i}\|_1 \tag{2.3}$$

Finally, discriminative algorithms can be applied to $\{a^*_{T_i}\}'s$ with corresponding labels to train classification or regression models for use in the target domain. One drawback of this method

is that the so-called higher-level basis vectors learned on the source domain in the optimization problem (2.2) may not be suitable for use in the target domain.

Recently, manifold learning methods have been adapted for transfer learning. In [193], Wang and Mahadevan proposed a Procrustes analysis based approach to manifold alignment without correspondences, which can be used to transfer the knowledge across domains via the aligned manifolds.

2.2.3 Transferring Knowledge of Parameters

Most parameter-transfer approaches to the inductive transfer learning setting assume that individual models for related tasks should share some parameters or prior distributions of hyper-parameters. Most approaches described in this section, including a regularization framework and a hierarchical Bayesian framework, are designed to work under multi-task learning. However, they can be easily modified for transfer learning. As mentioned above, multi-task learning tries to learn both the source and target tasks simultaneously and perfectly, while transfer learning only aims at boosting the performance of the target domain by utilizing the source domain data. Thus, in multi-task learning, weights of the loss functions for the source and target data are the same. In contrast, in transfer learning, weights in the loss functions for different domains can be different. Intuitively, we may assign a larger weight to the loss function of the target domain to make sure that we can achieve better performance in the target domain.

Lawrence and Platt [97] proposed an efficient algorithm known as MT-IVM, which is based on Gaussian Processes (GP), to handle the multi-task learning case. MT-IVM tries to learn parameters of a Gaussian Process over multiple tasks by sharing the same GP prior. Bonilla *et al.* [28] also investigated multi-task learning in the context of GP. The authors proposed to use a free-form covariance matrix over tasks to model inter-task dependencies, where a GP prior is used to induce correlations between tasks. Schwaighofer *et al.* [165] proposed to use a hierarchical Bayesian framework (HB) together with GP for multi-task learning.

Besides transferring the priors of the GP models, some researchers also proposed to transfer parameters of SVMs under a regularization framework. Evgeniou and Pontil [63] borrowed the idea of HB to SVMs for multi-task learning. The proposed method assumes that the parameter, w, in SVMs for each task can be separated into two terms. One is a common term over tasks and the other is a task-specific term. In inductive transfer learning,

$$w_S = w_0 + v_S, \ \ \& \ \ w_T = w_0 + v_T,$$

where, w_S and w_T are parameters of the SVMs for the source task and the target learning task, respectively. w_0 is a common parameter while v_S and v_T are specific parameters for the source task and the target task, respectively. By assuming $f_t = w_t \cdot x$ to be a hyper-plane for task t, an

extension of SVMs to multi-task learning case can be written as the following:

$$\begin{aligned} \min_{w_0, v_t, \xi_{t_i}} \quad & J(w_0, v_t, \xi_{t_i}) = \sum_{t \in \{S,T\}} \sum_{i=1}^{n_t} \xi_{t_i} + \frac{\lambda_1}{2} \sum_{t \in \{S,T\}} \|v_t\|^2 + \lambda_2 \|w_0\|^2 \\ s.t. \quad & y_{t_i}(w_0 + v_t) \cdot x_{t_i} \geq 1 - \xi_{t_i}, \\ & \xi_{t_i} \geq 0,\ i \in \{1, 2, ..., n_t\}\ \&\ t \in \{S, T\}, \end{aligned}$$

where x_{t_i}'s, $t \in \{S, T\}$, are input feature vectors in the source and target domains, respectively. y_{t_i}'s are the corresponding labels. ξ_{t_i}'s are slack variables to measure the degree of misclassification of the data x_{t_i}'s as used in standard SVMs. λ_1 and λ_2 are positive regularization parameters to tradeoff the importance between the misclassification and regularization terms. By solving the optimization problem above, we can learn the parameters w_0, v_S and v_T simultaneously.

2.2.4 Transferring Relational Knowledge

Different from other three contexts, the relational-knowledge-transfer approach deals with transfer learning problems in relational domains, where the data are non-i.i.d. and can be represented by multiple relations, such as networked data and social network data. This approach does not assume that the data drawn from each domain be independent and identically distributed (i.i.d.) as traditionally assumed. It tries to transfer the *relationship* among data from a source domain to a target domain. In this context, *statistical relational learning techniques* are proposed to solve these problems.

Mihalkova *et al.* [124] proposed an algorithm TAMAR that transfers relational knowledge with Markov Logic Networks (MLNs) [155] across relational domains. MLNs is a powerful formalism, which combines the compact expressiveness of first order logic with flexibility of probability, for statistical relational learning. In an MLN, entities in a relational domain are represented by predicates and their relationships are represented in first-order logic. TAMAR is motivated by the fact that if two domains are related to each other, there may exist mappings to connect entities and their relationships from a source domain to a target domain. For example, a professor can be considered as playing a similar role in an academic domain as a manager in an industrial management domain. In addition, the relationship between a professor and his or her students is similar to the relationship between a manager and his or her workers. Thus, there may exist a mapping from professor to manager and a mapping from the professor-student relationship to the manager-worker relationship. In this vein, TAMAR tries to use an MLN learned for a source domain to aid in the learning of an MLN for a target domain. Basically, TAMARis a two-stage algorithm. At the first stage, a mapping is constructed from a source MLN to the target domain based on weighted pseudo loglikelihood measure (WPLL).

At the second stage, a revision is done for the mapped structure in the target domain through the FORTE algorithm [152], which is an inductive logic programming (ILP) algorithm for revising first order theories. The revised MLN can be used as a relational model for inference or reasoning in the target domain.

In a follow-up work [125], Mihalkova *et al.* extended TAMAR to the single-entity-centered setting of transfer learning, where only one entity in a target domain is available. Davis *et al.* [52] proposed an approach to transferring relational knowledge based on a form of second-order Markov logic. The basic idea of the algorithm is to discover structural regularities in the source domain in the form of Markov logic formulas with predicate variables, by instantiating these formulas with predicates from the target domain.

2.3 Transductive Transfer Learning

The term *transductive transfer learning* was first proposed by Arnold *et al.* [9], where they required that the source and target tasks be the same, although the domains may be different. On top of these conditions, they further required that that all unlabeled data in the target domain are available at training time, but we believe that this condition can be relaxed; instead, in our definition of the transductive transfer learning setting, we only require that *part* of the unlabeled target domain data be seen at training time in order to obtain the marginal probability for the target domain data.

Note that the word "transductive" is used with several meanings. In the traditional machine learning setting, *transductive learning* [90] refers to the situation where all test data are required to be seen at training time, and that the learned model cannot be reused for future data. Thus, when some new test data arrive, they must be classified together with all existing data. In our categorization of transfer learning, in contrast, we use the term "transductive" to emphasize the concept that in this type of transfer learning, the tasks must be the same and there must be some unlabeled data available in the target domain.

Definition 3. (Transductive Transfer Learning) *Given a source domain $\mathcal{D}_S$ and a corresponding learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and a corresponding learning task $\mathcal{T}_T$,* transductive transfer learning *aims to improve the learning of the target predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$. In addition, some unlabeled target domain data must be available at training time.*

This definition covers the work of Arnold *et al.* [9], since the latter considered *domain adaptation*, where the difference lies between the marginal probability distributions of source and target data; i.e., the tasks are the same but the domains are different. For example, assume our task is to classify whether a document describes information about *laptop* or not, the source

domain documents are downloaded from news websites, and are annotated by human, the target domain documents are documents are downloaded from shopping websites. As a result, the data distributions may be different across domains. The goal of *transductive transfer learning* is to make use of some unlabeled (without human annotation) target documents with a lot of labeled source domain documents to train a good classifier to make predictions on the documents in the target domain (including unseen documents in training).

Similar to the traditional transductive learning setting, which aims to make the best use of the unlabeled test data for learning, in our classification scheme under transductive transfer learning, we also assume that some target-domain unlabeled data be given. In the above definition of transductive transfer learning, the source and target tasks are the same, which implies that one can adapt the predictive function learned in the source domain for use in the target domain through some unlabeled target-domain data. As mentioned in Chapter 2.1.2, this setting can be split to two cases: (a) The feature spaces between the source and target domains are different, $\mathcal{X}_S \neq \mathcal{X}_T$, and (b) the feature spaces between domains are the same, $\mathcal{X}_S = \mathcal{X}_T$, but the marginal probability distributions of the input data are different, $P(X_S) \neq P(X_T)$. This is similar to the requirements in domain adaptation and sample selection bias. Most approaches described in the following sections are related to case (b) above.

2.3.1 Transferring the Knowledge of Instances

Most instance-transfer approaches to the transductive transfer learning setting are motivated by importance sampling. To see how importance sampling based methods may help in this setting, we first review the problem of empirical risk minimization (ERM) [189]. In general, we might want to learn the optimal parameters θ^* of the model by minimizing the expected risk,

$$\theta^* = \arg\min_{\theta \in \Theta} \mathbb{E}_{(x,y)\in P}[l(x, y, \theta)],$$

where $l(x, y, \theta)$ is a loss function that depends on the parameter θ. However, since it is hard to estimate the probability distribution P, we choose to minimize the ERM instead,

$$\theta^* = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} [l(x_i, y_i, \theta)],$$

where x_i's are input feature vectors and y_i's are the corresponding labels. n is size of the training data.

In the transductive transfer learning setting, we want to learn an optimal model for the target domain by minimizing the expected risk,

$$\theta^* = \arg\min_{\theta \in \Theta} \sum_{(x,y)\in D_T} P(D_T) l(x, y, \theta)$$

However, since no labeled data in the target domain are observed in training data, we have to learn a model from the source domain data instead. If $P(D_S) = P(D_T)$, then we may simply learn the model by solving the following optimization problem for use in the target domain,

$$\theta^* = \arg\min_{\theta \in \Theta} \sum_{(x,y) \in D_S} P(D_S) l(x, y, \theta).$$

Otherwise, when $P(D_S) \neq P(D_T)$, we need to modify the above optimization problem to learn a model with high generalization ability for the target domain, as follows:

$$\begin{aligned} \theta^* &= \arg\min_{\theta \in \Theta} \sum_{(x,y) \in D_S} \frac{P(D_T)}{P(D_S)} P(D_S) l(x, y, \theta) \\ &\approx \arg\min_{\theta \in \Theta} \sum_{i=1}^{n_S} \frac{P_T(x_{T_i}, y_{T_i})}{P_S(x_{S_i}, y_{S_i})} l(x_{S_i}, y_{S_i}, \theta). \end{aligned} \tag{2.4}$$

Therefore, by adding different penalty values to each instance (x_{S_i}, y_{S_i}) with the corresponding weight $\frac{P_T(x_{T_i}, y_{T_i})}{P_S(x_{S_i}, y_{S_i})}$, we can learn a precise model for the target domain. Furthermore, since $P(Y_T|X_T) = P(Y_S|X_S)$. Thus the difference between $P(D_S)$ and $P(D_T)$ is caused by $P(X_S)$ and $P(X_T)$ and $\frac{P_T(x_{T_i}, y_{T_i})}{P_S(x_{S_i}, y_{S_i})} = \frac{P(x_{S_i})}{P(x_{T_i})}$. If we can estimate $\frac{P(x_{S_i})}{P(x_{T_i})}$ for each instance, we can solve the transductive transfer learning problems.

There exist various ways to estimate $\frac{P(x_{S_i})}{P(x_{T_i})}$. Zadrozny [219] proposed to estimate the terms $P(x_{S_i})$ and $P(x_{T_i})$ independently by constructing simple classification problems. Huang *et al.* [82] proposed a kernel-mean matching (KMM) algorithm to learn $\frac{P(x_{S_i})}{P(x_{T_i})}$ directly by matching the means between the source domain data and the target domain data in a reproducing-kernel Hilbert space (RKHS). KMM can be rewritten as the following quadratic programming (QP) optimization problem.

$$\begin{aligned} \min_{\beta} \quad & \frac{1}{2} \beta^T K \beta - \kappa^T \beta \\ s.t. \quad & \beta_i \in [0, B] \ \ and \ \ |\sum_{i=1}^{n_S} \beta_i - n_S| \leq n_S \epsilon \end{aligned} \tag{2.5}$$

where $K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix}$ and $K_{ij} = k(x_i, x_j)$. $K_{S,S}$ and $K_{T,T}$ are kernel matrices for the source domain data and the target domain data, respectively. That means $K_{S,S_{ij}} = k(x_i, x_j)$, where $x_i, x_j \in X_S$, and $K_{T,T_{ij}} = k(x_i, x_j)$, where $x_i, x_j \in X_T$. $K_{S,T}$ ($K_{T,S} = K_{S,T}^T$) is a kernel matrix across the source and target domain data, which implies $K_{S,T_{ij}} = k(x_i, x_j)$, where $x_i \in X_S$ and $x_j \in X_T$. $\kappa_i = \frac{n_S}{n_T} \sum_{j=1}^{n_T} k(x_i, x_{T_j})$, where $x_i \in X_S \bigcup X_T$, while $x_{T_j} \in X_T$.

It can be proved that $\beta_i = \frac{P(x_{S_i})}{P(x_{T_i})}$ [82]. An advantage of using KMM is that it can avoid performing density estimation of either $P(x_{S_i})$ or $P(x_{T_i})$, which is difficult when the size of the

data set is small. Sugiyama *et al.* [180] proposed an algorithm known as Kullback-Leibler Importance Estimation Procedure (KLIEP) to estimate $\frac{P(x_{S_i})}{P(x_{T_i})}$ directly, based on the minimization of the Kullback-Leibler divergence. KLIEP can be integrated with cross-validation to perform model selection automatically in two steps: (1) estimating the weights of the source domain data; (2) training models on the re-weighted data. Bickel *et al.* [21] combined the two steps in a unified framework by deriving a kernel-logistic regression classifier. Kanamori *et al.* [91] proposed a method called *unconstrained least-squares importance fitting* (uLSIF) to estimate the importance efficiently by formulating the direct importance estimation problem as a least-squares function fitting problem. More recently, Sugiyama *et al.* [179] further extended the uLSIF algorithm by estimating importance in a non-stationary subspace, which performs well even when the dimensionality of the data domains is high. However, this method is focused on estimating the *importance* in a latent space instead of learning a latent space for adaptation. For more information on importance sampling and re-weighting methods for co-variate shift or sample selection bias, readers can refer to a recently published book [148] by Quionero-Candela et al. To be emphasized that besides covariate shift adaptation, these importance estimation techniques have also been applied to various applications, such as independent component analysis (ICA) [181], outlier detection [78] and change-point detection [92]. Besides sample re-weighting techniques, Dai *et al.* [48] extended a traditional Naive Bayesian classifier for the transductive transfer learning problems.

2.3.2 Transferring Knowledge of Feature Representations

Most feature-representation transfer approaches to the transductive transfer learning setting are under unsupervised learning frameworks. Blitzer *et al.* [26] proposed a structural correspondence learning (SCL) algorithm, which modifies the ASO [5] algorithm for transductive transfer learning, to make use of the unlabeled data from the target domain to extract some common features to reduce the difference between the source and target domains.

As described in Chapter 2.2, ASO was proposed for multi-task learning. Thus, a first step is to construct some pseudo related tasks. In SCL, Blitzer *et al.* proposed to first define a set of *pivot* features (the number of *pivot* feature is denoted by m), which are common features that occur frequently and similarly across domains, using labeled source domain and unlabeled target domain data. For example, the words "good", "nice" and "bad", etc, are examples of pivot features, which are sentiment words and used commonly across different domains (e.g, reviews on different products). Then, SCL treats each *pivot* feature as a new output vector to construct a task and non-pivot features as inputs. The m linear classifiers are trained to model the relationship between the non-pivot features and the pivot features as shown as follows,

$$f_l(x) = sgn(w_l^T \cdot x),\ l = 1, \ldots, m$$

as in ASO singular value decomposition (SVD) is applied on the weight matrix $W = [w_1 w_2 \dots w_m] \in \mathbb{R}^{q \times m}$, where q is number of features of the original data, such that $W = UDV^T$, where $U_{q \times r}$ and $V_{r \times m}$ are the matrices of the left and right singular vectors. The matrix $D_{r \times r}$ is a diagonal matrix consists of non-negative singular values, which are ranked in non-increasing order. Let $\theta = U^T_{[1:h,:]}$ (h is the number of the shared features) is a transformation mapping to map non-pivot features to a latent space, where the difference between domains can be reduced. Finally, standard classification algorithms can be applied on the new representations to train classifiers. In [26], Blitzer *et al.* used a heuristic method to select pivot features for natural language processing (NLP) problems, such as tagging of sentences. In their follow-up work [25], Mutual Information (MI) is proposed for choosing the pivot features.

Daumé III [51] proposed a simple feature augmentation method for transfer learning problems in the Natural Language Processing (NLP) area. It aims to augment each of the feature vectors of different domains to a high dimensional feature vector as follows,

$$\widetilde{x}_S = [x_S, x_S, \mathbf{0}] \ \& \ \widetilde{x}_T = [x_T, \mathbf{0}, x_T],$$

where x_S and x_T are original features vectors of the source and target domains, respectively. $\mathbf{0}$ is a vector of zeros, whose length is equivalent to that of the original feature vector. The idea is to reduce the difference between domains while ensure the similarity between data within domains is larger than that across different domain data. Dai *et al.* [47] proposed a co-clustering based algorithm to discover common feature clusters, such that label information can be propagated across different domains by using the common clusters as a bridge. Xing *et al.* [205], proposed a novel algorithm known as *bridged refinement* to correct the labels predicted by a shift-unaware classifier towards a target distribution and take the mixture distribution of the training and test data as a bridge to better transfer from the training data to the test data. Ling *et al.* [108] proposed a spectral classification framework for cross-domain transfer learning problem, where the objective function is introduced to seek consistency between the in-domain supervision and the out-of-domain intrinsic structure. Xue *et al.* [207] proposed a cross-domain text classification algorithm that extended the traditional probabilistic latent semantic analysis (PLSA) algorithm to integrate labeled and unlabeled data from different but related domains, into a unified probabilistic model.

Most feature-based transductive transfer learning methods do not minimize the distance in distributions between domains directly. Recently, von Bünau *et al.* [191] proposed a method known as stationary subspace analysis (SSA) to match distributions in a latent space for time series data analysis. However, SSA is focused on the identification of a stationary subspace, without considering the preservation of properties such as data variance in the subspace. More specifically, SSA theoretically studied the conditions under which a stationary space can be identified from multivariate time series. They also proposed the SSA procedure to find stationary components by matching the first two moments of the data distributions in different epochs.

However, SSA is focused on how to identify a stationary subspace without considering how to preserve data properties in the latent space as well. As a result, SSA may map the data to some noisy factors which are stationary across domains but completely irrelevant to the target supervised task. Then classifiers trained on the new representations learned by SSA may not get good performance for transductive transfer learning.

2.4 Unsupervised Transfer Learning

Definition 4. (Unsupervised Transfer Learning) *Given a source domain $\mathcal{D}_S$ with a learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and a corresponding learning task $\mathcal{T}_T$,* unsupervised transfer learning *aims to help improve the learning of the target predictive function $f_T(\cdot)$ [4] in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{T}_S \neq \mathcal{T}_T$ and $\mathcal{Y}_S$ and $\mathcal{Y}_T$ are not observable.*

Based on the definition of the unsupervised transfer learning setting, no labeled data are observed in the source and target domains in training. For example, assume we have a lot of documents downloaded from news websites, which are referred to as source domain documents, and have a few documents downloaded from shopping websites, which are referred to as target domain documents. The task is to cluster the target domain documents into some hidden categories. Note that it is usually unable to find precise clusters if the data are sparse. Thus, the goal of unsupervised transfer learning is to make use of the documents in the source domain, where precise clusters can be obtained because of the sufficient of training data, to guide clustering on the target domain documents. So far, there is little research work in this setting.

2.4.1 Transferring Knowledge of Feature Representations

Dai *et al.* [50] studied a new case of clustering problems, known as self-taught clustering. Self-taught clustering (STC) is an instance of *unsupervised transfer learning*, which aims at clustering a small collection of unlabeled data in the target domain with the help of a large amount of unlabeled data in the source domain. STC tries to learn a common feature space across domains, which helps in clustering in the target domain. The objective function of STC is shown as follows.

$$J(\tilde{X_T}, \tilde{X_S}, \tilde{Z}) = I(X_T, Z) - I(\tilde{X_T}, \tilde{Z}) + \lambda \left[I(X_S, Z) - I(\tilde{X_S}, \tilde{Z}) \right]$$

where X_S and X_T are the source and target domain data, respectively. Z is a shared feature space by X_S and X_T, and $I(\cdot, \cdot)$ is the mutual information between two random variables. Suppose that there exist three clustering functions $C_{X_T} : X_T \rightarrow \tilde{X_T}$, $C_{X_S} : X_S \rightarrow \tilde{X_S}$ and

[4] In unsupervised transfer learning, the predicted labels are latent variables, such as clusters or reduced dimensions

$C_Z : Z \rightarrow \tilde{Z}$, where $\tilde{X_T}$, $\tilde{X_S}$ and $\tilde{Z}$ are corresponding clusters of X_T, X_S and Z, respectively. The goal of STC is to learn $\tilde{X_T}$ by solving the optimization problem (2.6):

$$\arg\min_{\tilde{X_T}, \tilde{X_S}, \tilde{Z}} J(\tilde{X_T}, \tilde{X_S}, \tilde{Z}) \tag{2.6}$$

An iterative algorithm for solving the optimization function (2.6) was presented in [50].

Similarly, Wang *et al.* [197] proposed a transferred discriminative analysis (TDA) algorithm to solve the transfer dimensionality reduction problem. TDA first applies clustering methods to generate pseudo-class labels for the target domain unlabeled data. It then applies dimensionality reduction methods to the target domain data and labeled source domain data to reduce the dimensionalities. These two steps run iteratively to find the best subspace for the target domain data. More recently, Bhattacharya *et al.* [19] proposed a new clustering algorithm to transfer supervision to a clustering task in a target domain from a different source domain, where a relevant supervised data partition is provided. The main idea is to define a cross-domain similarity measure to align clusters in the target domain and the partitions in the source domain, such that the performance of clustering in the target domain can be improved.

2.5 Transfer Bounds and Negative Transfer

An important issue is to recognize the limit of the power of transfer learning. In transductive transfer learning, there are some research works focusing on the generalization bound when the training and test data distributions are different [16, 24, 15, 17]. Though the generalization bounds proved in different literatures are different slightly, there is a common conclusion that the generalization bound of a learning model in transductive transfer learning consists of two terms, one is the error bound of the learning model on the labeled source domain data, the other is the distance between the source and target domains, more specifically the distance between marginal probability distributions of input features across domains.

In inductive transfer learning, to date, there are few work proposed to study the issue of transferability. Hassan Mahmud and Ray [120] analyzed the case of transfer learning using Kolmogorov complexity, where some theoretical bounds are proved. In particular, the authors used conditional Kolmogorov complexity to measure relatedness between tasks and transfer the "right" amount of information in a sequential transfer learning task under a Bayesian framework. Recently, Eaton *et al.* [60] proposed a novel graph-based method for knowledge transfer, where the relationships between source tasks are modeled by embedding the set of learned source models in a graph using transferability as the metric. Transferring to a new task proceeds by mapping the problem into the graph and then learning a function on this graph that automatically determines the parameters to transfer to the new learning task.

How to avoid negative transfer and then ensure a "safe transfer" of knowledge is another crucial issue in transfer learning. Negative transfer happens when the source domain data and task contribute to the reduced performance of learning in the target domain. Despite the fact that how to avoid negative transfer is a very important issue, little research work was published on this topic in the past. Rosenstein *et al.* [159] empirically showed that if two tasks are too dissimilar, then brute-force transfer may hurt the performance of the target task. Some works have been exploited to analyze relatedness among tasks and task clustering techniques, such as [18, 11], which may help provide guidance on how to avoid negative transfer automatically. Bakker and Heskes [11] adopted a Bayesian approach in which some of the model parameters are shared for all tasks and others more loosely connected through a joint prior distribution that can be learned from the data. Thus, the data are clustered based on the task parameters, where tasks in the same cluster are supposed to be related to each others.

More recently, Argyriou *et al.* [7] considered situations in which the learning tasks can be divided into groups. Tasks within each group are related by sharing a low-dimensional representation, which differs among different groups. As a result, tasks within a group can find it easier to transfer useful knowledge. Jacob *et al.* [83] presented a convex approach to clustered multi-task learning by designing a new spectral norm to penalize over a set of weights, each of which is associated to a task. Bonilla *et al.* [28] proposed a multi-task learning method based on Gaussian Process (GP), which provides a global approach to model and learn task relatedness in the form of a task covariance matrix. However, the optimization procedure introduced in [28] is non-convex, whose results may be sensitive to parameter initialization. Motivated by [28], Zhang and Yeung [224] proposed an improved regularization framework to model the negative and positive correlation between tasks, where the resultant optimization procedure is convex.

As described above, most research works on model task correlation are from the context of multi-task learning [18, 11, 7, 83, 28, 224]. However, in inductive transfer learning, one may be particularly interested in transferring knowledge from one or more source tasks to a target task rather than learning these tasks simultaneously. The main concern of inductive transfer learning is the learning performance in the target task only. Thus, we need to give an answer to the question that given a target task and a source task, whether transfer learning techniques should be applied or not. Cao *et al.* [30] proposed an Adaptive Transfer learning algorithm based on GP (AT-GP), which aims to adapt the transfer learning schemes by automatically estimating the similarity between the source and target tasks. In AT-GP, a new semi-parametric kernel is designed to model correlation between tasks, and the learning procedure targets at improving performance of the target task only. Seah *et al.* [167] empirically study the negative transfer problem by proposing a predictive distribution matching classifier based on Support Vector Machines (SVMs) to identify the regions of relevant source domain data where the predictive distributions maximally align with that of the target domain data, and then avoid negative

transfer.

2.6 Other Research Issues of Transfer Learning

Besides the negative transfer issue, in recent few years, there are several research issues of transfer learning that have attracted more and more attention from the machine learning and data mining communities. We summarize them as follows,

- **Transfer learning from multiple source domains**. Most transfer learning methods introduced in previous chapters focus on one-to-one transfer, which means there are only one source domain and one target domain. However, in some real-world scenarios, we may have multiple sources at hand. Developing algorithms to make use of multiple sources for help learning models in the target domain is useful in practice. Yang *et al.* [209] and Duan *et al.* [57] proposed algorithms to a new SVM for the target domains by adapting some SVMs learned from multiple source domains. Luo *et al.* [119] proposed to train a classifier for use in the target domain by maximizing the consensus of predictions from multiple sources. Mansour *et al.* [122] proposed a distribution weighted linear combining framework for learning from multiple sources. The main idea is to estimate the data distribution of each source to reweight the data from different source domains. Theoretical studies on transfer learning from multiple source domains have also been presented in [122, 123, 15]

- **Transfer learning across different feature space** As described in the definition of transfer learning, the source and target domain data may come from different feature spaces. Thus, how to transfer knowledge successfully across different feature spaces is another interesting issue. It is related to unlike multi-view learning [27], which assumes the features for each instance can be divided into several views, each with its own distinct feature space. Though multi-view learning techniques can be applied to model multi-modality data, it requires each instance in one view must have its correspondence in other views. In contrast, transfer learning across different feature spaces aims to solve the problem when the source and target domain data belong to two different feature spaces such as image vs. text, without correspondences across feature spaces. Ling *et al.* [109] proposed an information-theoretic approach for transfer learning to address the *cross-language classification problem*. The approach addressed the problem when there are plenty of labeled English text data whereas there are only a small number of labeled Chinese text documents. Transfer learning across the two feature spaces are achieved by designing a suitable mapping function across feature spaces as a bridge. Dai *et al.* [45] proposed a new risk minimization framework based on a language model for machine translation [44] for

dealing with the problem of learning heterogeneous data that belong to quite different feature spaces. Yang *et al.* [211] and Chen *et al.* [40] proposed probabilistic models to leverage textual information for help image clustering and image advertising, respectively.

- **Transfer learning with active learning** As mentioned at the beginning of this Chapter 2, active learning and transfer learning both aims at learning a precise model with minimal human supervision for a target task. Several researchers have proposed to combine these two techniques together in order to learn a more precise model with even less supervision. Liao *et al.* [107] proposed a new active learning method to select the unlabeled data in a target domain to be labeled with the help of the source domain data. Shi *et al.* [169] applied an active learning algorithm to select important instances for transfer learning with TrAdaBoost [49] and standard SVM. In [33], Chan and NG proposed to adapt existing Word Sense Disambiguation (WSD) systems to a target domain by using domain adaptation techniques and employing an active learning strategy [101] to actively select examples to be annotated from the target domain. Harpale and Yang [75] proposed an active learning framework for the multi-task adaptive filtering [157] problem. They first proposed to apply multi-task learning techniques for adaptive filtering, and then explore various active learning approaches to the multi-task adaptive filter to improve the performance.

- **Transfer learning for other tasks** Besides classification, regression, clustering and dimensionality reduction, transfer learning has also been proposed for other tasks, such as metric learning [221, 226], structure learning [79] and online learning [227]. Zha *et al.* [221] proposed a regularization to learn a new distance metric in a target domain by leverage pre-learned distance metrics from auxiliary domains. Zhang and Yeung [226] first proposed a convex formulation for multi-task metric learning by modeling the task relationships in the form of a task covariance matrix, and then adapt it to the transfer learning setting. In [79], Honorio and Samaras proposed to apply multi-task learning techniques to learn structures across multiple gaussian graphical models simultaneously. In [227], Zhao and Hoi investigated a framework to transfer knowledge from a source domain to an online learning task in a target domain.

2.7 Real-world Applications of Transfer Learning

Recently, transfer learning has been applied successfully to many application areas, such Natural Language Processing (NLP), Information Retrieval (IR), recommendation systems, computer vision, image analysis, multimedia data mining, bioinformatics, activity recognition and wireless sensor networks.

In the fields of Natural Language Processing (NLP), transfer learning, which is known as domain adaptation, has been widely studied for solving various tasks [42], such as name entity recognition [71, 9, 156, 51, 201], part-of-speech tagging [4, 26, 87, 51], sentiment classification [25, 104, 113], word sense disambiguation [33, 2] and information extraction [200, 86]

Information Retrieval (IR) is another application area where transfer learning techniques have been widely studied and applied. Web applications of transfer learning include text classification across domains [151, 47, 48, 72, 195, 207, 36, 204, 213, 203], advertising [40, 39], learn to rank across domains [10, 192, 35, 68] and multi-domain collaborative filtering [103, 102, 29, 223, 143].

Besides NLP and IR, in the past two years, transfer learning techniques have attracted more and more attention in the fields of computer vision, image and multimedia analysis. Applications of transfer learning in these fields include image classification [202, 158, 185, 218, 161, 94, 177], image retrieval [73, 115, 38], face verification from images [196], age estimation from facial images [225], image semantic segmentation [190], video retrieval [208, 73], video concept detection [209, 58] and event recognition from videos [59].

In Bioinformatics, motivated by that different biological entities, such as organisms, genes, etc, may be related to each other from a biological point of view, some research works have been proposed to apply transfer learning techniques to solve various computational biological problems, such as identifying molecular association of phenotypic responses [222], splice site recognition of eukaryotic genomes [199], mRNA splicing [166], protein protein subcellular location prediction [206] and genetic association analysis [214]

Transfer learning techniques has also explored to solve WiFi localization and sensor-based activity recognition problems [210]. For example, transfer learning techniques were proposed to extract knowledge from WiFi localization models across time periods [217, 136, 230], space [138, 194] and mobile devices [229], to benefit WiFi localization tasks in other settings. Rashidi and Cook [153] and Zheng *et al.* [228] proposed to apply transfer learning techniques for solving indoor sensor-based activity recognition problems, respectively.

Furthermore, Zhuo *et al.* [235] studied how to transfer domain knowledge to learn relational action models across domains in automated planning. Chai *et al.* [32] studied how to apply a GP based multi-task learning method to solve the inverse dynamics problem for a robotic manipulator [32]. Alamgir *et al.* [3] applied multi-task learning techniques to solve brain-computer interfaces problems. In [154], Raykar *et al.* proposed a novel Bayesian multiple-instance learning algorithm, which can automatically identify the relevant feature subset and use inductive transfer for learning multiple, but conceptually related, classifiers, for computer aided design (CAD).

CHAPTER 3

TRANSFER LEARNING VIA DIMENSIONALITY REDUCTION

In this chapter, we aim at proposing a novel and general dimensionality reduction framework for transductive transfer learning. Our proposed framework can be referred to as a feature-representation-transfer approach. As reviewed in Chapter 2.3, most previous feature-based transductive transfer learning methods do not explicitly minimize the distance between different domains in the new feature space, which limits the transferability across domains. In contrast, in the proposed dimensionality reduction framework, one objective is to minimize the distance between domains explicitly, such that the difference between different domain data can be reduced in the latent space. Another objective of the proposed framework is to preserve the properties of the original data as much as possible. Note that this objective is important for the final target learning tasks in the latent space. This chapter is organized as follows, we first introduce our motivation and some preliminaries used in the proposed methods in Chapter 3.1 and Chapter 3.2, respectively. Then we present the proposed dimensionality reduction framework in Chapter 3.3, and three algorithms based on the frameworks known as Maximum Mean Discrepancy Embedding (MMDE), Transfer Component Analysis (TCA) and Semi-supervised Transfer Component Analysis (SSTCA) in Chapters 3.4-3.6. The relationship between the proposed methods to other methods are discussed in Chapter 3.7, respectively. Finally, we apply the proposed methods to two diverse applications, cross-domain WiFi localization and cross-domain text classification, in Chapter 4 and Chapter 5, respectively.

3.1 Motivation

In many real world applications, observed data are controlled by a few latent factors. If the two domains are related to each other, they may share some latent factors (or components). Some of these common latent factors may cause the data distributions between domains to be different, while others may not. Meanwhile, some of these factors may capture the intrinsic structure or discriminative information underlying the original data, while others may not. Our goal is to recover those common latent factors that do not cause much difference between data distributions and do preserve properties of the original data. Then the subspace spanned by these latent factors can be treated as a bridge to make knowledge transfer possible.

We illustrate our idea using a learning-based indoor localization problem as an example, where a client moving in a WiFi environment wishes to use the received signal strength (RSS)

values to locate itself. In an indoor building, RSS values are affected by many latent factors, such as temperature, human movement, building structure, properties of access points (APs), etc. Among these hidden factors, the temperature and the human movement may vary in time, resulting in changes in RSS values over time. However, the building structure and properties of APs are relatively stable. Thus, if we use the latter two factors to represent the RSS data, the distributions of the data collected in different time periods may be close to each other. Thus, this is the latent space where we can ensure a transferring of a learned localization model from one time period to another. Another example is learn to do text classification across domains. If two text-classification domains have different distributions, but are related to each other (e.g., news articles and blogs), there may be some *latent topics* shared by these domains. Some of them may be relatively stable while others may not. If we use the stable latent topics to represent documents, the distance between the distributions of documents in related domains may be small. Then, in the latent space spanned by latent topics, we can transfer the text-classification knowledge.

Motivated by the observations mentioned above, in this chapter, we propose a novel and general dimensionality reduction framework for transductive transfer learning. The key idea is to learn a low-dimensional latent feature space where the distributions of the source domain data and target domain data are close to each other and the data properties can be preserved as much as possible. We then project the data in both domains to this latent feature space, on which we subsequently apply standard learning algorithms to train classification or regression models from labeled source domain data to make predictions on the unlabeled target domain data. Before presenting our proposed dimensionality reduction framework in detail, we first introduce some preliminaries that are used in our proposed solutions based on the framework.

3.2 Preliminaries

3.2.1 Dimensionality Reduction

Dimensionality reduction has been studied widely in the machine learning community. van der Maaten *et al.* [188] gave a recent survey on various dimensionality reduction methods. Traditional dimensionality reduction methods try to project the original data to a low-dimensional latent space while preserving some properties of the original data. Since they cannot guarantee that the distributions between different domain data are similar in the reduced latent space, they cannot directly be used to solve domain adaptation problems. Thus we need to develop a new dimensionality reduction algorithm for domain adaptation. A powerful dimensionality reduction technique is principal component analysis (PCA) and its kernelized version kernel PCA (KPCA). KPCA is an unsupervised method, which aims at extracting a low-dimensional representation of the data by maximizing the variance of the embedding in a kernel space [163].

In has been shown that in many real world applications, PCA or KPCA works pretty well as a preprocess of supervised target tasks [186, 162].

3.2.2 Hilbert Space Embedding of Distributions

Given samples $X = \{x_i\}$ and $Z = \{z_i\}$ drawn from two distributions, there exist many criteria (such as the Kullback-Leibler (KL) divergence) that can be used to estimate their distance. However, many of these estimators are parametric and require an intermediate density estimate. To avoid such a non-trivial task, a non-parametric distance estimate between distributions without density estimation is more desirable.

3.2.3 Maximum Mean Discrepancy

Recently, a non-parametric distance estimate was designed by embedding distributions in a RKHS [172]. Gretton *et al.* [69] introduced the Maximum Mean Discrepancy (MMD) for comparing distributions based on the corresponding RKHS distance. Let the kernel-induced feature map be ϕ. The estimate of MMD between $\{x_1, \ldots, x_{n_1}\}$ and $\{z_1, \ldots, z_{n_2}\}$, which follow distributions $\mathcal{P}$ and $\mathcal{Q}$, respectively, is defined as follow,

$$\text{Dist}(X, Z) = \text{MMD}(X, Z) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \left(\frac{1}{n_1} \sum_{i=1}^{n_1} f(x_i) - \frac{1}{n_2} \sum_{i=1}^{n_2} f(z_i)\right), \tag{3.1}$$

where $\|\cdot\|_{\mathcal{H}}$ is the RKHS norm. By the fact that in a RKHS, function evaluation can be written as $f(x) = \langle \phi(x), f \rangle$, where $\phi(x) : \mathcal{X} \to \mathcal{H}$, the empirical estimate of MMD can be rewritten as follows:

$$\text{Dist}(X, Z) = \text{MMD}(X, Z) = \left\| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi(x_i) - \frac{1}{n_2} \sum_{i=1}^{n_2} \phi(z_i) \right\|_{\mathcal{H}}^2, \tag{3.2}$$

Therefore, the distance between two distributions is simply the distance between the two mean elements in the RKHS. It can be shown that when the RKHS is universal [178], MMD will asymptotically approach zero if and only if the two distributions are the same.

3.2.4 Dependence Measure

Hilbert-Schmidt Independence Criterion

Related to the MMD, the Hilbert-Schmidt Independence Criterion (HSIC) [70] is a simple yet powerful non-parametric criterion for measuring the dependence between the sets X and Y. As its name implies, it computes the Hilbert-Schmidt norm of a cross-covariance operator in the RKHS, which is defined as follows,

$$\text{HSIC}(X,Y) = \|\mathcal{C}_{xy}\|_{\mathcal{H}} = \frac{1}{n^2}\left\|\sum_{i,j=1}^{n,n}[(\phi(x_i) - \frac{1}{n}\sum_{k=1}^{n}\phi(x_k)) \otimes (\psi(y_j) - \frac{1}{n}\sum_{k=1}^{n}\psi(y_k))]\right\|_{\mathcal{H}}, \quad (3.3)$$

where $\otimes$ denotes a tensor product, $\phi : \mathcal{X} \to \mathcal{H}$, $\psi : \mathcal{Y} \to \mathcal{H}$ and $\mathcal{H}$ is a RKHS. An (biased) empirical estimate can be easily obtained from the corresponding kernel matrices, as

$$\text{Dep}(X,Y) = \text{HSIC}(X,Y) = \frac{1}{(n-1)^2}\text{tr}(HKHK_{yy}), \quad (3.4)$$

where K, K_{yy} are kernel matrices defined on X and Y, respectively, $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$ is the centering matrix and n is the number of samples in X and Y. Similar to MMD, it can be shown that if the RKHS is universal, HSIC asymptotically approaches zeros if and only if X and Y are independent [178]. Conversely, a large HSIC value suggests strong dependence.

Embedding via HSIC

In embedding or dimensionality reduction, it is often desirable to preserve the local data geometry while at the same time maximally align the embedding with available side information (such as labels). For example, in Colored Maximum Variance Unfolding (colored MVU) [174], the local geometry is captured in the form of local distance constraints on the target embedding K, while the alignment with the side information (represented as kernel matrix K_{yy}) is measured by the HSIC criterion in (3.4). Mathematically, this leads to the following semi-definite program (SDP) [95]:

$$\begin{aligned} \max_{K\succeq 0} \quad & \text{tr}(HKHK_{yy}) \\ \text{s.t.} \quad & K_{ii} + K_{jj} - 2K_{ij} = d_{ij}^2,\ \forall (i,j) \in \mathcal{N}, \\ & K\mathbf{1} = \mathbf{0}, \end{aligned} \quad (3.5)$$

Where we denote d_{ij} the distance between x_i and x_j in the original feature space. For all i, j, if x_i and x_j are k-nearest neighbors, we denote this by using $(i,j) \in \mathcal{N}$. In particular, (3.5) reduces to Maximum Variance Unfolding (MVU) [198] when no side information is given (i.e., $K_{yy} = I$). In that case, MVU, which is an unsupervised dimensionality reduction, aims at maximizing the variance in the high-dimensional feature space instead of the label dependence.

3.3 A Novel Dimensionality Reduction Framework

Recall that given a lot of labeled source domain data

$$\mathcal{D}_S = \{(x_{S_1}, y_{S_1}), \ldots, (x_{S_{n_1}}, y_{S_{n_S}})\},$$

where $x_{S_i} \in \mathcal{X}$ is the input and $y_{S_i} \in \mathcal{Y}$ is the corresponding output[1], and some unlabeled target domain data

$$\mathcal{D}_T = \{x_{T_1}, \ldots, x_{T_{n_T}}\},$$

where the input x_{T_i} is also in $\mathcal{X}$.

Let $\mathcal{P}(X_S)$ and $\mathcal{Q}(X_T)$ (or $\mathcal{P}$ and $\mathcal{Q}$ in short) be the marginal distributions of the input sets $X_S = \{x_{S_i}\}$ and $X_T = \{x_{T_i}\}$ from the source and target domains, respectively. In general, $\mathcal{P}$ and $\mathcal{Q}$ can be different. Our task is then to predict the labels y_{T_i}'s corresponding to the inputs x_{T_i}'s in the target domain.

As mentioned in Chapter 2.3, a major assumption in most transductive transfer learning methods is that $\mathcal{P} \neq \mathcal{Q}$, but $P(Y_S|X_S) = P(Y_T|X_T)$. However, in many real-world applications, the conditional probability $P(Y|X)$ may also change across domains due to noisy or dynamic factors underlying the observed data. In this chapter, we use new weaker assumption that $\mathcal{P} \neq \mathcal{Q}$, but there exists a transformation ϕ such that $P(\phi(X_S)) \approx P(\phi(X_T))$ and $P(Y_S|\phi(X_S)) \approx P(Y_T|\phi(X_T))$. Standard supervised learning methods can then be applied on the mapped source domain data $\phi(X_S)$, together with the corresponding labels Y_S, to train models for use on the mapped target domain data $\phi(X_T)$.

Hence, a key issue is how to find this transformation ϕ. Since we have no labeled data in the target domain, ϕ cannot be learned by directly minimizing the distance between $P(Y_S|\phi(X_S))$ and $P(Y_T|\phi(X_T))$. Here, we propose a new dimensionality reduction framework to learn ϕ such that (1) the distance between the marginal distributions $P(\phi(X_S))$ and $P(\phi(X_T))$ is small (Chapter 3.3.1), and (2) $\phi(X_S)$ and $\phi(X_T)$ preserve some important properties of X_S and X_T (Chapter 3.3.2). We then assume that such a ϕ satisfies $P(Y_S|\phi(X_S)) \approx P(Y_T|\phi(X_T))$. We believe that domain adaptation under this assumption is more realistic, though also much more challenging. In this thesis, we study domain adaptation under this assumption empirically. We propose a new framework to learn a transformation ϕ, such that $P(\phi(X_S)) \approx P(\phi(X_T))$ and $\phi(X_S)$ and $\phi(X_T)$ preserve some important properties of X_S and X_T, and apply it solve two real-world application successfully.

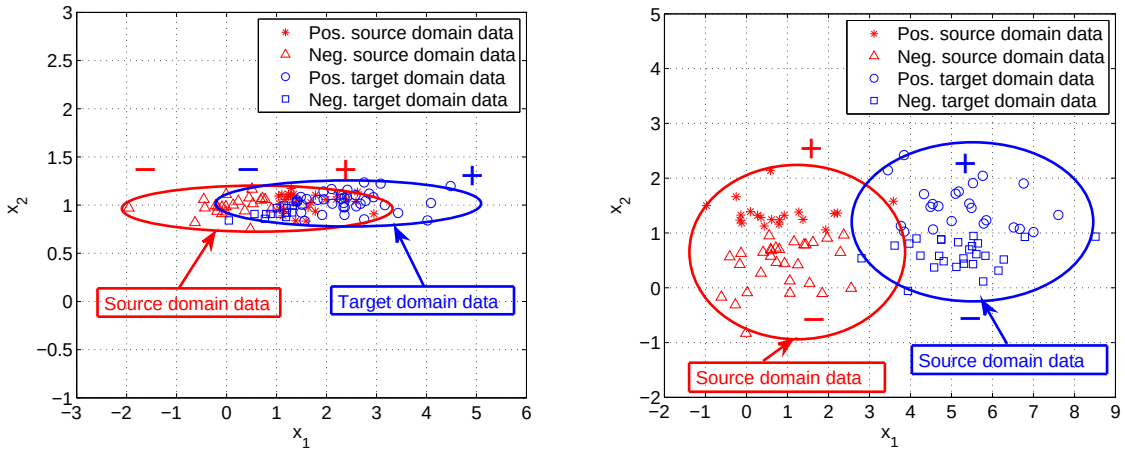
3.3.1 Minimizing Distance between $P(\phi(X_S))$ and $P(\phi(X_T))$

To reduce the distance between domains, a first objective in our dimensionality reduction framework is to discover a desired mapping ϕ by minimizing the distance between $P(\phi(X_S))$ and $P(\phi(X_T))$, denoted by Dist$(P(\phi(X_S)), P(\phi(X_T)))$.

[1]In general, $\mathcal{D}_S$ may contain unlabeled data. Here, for simplicity, we assume that $\mathcal{D}_S$ are fully labeled.

3.3.2 Preserving Properties of X_S and X_T

However, in transductive transfer learning, learning the transformation ϕ by only minimizing the distance between $P(\phi(X_S))$ and $P(\phi(X_T))$ may not be enough. Figure 3.1(a) shows a simple two-dimensional example. Here, the source domain data are shown in red, and the target domain data are in blue. For both the source and target domains, x_1 is the discriminative direction that can separate the positive and negative samples, while x_2 is a noisy dimension with small variance. However, by focusing only on minimizing the distance between $P(\phi(X_S))$ and $P(\phi(X_T))$, one would select the noisy component x_2, which is completely irrelevant to the target supervised learning task.



(a) Only minimizing the distance between $P(\phi(X_S))$ and $P(\phi(X_T))$.

(b) Only maximizing the data variance.

Figure 3.1: Motivating examples for the dimensionality reduction framework.

Hence, besides reducing the distance between the two marginal distributions, ϕ should also preserve data properties that are useful for the target supervised learning task. An obvious choice is to maximally preserve the data variance, as is performed by the well-known PCA and KPCA (Chapter 3.2.1).

However, focusing only on the data variance is again not desirable in domain adaptation. An example is shown in Figure 3.1(b), where the direction with the largest variance (x_1) cannot be used to reduce the distance of distributions across domains and is not useful in boosting the performance for domain adaptation.

Thus, an effective dimensionality reduction framework for transfer learning should satisfy that in the reduced latent space,

1. distance between data distributions across domains should be reduced;

2. data properties should be preserved as much as possible.

Then standard supervised techniques can be applied in the reduced latent space to train classification or regression models from source domain labeled data for use in the target domain. The proposed framework is summarized in Algorithm 3.1.

Algorithm 3.1 A dimensionality reduction framework for transfer learning

Require: A labeled source domain data set $\mathcal{D}_S = \{(x_{S_i}, y_{S_i})\}$, an unlabeled target domain data set $\mathcal{D}_T = \{x_{T_i}\}$.
Ensure: Predicted labels Y_T of the unlabeled data X_T in the target domain.
1: Learn a transformation mapping ϕ, such that Dist$(\phi(X_S), \phi(X_T))$ is small, $\phi(X_S)$ and $\phi(X_T)$ can preserve properties of X_S and X_T, respectively.
2: Train a classification or regression model f on $\phi(X_S)$ with the corresponding labels Y_S.
3: For unlabeled data x_{T_i}'s in $\mathcal{D}_T$, map them to the latent space to get new representations $\phi(x_{T_i})$'s. Then, use the model f to make predictions $f(\phi(x_{T_i}))$'s.
4: **return** ϕ and $f(\phi(x_{T_i}))$'s.

As can be seen, the first step is the key step, because once the transformation mapping ϕ is learned in the first step, one just need to apply existing machine learning and data mining methods on the mapped data $\phi(X_S)$ with the corresponding labels Y_S for training models for use in the target domain. Thus, one advantage of this framework is that most existing machine learning methods can be easy integrated into this framework. Another advantage is that it works for diverse machine learning tasks, such as classification and regression problems.

3.4 Maximum Mean Discrepancy Embedding (MMDE)

As introduced in Chapter 3.2.3, on using the empirical measure of MMD(3.2), the distance between two distributions $P(\phi(X_S))$ and $P(\phi(X_T))$ can be empirically measured by the (squared) distance between the empirical means of the two domains:

$$\text{Dist}(\phi(X_S), \phi(X_T)) = \left\| \frac{1}{n_S} \sum_{i=1}^{n_S} \varphi \circ \phi(x_{S_i}) - \frac{1}{n_T} \sum_{i=1}^{n_T} \varphi \circ \phi(x_{T_i}) \right\|_{\mathcal{H}}^2, \tag{3.6}$$

for some $\varphi \in \mathcal{H}$, which is the feature map induced by a universal kernel. Note that in practice, the corresponding kernel may not need to be universal [173]. Furthermore, we denote $\varphi(\phi(x))$ by $\varphi \circ \phi(x)$. Therefore, a desired nonlinear mapping ϕ can be found by minimizing the above quantity. However, φ is usually highly nonlinear. As a result, a direct optimization of (3.6) with respect to ϕ may be intractable and can get stuck in poor local minima.

3.4.1 Kernel Learning for Transfer Latent Space

Instead of finding the transformation ϕ explicitly, in this section, we propose a kernel based dimensionality reduction method called Maximum Mean Discrepancy Embedding (MMDE)

[135] to construct the ϕ implicitly. Before describing the details of MMDE, we first introduce a lemma shown as follows.

Given that $\varphi \in \mathcal{H}$, we can get the following lemma:

Lemma 1. *Let φ be the feature map induced by a kernel. Then $\varphi \circ \phi$ is also the feature map induced by a kernel for any arbitrary map ϕ.*

Proof. Denote $x' = \phi(x)$, and $k(x_i, x_j) = \langle \varphi \circ \phi(x_i), \varphi \circ \phi(x_j) \rangle$. For any finite sample $X = \{x_1, ..., x_n\}$, one can find their corresponding sample in the latent space, $X' = \{\phi(x_1), ..., \phi(x_n)\} = \{x'_1, ..., x'_n\}$. Thus, $k(x_i, x_j) = \langle \varphi \circ \phi(x_i), \varphi \circ \phi(x_j) \rangle = \langle \varphi(x'_i), \varphi(x'_j) \rangle$. Since φ is the feature map induced by a kernel, thus the corresponding matrix K, where $K_{ij} = k(x_i, x_j)$ is positive semi-definite for sample X'. Based on the Mercer's theory [164], $k = (.,.)$ is a valid kernel function. Therefore, $\varphi \circ \phi$ is also the feature map induced by a kernel for any arbitrary map ϕ. □

Therefore, our goal becomes finding the feature map $\varphi \circ \phi$ of some kernel such that (3.6) is minimized. Here we translate the problem of learning ϕ to the problem of learning $\varphi \circ \phi$, which makes the optimization problem tractable. Moreover, by using the kernel trick, we can write $\langle \varphi \circ \phi(x_i), \varphi \circ \phi(x_j) \rangle = k(x_i, x_j)$, where k is the corresponding kernel. Equation (3.6) can be written in terms of the kernel matrices defined by k, as:

$$\begin{aligned} \text{Dist}(X'_S, X'_T) &= \frac{1}{n_S^2} \sum_{i,j=1}^{n_S} k(x_{S_i}, x_{S_j}) + \frac{1}{n_T^2} \sum_{i,j=1}^{n_T} k(x_{T_i}, x_{T_j}) - \frac{2}{n_S n_T} \sum_{i,j=1}^{n_S, n_T} k(x_{S_i}, x_{T_j}) \\ &= \text{tr}(KL), \end{aligned} \tag{3.7}$$

where

$$K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S}^T & K_{T,T} \end{bmatrix} \in \mathbb{R}^{(n_S+n_T)\times(n_S+n_T)} \tag{3.8}$$

is a composite kernel matrix with K_S and K_T being the kernel matrices defined by k on the data in the source and target domains, respectively, and $L = [L_{ij}] \succeq 0$ with

$$L_{ij} = \begin{cases} \frac{1}{n_S^2} & x_i, x_j \in X_S, \\ \frac{1}{n_T^2} & x_i, x_j \in X_T, \\ -\frac{1}{n_S n_T} & \text{otherwise.} \end{cases}$$

In the transductive setting[2], we can learn the kernel matrix K instead of the universal kernel k by minimizing the distance (measured w.r.t. the MMD) between the projected source and

[2]Note that, here, the word "transductive" is from the *transductive learning* [90] setting in traditional machine learning, which refers to the situation where all test data are required to be seen at training time, and that the learned model cannot be reused for future data. Thus, when some new test data arrive, they must be classified together with all existing data. Recall that in *transductive transfer learning*, the term "transductive" is to emphasize the concept that in this type of transfer learning, the tasks must be the same and there must be some unlabeled data available in the target domain.

target domain data. Then, similar to MVU as introduce in Chapter 3.2.4, we can apply PCA on the resultant kernel matrix to reconstruct the low-dimensional representations X'_S and X'_T.

However, as mentioned in Chapter 3.3.2, for transductive transfer learning, it may not be sufficient to learning the transformation by only minimizing the distance between the projected source and target domain data. Thus, besides minimizing the trace of KL in 3.7, in MMDE, we also have the following objectives/constraints to preserve the properties of the original data.

1. The trace of K is maximized, which aims to preserve as much as variance in the feature space, as proposed in MVU.
2. The distance is preserved, i.e., $K_{ii} + K_{jj} - 2K_{ij} = d_{ij}^2$ for all i, j such that $(i, j) \in \mathcal{N}$, as proposed in MVU and colored MVU.
3. The embedded data are centered, which is a standard condition for PCA applied on the resultant kernel matrix.

As mentioned in Chapter 3.2.4, maximizing the trace of K is equivalent to maximizing the variance of the embedded data. The distance preservation constraint is motivated by MVU, which can make the kernel matrix learning more tractable. The centering constraint is used for the post-process PCA on the resultant kernel matrix K.

The optimization problem of MMDE can then be written as:

$$\begin{aligned} \min_{K \succeq 0} \quad & \text{tr}(KL) - \lambda \text{tr}(K) \\ \text{s.t.} \quad & K_{ii} + K_{jj} - 2K_{ij} = d_{ij}^2, \ \forall (i, j) \in \mathcal{N}, \\ & K\mathbf{1} = \mathbf{0}, \end{aligned} \tag{3.9}$$

where the first term in the objective minimizes the distance between distributions, while the second term maximizes the variance in the feature space, and $\lambda \geq 0$ is a tradeoff parameter. $\mathbf{0}$ and $\mathbf{1}$ are vectors of zeros and ones. Computationally, this leads to a SDP involving K [95]. After learning the kernel matrix K in (3.9), PCA is applied on the resultant kernel matrix and select the leading eigenvectors to reconstruct the desired mapping $\varphi \circ \phi$ implicitly to map data into a low-dimensional latent space across domains, X'_S and X'_T. Note that, the optimization problem (3.9) is similar to colored MVU as introduced in Chapter 3.2.4. However, there are two major differences between MMDE and colored MVU. First, the L matrix in colored MVU is a kernel matrix that encodes label information of the data, while the L in MMDE can be treated as a kernel matrix that encode distribution information of different data sets. Second, besides minimizing the trace of KL, MMDE also aims to unfold the high dimensional data by maximizing the trace of K. In Chapter 3.7, we will discuss the relationship between these methods in detail.

3.4.2 Make Predictions in Latent Space

After obtaining the new representations X'_S and X'_T, we can train a classification or regression model f from X'_S with the corresponding labels Y_S. This can then be used to make predictions on X'_T. However, since we do not learn a mapping explicitly to project the original data X_S and X_T to the embeddings X'_S and X'_T, respectively, for out-of-sample data in the target domain, we need to apply other techniques to make predictions on the out-of-sample test data. Here, we use the method of harmonic functions [234], which is defined in (3.10), to estimate the labels of the new test data in the target domain.

$$\widetilde{f}_i = \frac{\sum_{j \in \mathcal{N}} w_{ij} f_j}{\sum_{j \in \mathcal{N}} w_{ij}}, \tag{3.10}$$

where $\mathcal{N}$ is the set of k nearest neighbors of $\widetilde{x}_{T_i}$ in X_T, w_{ij} is the similarity between $\widetilde{x}_{T_i}$ and x_{T_j}, which can be obtained based on Euclid distance or other similarity measure, and f_j's is the predicted labels of x_{T_j}'s. The MMDE algorithm for transfer learning is summarized in Algorithm 3.2.

Algorithm 3.2 Transfer learning via Maximum Mean Discrepancy Embedding (MMDE)

Require: A labeled source domain data set $\mathcal{D}_S = \{(x_{S_i}, y_{S_i})\}$, an unlabeled target domain data set $\mathcal{D}_T = \{x_{T_i}\}$ and $\lambda > 0$.
Ensure: Predicted labels Y_T of the unlabeled data X_T in the target domain.
1: Solve the SDPproblem in (3.9) to obtain a kernel matrix K.
2: Apply PCA to the learned K to get new representations $\{x'_{S_i}\}$ and $\{x'_{T_i}\}$ of the original data $\{x_{S_i}\}$ and $\{x_{T_i}\}$, respectively.
3: Learn a classifier or regressor $f : x'_{S_i} \rightarrow y_{S_i}$
4: For unlabeled data x_{T_i}'s in $\mathcal{D}_T$, the learned classifier or regressor to predict the labels of $\mathcal{D}_T$, as: $y_{T_i} = f(x'_{T_i})$.
5: For new test data $\widetilde{x}_T$'s in the target domain, use harmonic functions with $\{x_{T_i}, f(x'_{T_i})\}$ to make predicts.
6: **return** Predicted labels Y_T.

As can be seen in the algorithm, the key step of MMDE is to apply a SDP solver to the optimization problem in (3.9). In general, as there are $O((n_S + n_T)^2)$ variables in $\widetilde{K}$, the overall time complexity is $O((n_S + n_T)^{6.5})$ [129]. In the second step, standard PCA is applied on the learned kernel matrix to construct latent representations X'_S and X'_T of X_S and X_T, respectively. A model is then trained on X'_S with the corresponding labels Y_S. Finally, the method of harmonic functions introduced in (3.10) is used to make predictions on unseen target domain data.

3.4.3 Summary

In MMDE, we translate the problem of learning the transformation mapping ϕ in the dimensionality reduction framework to a kernel matrix learning problem. We then apply the standard

PCAmethod on the resultant kernel matrix to reconstruct low-dimensional representations for different domain data. This translation makes the problem of learning ϕ tractable. Furthermore, since MMDE learns the kernel matrix from the data automatically, it can fit the data more perfectly. However, there are several limitations of MMDE. Firstly, it cannot generalize to unseen data. For any unseen target domain data, we need to apply the MMDE method on the source domain and new target domain data to reconstruct their new representations. In order to make predictions on unseen target domain unlabeled data, other techniques, such as the method of harmonic functions need to be used. Secondly, the criterion (3.9) in MMDE, requires K to be positive semi-definite and the resultant kernel learning problem has to be solved by expensive SDP solvers. The overall time complexity is $O((n_S + n_T)^{6.5})$ in general. This becomes computationally prohibitive even for small-sized problems. Finally, in order to construct low-dimensional representations of X'_S and X'_T, the obtained K has to be further post-processed by PCA, which may discards potentially useful information in K.

3.5 Transfer Component Analysis (TCA)

In order to overcome the limitations of MMDE as described in the previous section, in this section, we propose an efficient framework to find the nonlinear mapping $\varphi \circ \phi$ based on empirical kernel feature extraction. It avoids the use of SDP and thus its high computational burden. Moreover, the learned kernel can be generalized to out-of-sample data directly. Besides, instead of using a two-step approach as in MMDE, we propose a unified kernel learning method which utilizes an explicit low-rank representation.

3.5.1 Parametric Kernel Map for Unseen Data

First, note that the kernel matrix K in (3.8) can be decomposed as $K = (KK^{-1/2})(K^{-1/2}K)$, which is often known as the empirical kernel map [163]. Consider the use of a $(n_S + n_T) \times m$ matrix $\widetilde{W}$ that transforms the empirical kernel map features to a m-dimensional space (where $m \ll n_S + n_T$). The resultant kernel matrix[3] is then

$$\widetilde{K} = (KK^{-1/2}\widetilde{W})(\widetilde{W}^\top K^{-1/2}K) = KWW^\top K, \tag{3.11}$$

where $W = K^{-1/2}\widetilde{W} \in \mathbb{R}^{(n_S+n_T)\times m}$. In particular, the corresponding kernel evaluation between any two patterns x_i and x_j is given by

$$\widetilde{k}(x_i, x_j) = k_{x_i}^\top WW^\top k_{x_j}, \tag{3.12}$$

[3] As is common practice, one can ensure that the kernel matrix K is positive definite by adding a small $\epsilon > 0$ to its diagonal [135].

where $k_x = [k(x_1, x), \ldots, k(x_{n_S+n_T}, x)]^\top \in \mathbb{R}^{n_S+n_T}$. Hence, this kernel $\widetilde{k}$ facilitates a readily parametric form for out-of-sample kernel evaluations.

Moreover, on using the definition of $\widetilde{K}$ in (3.11), the MMD distance between the empirical means of the two domains X'_S and X'_T can be rewritten as:

$$\text{Dist}(X'_S, X'_T) = \text{tr}((KWW^\top K)L) = \text{tr}(W^\top KLKW). \tag{3.13}$$

In minimizing objective (3.13), a regularization term $\text{tr}(W^\top W)$ is usually needed to control the complexity of W. As will be shown later in this section, this regularization term can also avoid the rank deficiency of the denominator in the generalized eigenvalue decomposition.

Besides reducing the distance between the two marginal distributions in (3.13), we also need to preserve the data properties such as the data variance using the parametric kernel map, as is performed by the well-known PCA and KPCA (Chapter 3.2.1). Note from (3.11) that the embedding of the data in the latent space is $W^\top K$, where the ith column $[W^\top K]_i$ provides the embedding coordinates of x_i. Hence, the variance of the projected samples is $W^\top KHKW$, where $H = I_{n_1+n_2} - \frac{1}{n_1+n_2}\mathbf{1}\mathbf{1}^\top$ is the centering matrix, $\mathbf{1} \in \mathbb{R}^{n_1+n_2}$ is the column vector with all ones, and $I_{n_1+n_2} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$ is the identity matrix.

3.5.2 Unsupervised Transfer Component Extraction

Combining the parametric kernel representations for distance between distributions and data variance in the previous section, we develop a new dimensionality reduction method such that in the latent space spanned by the learned components, the variance of the data can be preserved as much as possible and the distance between different distributions across domains can be reduced. The kernel learning problem then becomes:

$$\begin{aligned} \min_W \quad & \text{tr}(W^\top KLKW) + \mu\, \text{tr}(W^\top W) \\ \text{s.t.} \quad & W^\top KHKW = I_m, \end{aligned} \tag{3.14}$$

where $\mu > 0$ is a trade-off parameter, and $I_m \in \mathbb{R}^{m\times m}$ is the identity matrix. For notational simplicity, we will drop the subscript m from I_m in the sequel. Though this optimization problem involves a non-convex norm constraint $W^\top KHKW = I$, it can still be solved efficiently by the following trace optimization problem:

Proposition 1. *The optimization problem (3.14) can be re-formulated as*

$$\min_W \; tr((W^\top KHKW)^\dagger W^\top (KLK + \mu I)W), \tag{3.15}$$

or

$$\max_W \; tr((W^\top (KLK + \mu I)W)^{-1} W^\top KHKW). \tag{3.16}$$

Proof. The Lagrangian of (3.14) is

$$\text{tr}(W^\top(KLK + \mu I)W) - \text{tr}((W^\top KHKW - I)Z), \tag{3.17}$$

where Z is a diagonal matrix containing Lagrange multipliers. Setting the derivative of (3.17) w.r.t. W to zero, we have

$$(KLK + \mu I)W = KHKWZ. \tag{3.18}$$

Multiplying both sides on the left by W^T, and then on substituting it into (3.17), we obtain (3.15). Since the matrix $KLK+\mu I$ is non-singular, we obtain an equivalent trace maximization problem (3.16). □

Similar to kernel Fisher discriminant analysis (KFD) [126, 216], the W solution in (3.16) are the m leading eigenvectors of $(KLK+\mu I)^{-1}KHK$, where $m \leq n_S+n_T-1$. In the sequel, this will be referred to as *Transfer Component Analysis* (TCA), and the extracted components are called the *transfer components*.

The TCA algorithm for transfer learning is summarized in Algorithm 3.3.

Algorithm 3.3 Transfer learning via Transfer Component Analysis (TCA).

Require: Source domain data set $\mathcal{D}_S = \{(x_{S_i}, y_{S_i})\}_{i=1}^{n_S}$, and target domain data set $\mathcal{D}_T = \{x_{T_j}\}_{j=1}^{n_T}$.

Ensure: Transformation matrix W and predicted labels Y_T of the unlabeled data X_T in the target domain..

1: Construct kernel matrix K from $\{x_{S_i}\}_{i=1}^{n_S}$ and $\{x_{T_j}\}_{j=1}^{n_T}$ based on (3.8), matrix L from (3.7), and centering matrix H.
2: Compute the matrix $(KLK + \mu I)^{-1}KHK$, where I is the identity matrix.
3: Do Eigen-decomposition and select the m leading eigenvectors to construct the transformation matrix W.
4: Map the data x_{S_i}'s and x_{T_j}'s to x'_{S_i}'s and x'_{T_j}'s via using $X'_S = [K_{S,S}\ K_{S,T}]W$ and $X'_T = [K_{T,S}\ K_{T,T}]W$, respectively.
5: Train a model f on x_{S_i}'s with y_{S_i}'s.
6: For new test data $\widetilde{x}_T$ from the target domain, $\widetilde{x}'_T = \kappa W$, where κ is a row vector, and $\kappa_i = k(\widetilde{x}_T, x_t)$, $t = 1, ..., n_S, n_S + 1, ..., n_S + n_T$.
7: **return** transformation matrix W and $f(\widetilde{x}'_T)$'s.

As can be seen in the algorithm, the key of TCA is the second step, to do eigen-decomposition on the matrix $(KLK + \mu I)^{-1}KHK$ to find m leading eigenvectors to construct the transformation W. In general, it takes only $O(m(n_S + n_T)^2)$ time when m nonzero eigenvectors are to be extracted [175], which is much more efficient than MMDE. Furthermore, once the W is learned , for new test data $\widetilde{x}_T$ from the target domain, we can use W to map it to the latent space directly. Thus, TCA can be generalized to out-of-sample data.

3.5.3 Experiments on Synthetic Data

As described in previous sections, in TCA, there are two main objectives: minimizing the distance between domains and maximizing the data variance in the latent space. In this section, we perform experiments on synthetic data to demonstrate the effectiveness of these two objectives of TCA in learning a 1D latent space from the 2D data. For TCA, we use the linear kernel on inputs, and fix $\mu = 1$. For fully testing the effectiveness of TCA, we will conduct more detailed experiments by comparing with other exciting methods in two real-world datasets in Chapter 4 and Chapter 5, respectively.

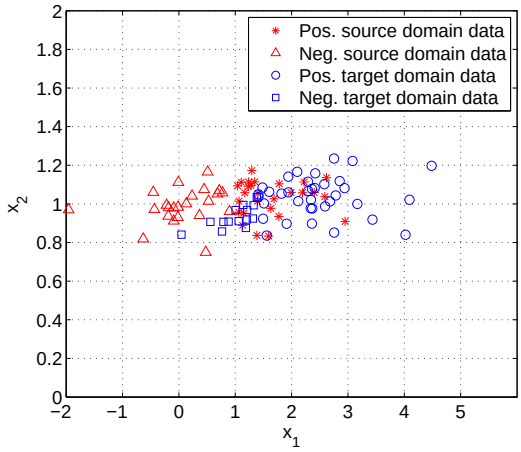
Only Minimizing Distance between Distributions

As discussed in Chapter 3.3.2, it is not desirable to learn the transformation ϕ by only minimizing the distance between the marginal distributions $P(\phi(X_S))$ and $P(\phi(X_T))$. Here, we illustrate this by using the synthetic data from the example in Figure 3.1(a) (which is also reproduced in Figure 3.2(a)). We compare TCA with the method of stationary subspace analysis (SSA) as introduced in Chapter 2.3, which is an empirical method to find an identical stationary latent space of the source and target domain data.

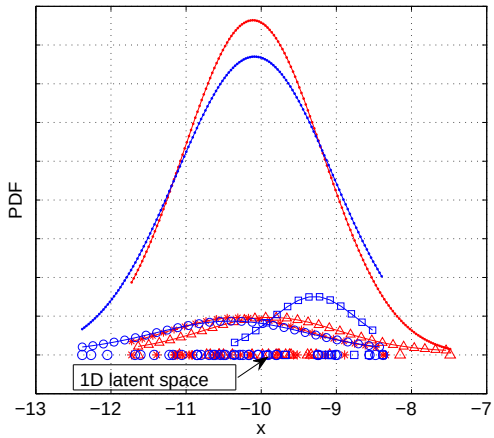
As can be seen from Figures 3.2(b), the distance between distributions of different domain data in the one-dimensional space learned by SSA is small. However, the positive and negative samples are overlapped together in the latent space, which is not useful for making predictions on the mapped target domain data. On the other hand, as can be seen from Figure 3.2(c), though the distance between distributions of different domain data in the latent space learned by TCA is larger than that learned by SSA, the two classes are now more separated. We further apply the one-nearest-neighbor (1-NN) classifier to make predictions on the target domain data in the original 2D space, and latent spaces learned by SSA and TCA. As can be seen from Figures 3.2(a), 3.2(b) and 3.2(c), TCA leads to significantly better accuracy than SSA.

Only Maximizing the Data Variance

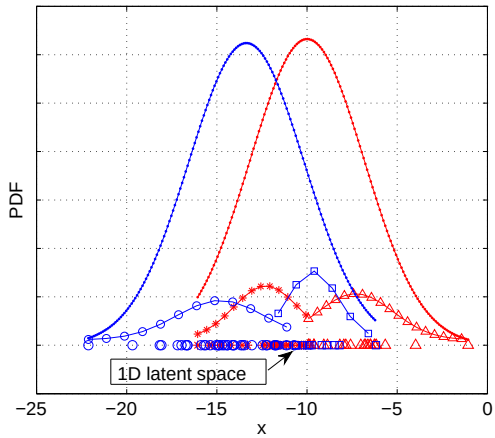
As discussed in Chapter 3.3.2, learning the transformation ϕ by only maximizing the data variance may not be useful in domain adaptation. Here, we reproduce Figure 3.1(b) in Figure 3.3(a). As can be seen from Figures 3.3(b) and 3.3(c), the variance of the mapped data in the 1D space learned by PCA is very large. However, the distance between the mapped data across different domains is still large and the positive and negative samples are overlapped together in the latent space, which is not useful for domain adaptation. On the other hand, though the variance of the mapped data in the 1D space learned by TCA is smaller than that learned by PCA, the distance between different domain data in the latent space is reduced and the positive and negative samples are more separated in the latent space.



(a) Data set 1 (acc: 82%).



(b) 1D projection by SSA (acc: 60%).

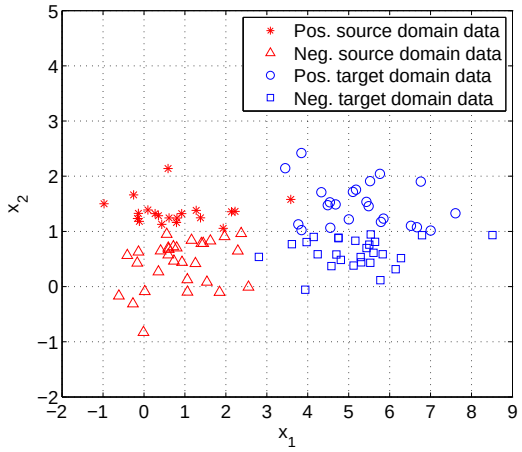


(c) 1D projection by TCA (acc: 86%).

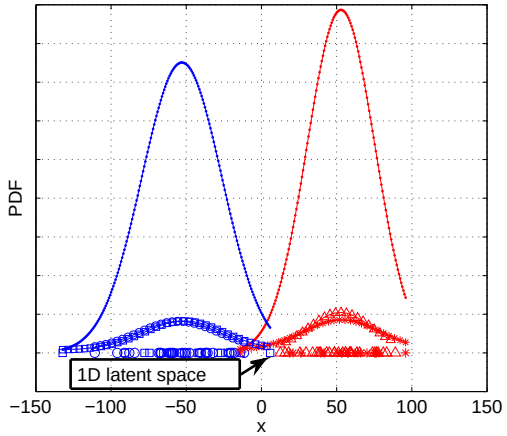
Figure 3.2: Illustrations of the proposed TCA and SSTCA on synthetic dataset 1. Accuracy of the 1-NN classifier in the original input space / latent space is shown inside brackets.

3.5.4 Summary

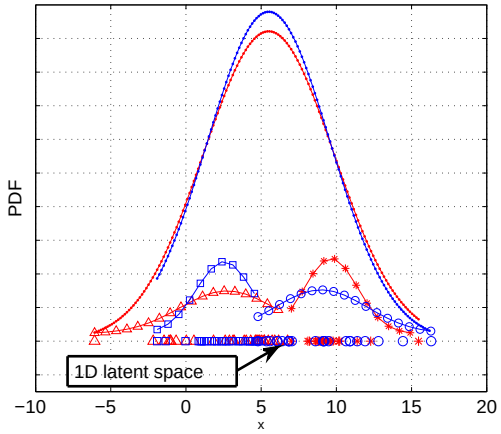
In TCA, we propose to learn a low-rank parametric kernel for transfer learning instead of the entire kernel matrix. Indeed, the parametric kernel is a composing kernel that consists of an empirical kernel and a linear transformation. Parameter values of the empirical kernel needs to be tuned by human experience or by using the cross-validation method, which may be sensitive to different application areas. However, compared to MMDE, TCA has two advantages, (1) It is much more efficient. As can be seen from Algorithm 3.3, TCA only requires a simple and efficient eigenvalue decomposition, which takes only $O(m(n_S + n_T)^2)$ time when m nonzero eigenvectors are to be extracted. (2) It can be generalized to out-of-sample target domain domain naturally. Once the transformation W is learned, data from the source and target domain, including unseen data, can be mapped to the latent space directly.



(a) Data set 2 (accuracy: 50%).



(b) 1D projection by PCA (acc: 48%).



(c) 1D projection by TCA (acc: 82%).

Figure 3.3: Illustrations of the proposed TCA and SSTCA on synthetic dataset 2. Accuracy of the 1-NN classifier in the original input space / latent space is shown inside brackets.

3.6 Semi-Supervised Transfer Component Analysis (SSTCA)

As Ben-David *et al.* [16], Blitzer *et al.* [24] and Mansour *et al.* [121] mentioned in their works, repectively, a good representation should (1) reduce the distance between the distributions of the source and target domain data; and (2) minimize the empirical error on the labeled data in the source domain[4]. As shown in Figure 3.4, the direction with the largest variance (x_1) is orthogonal to the discriminative direction (x_2). As a result, the transfer components learned by TCA may not be useful for the target classification task. A solution to overcome this is to encode the source domain label information into the embedding learning, such that the learned components are discriminative to labels in the source domain and can be used to reduce the distance between the source and target domains as well. However, the unsupervised TCA proposed in Chapter 3.5 does not consider the label information in learning the components.

[4]Recall that in our setting, there is no labeled data in the target domain.

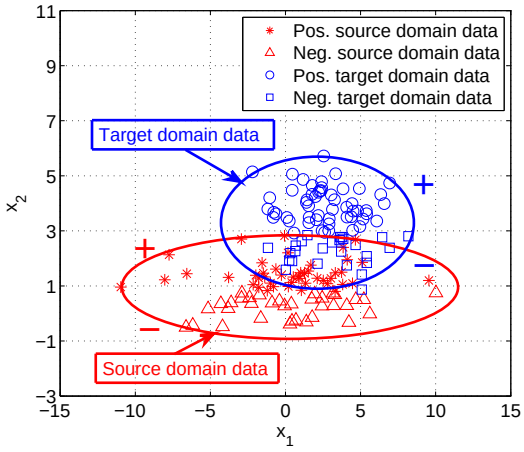


Figure 3.4: The direction with the largest variance is orthogonal to the discriminative direction

Moreover, in many real-world applications (such as WiFi localization), there exists an intrinsic low-dimensional manifold underlying the high-dimensional observations. The effective use of manifold information is an important component in many semi-supervised learning algorithms [34, 233].

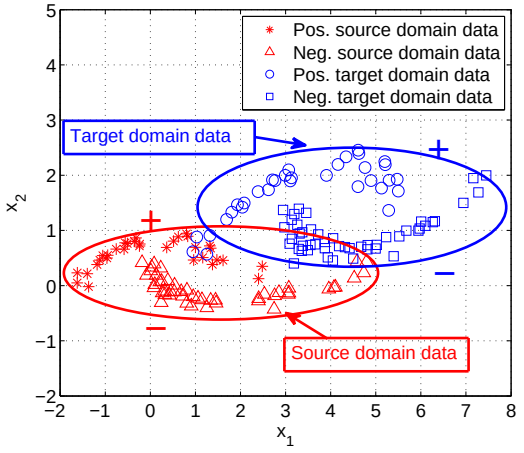


Figure 3.5: There exists an intrinsic manifold structure underlying the observed data.

In this section, we extend the *unsupervised* TCA in Chapter 3.5 to the semi-supervised learning setting. Motivated by the kernel target alignment [43], a representation that maximizes its dependence with the data labels may lead to better generalization performance. Hence, we can maximize the label dependence instead of minimizing the empirical error (Chapter 3.6.1). Moreover, we encode the manifold structure into the embedding learning so as to propagate label information from the labeled (source domain) data to the unlabeled (target domain) data (Chapter 3.6.1). Note that in traditional semi-supervised learning settings [233, 34], the labeled and unlabeled data are from the same domain. However, in the context of domain adaptation here, the labeled and unlabeled data are from different domains.

3.6.1 Optimization Objectives

In this section, we delineate three desirable properties for this semi-supervised embedding, namely, (1) maximal alignment of distributions between the source and target domain data in the embedded space; (2) high dependence on the label information; and (3) preservation of the local geometry.

Objective 1: Distribution Matching

As in the *unsupervised* TCA, our first objective is to minimize the MMD (3.13) between the source and target domain data in the embedded space.

Objective 2: Label Dependence

Our second objective is to maximize the dependence (measured w.r.t. HSIC) between the embedding and labels. Recall that while the source domain data are fully labeled, the target domain data are unlabeled. We propose to maximally align the embedding (which is represented by $\widetilde{K}$ in (3.11)) with

$$\tilde{K}_{yy} = \gamma K_l + (1-\gamma) K_v, \tag{3.19}$$

where $\gamma \geq 0$. Here,

$$[K_l]_{ij} = \begin{cases} k_{yy}(y_i, y_j) & i,j \leq n_S, \\ 0 & \text{otherwise}, \end{cases} \tag{3.20}$$

serves to maximize label dependence on the labeled data, while

$$K_v = I, \tag{3.21}$$

serves to maximize the variance on both the source and target domain data, which is in line with MVU [198]. By substituting $\widetilde{K}$ (3.11) and $\tilde{K}_{yy}$ (3.19) into HSIC (3.4), our objective is thus to maximize

$$\text{tr}(H(KWW^\top K)H\tilde{K}_{yy}) = \text{tr}(W^\top K H \tilde{K}_{yy} H K W). \tag{3.22}$$

Note that γ is a tradeoff parameter that balances the label dependence and data variance terms. Intuitively, if there are sufficient labeled data in the source domain, the dependence between features and labels can be estimated more precisely via HSIC, and a large γ may be used. Otherwise, when there are only a few labeled data in the source domain and a lot of unlabeled data in the target domain, we may use a small γ. Empirically, simply setting $\gamma = 0.5$ works well on all the data sets. The sensitivity of the performance to γ will be studied in more detail in Chapters 4 and 5.

Objective 3: Locality Preserving

As reviewed in Chapters 3.2.4 and 3.4, Colored MVU and MMDE preserve the local geometry of the manifold by enforcing distance constraints on the desired kernel matrix K. More specifically, let $\mathcal{N} = \{(x_i, x_j)\}$ be the set of sample pairs that are k-nearest neighbors of each other, and $d_{ij} = \|x_i - x_j\|$ be the distance between x_i, x_j in the original input space. For each (x_i, x_j) in $\mathcal{N}$, a constraint $K_{ii} + K_{jj} - 2K_{ij} = d_{ij}^2$ will be added to the optimization problem. Hence, the resultant SDP will typically have a very large number of constraints.

To avoid this problem, we make use of the locality preserving property of the Laplacian Eigenmap [13]. First, we construct a graph with the affinity $m_{ij} = \exp(-d_{ij}^2/2\sigma^2)$ if x_i is one of the k nearest neighbors of x_j, or vice versa. Let $M = [m_{ij}]$. The graph Laplacian matrix is $\mathcal{L} = D - M$, where D is the diagonal matrix with entries $d_{ii} = \sum_{j=1}^{n} m_{ij}$. Intuitively, if x_i, x_j are neighbors in the input space, the distance between the embedding coordinates of x_i and x_j should be small. Note from (3.11) that the embedding of the data in $\mathbb{R}^m$ is $W^\top K$, where the ith column $[W^\top K]_i$ provides the embedding coordinates of x_i. Hence, our third objective is to minimize

$$\sum_{(i,j)\in\mathcal{N}} m_{ij} \left\|[W^\top K]_i - [W^\top K]_j\right\|^2 = \text{tr}(W^\top K\mathcal{L}KW). \tag{3.23}$$

3.6.2 Formulation and Optimization Procedure

In this section, we present how to combine the three objectives to find a W that maximizes (3.22), while simultaneously minimizes (3.13) and (3.23). The final optimization problem can be written as

$$\begin{aligned} \min_{W} \quad & \text{tr}(W^\top KLKW) + \mu\,\text{tr}(W^\top W) + \frac{\lambda}{n^2}\,\text{tr}(W^\top K\mathcal{L}KW) \\ \text{s.t.} \quad & W^\top KH\widetilde{K}_{yy}HKW = I, \end{aligned} \tag{3.24}$$

where $\lambda \geq 0$ is another tradeoff parameter, and $n^2 = (n_S + n_T)^2$ is a normalization term. For simplicity, we use λ to denote $\frac{\lambda}{n^2}$ in the rest of this paper. Similar to the *unsupervised* TCA, (3.24) can be formulated as the following quotient trace problem:

$$\max_{W}\ \text{tr}\{(W^\top K(L + \lambda\mathcal{L})KW + \mu I)^{-1}(W^\top KH\widetilde{K}_{yy}HKW)\}, \tag{3.25}$$

In the sequel, this will be referred to as Semi-Supervised Transfer Component Analysis (SSTCA). It is well-known that (3.25) can be solved by eigendecomposing

$$(K(L + \lambda\mathcal{L})K + \mu I)^{-1}KH\widetilde{K}_{yy}HK.$$

The procedure for both the unsupervised and semi-supervised TCA is summarized in Algorithm 3.4.

Algorithm 3.4 Transfer learning via Semi-Supervised Transfer Component Analysis (SSTCA).

Require: Source domain data set $\mathcal{D}_S = \{(x_{S_i}, y_{S_i})\}_{i=1}^{n_S}$, and target domain data set $\mathcal{D}_T = \{x_{T_j}\}_{j=1}^{n_T}$.

Ensure: Transformation matrix W and predicted labels Y_T of the unlabeled data X_T in the target domain..

1: Construct kernel matrix K from $\{x_{S_i}\}_{i=1}^{n_S}$ and $\{x_{T_j}\}_{j=1}^{n_T}$ based on (3.8), matrix L from (3.7), and centering matrix H.
2: Compute the matrix $(K(L + \lambda\mathcal{L})K + \mu I)^{-1}KH\widetilde{K}_{yy}HK$.
3: Do eigen-decomposition and select the m leading eigenvectors to construct the transformation matrix W.
4: Map the data x_{S_i}'s and x_{T_j}'s to x'_{S_i}'s and x'_{T_j}'s via using $X'_S = [K_{S,S}\ K_{S,T}]W$ and $X'_T = [K_{T,S}\ K_{T,T}]W$, respectively.
5: Train a model f on x_{S_i}'s with y_{S_i}'s.
6: For new test data $\widetilde{x}_T$ from the target domain, $\widetilde{x}'_T = \kappa W$, where κ is a row vector, and $\kappa_i = k(\widetilde{x}_T, x_t)$, $t = 1, ..., n_S, n_S + 1, ..., n_S + n_T$.
7: **return** transformation matrix W and $f(\widetilde{x}'_T)$'s.

As can be seen in the algorithm, it is very similar to that of TCA. The only difference between these algorithms is that in SSTCA, the matrix need to be eigen-decomposed is $(K(L + \lambda\mathcal{L})K + \mu I)^{-1}KH\widetilde{K}_{yy}HK$ instead of $(KLK + \mu I)^{-1}KHK$. However, the overall time complexity is still $O(m(n_S + n_T)^2)$. In addition, SSTCA is also easy to be generalized to out-of-sample data, because the transformation W is learned explicitly.

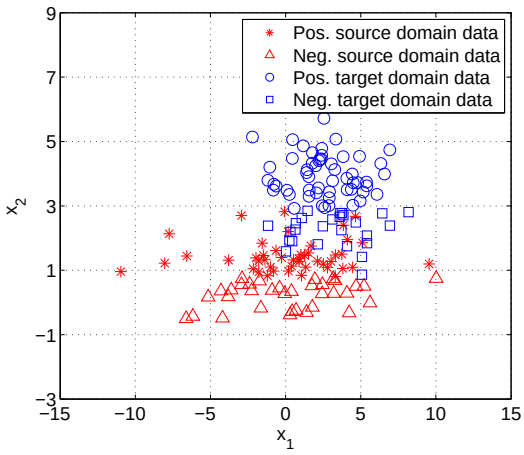
3.6.3 Experiments on Synthetic Data

As described in previous sections, in SSTCA, we aims to optimize three objectives simultaneously. The objectives include minimizing the distance between domains, maximizing the source domain label dependence in the latent space and preserving local geometric structure in the latent space. In this section, we perform experiments on synthetic data to demonstrate the effectiveness of these three objectives of SSTCA in learning a 1D latent space from the 2D data. For SSTCA, we use the linear kernel on both inputs and outputs, and fix $\mu = 1$, $\gamma = 0.5$. We will fully test the effectiveness of SSTCA next in two real-world applications in Chapter 4 and Chapter 5, respectively.

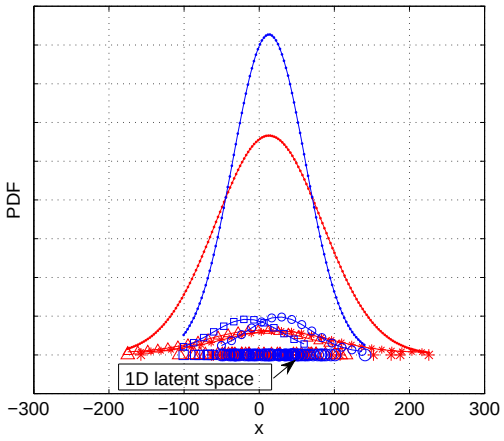
Label Information

In this experiment, we demonstrate the advantage of using label information in the source domain data to improve classification performance (Figure 3.6(a)). Since the focus is not on locality preserving, we set the λ in SSTCA to zero. Consequently, the difference between SSA and SSTCA is in the use of label information. As can be seen from Figure 3.6(b), the positive and negative samples overlap significantly in the latent space learned by TCA. On the other hand, with the use of label information, the positive and negative samples are more separated in

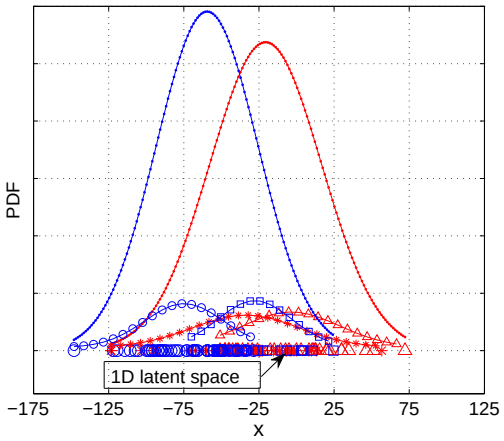
the latent space learned by SSTCA (Figure 3.6(c)), and thus classification also becomes easier.



(a) Data set 3 (acc: 69%).



(b) 1D projection by TCA (acc: 56%).

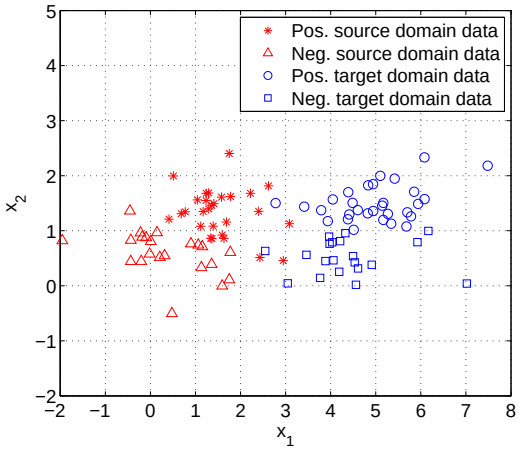


(c) 1D projection by SSTCA (acc: 79%).

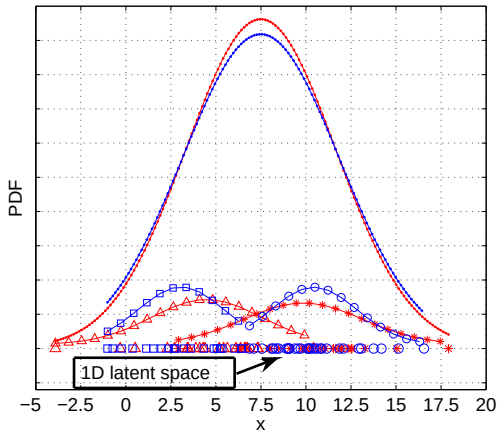
Figure 3.6: Illustrations of the proposed TCA and SSTCA on synthetic dataset 3. Accuracy of the 1-NN classifier in the original input space / latent space is shown inside brackets.

However, in some applications, it may be possible that the discriminative direction of the source domain data is quite different from that of the target domain data. An example is shown in Figure 3.7(a). In this case, encoding label information from the source domain (as SSTCA does) may not help or even hurt the classification performance as compared to the unsupervised TCA. As can be seen from Figures 3.7(b) and 3.7(c), positive and negative samples in the target domain are more separated in the latent space learned by TCA than in that learned by SSTCA.

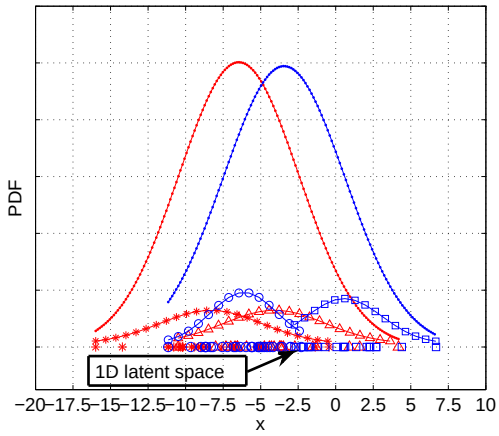
In summary, when the discriminative directions across different domains are similar, SSTCA can outperform TCA by encoding label information into the embedding learning. However, when the discriminative directions across different domains are different, SSTCA may not improve the performance or even performs worse than TCA. Nevertheless, compared to non-adaptive methods, both SSTCA and TCA can obtain better performance.



(a) Data set 4 (accuracy: 60%).



(b) 1D projection by TCA (acc: 90%).



(c) 1D projection by SSTCA (acc: 68%).

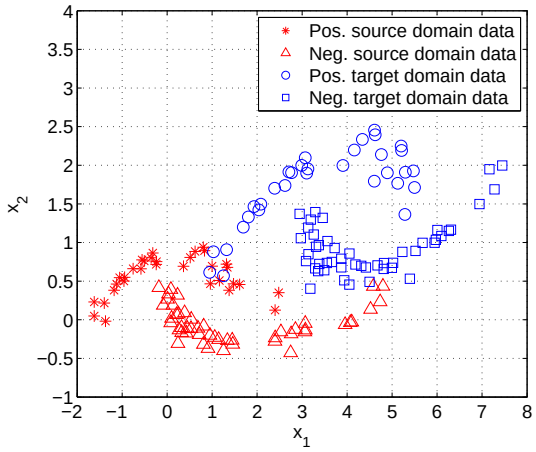
Figure 3.7: Illustrations of the proposed TCA and SSTCA on synthetic dataset 4. Accuracy of the 1-NN classifier in the original input space / latent space is shown inside brackets.

Manifold Information

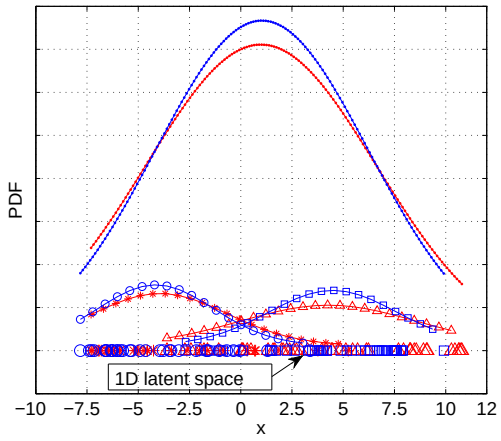
In this experiment, we demonstrate the advantage of using manifold information to improve classification performance. Both the source and domain data have the well-known two-moon manifold structure [232] (Figure 3.8(a)). SSTCA is used with and without Laplacian smoothing (by setting λ in (3.24) to 1000 and 0, respectively). As can be seen from Figures 3.8(b) and 3.8(c), Laplacian smoothing can indeed help improve classification performance when the manifold structure is available underlying the observed data.

3.6.4 Summary

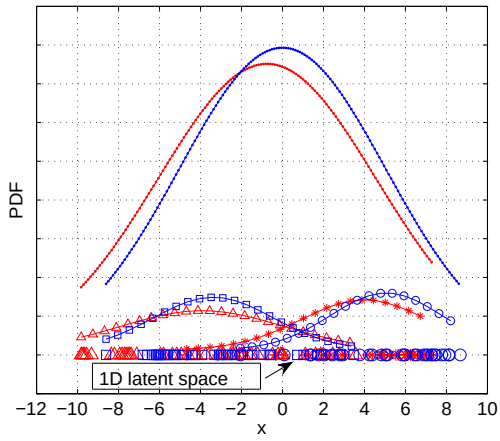
In SSTCA, we extend unsupervised TCA in the semi-supervised manner by maximizing the source domain label dependence in embedding learning. Similar to TCA, the time complexity



(a) Data set 5 (acc: 70%).



(b) 1D projection by SSTCA without Laplacian smoothing (acc: 83%).



(c) 1D projection by SSTCA with Laplacian smoothing (acc: 91%).

Figure 3.8: Illustrations of the proposed TCA and SSTCA on synthetic dataset 5. Accuracy of the 1-NN classifier in the original input space / latent space is shown inside brackets.

of SSTCA is $O(m(n_S+n_T)^2)$. Experiments on synthetic data show that when the discriminative directions of the source and target domains are the same or close to each other, then SSTCA can boost the classification accuracy compared to TCA. However, in some cases, when the discriminative directions of the source and target domains are different, then SSTCA does not work well. Furthermore, when the source and target domain data have a intrinsic manifold structure, SSTCA is more effective. More completed comparison between MMDE, TCA and SSTCA will conducted in Chapters 4-5.

3.7 Further Discussion

As mentioned in Chapter 3.2.4, embedding approaches, such as MVU, Colored MVU, MMDE and the proposed TCA and SSTCA, are all based on Hilbert space embedding of distributions

Table 3.1: Summary of dimensionality reduction methods based on Hilbert space embedding of distributions.

method	setting	out-of-sample	kernel	label	distribution matching	geometry	variance
MVU	unsupervised		nonparametric			$\surd$	$\surd$
Color MVU	supervised		nonparametric	$\surd$		$\surd$	
MMDE	unsupervised		nonparametric		$\surd$	$\surd$	$\surd$
TCA	unsupervised	$\surd$	parametric		$\surd$		$\surd$
SSTCA	semi-supervised	$\surd$	parametric	$\surd$	$\surd$	$\surd$	$\surd$

via MMD and HSIC. We summarize the relationships among these approaches in Table 3.1. Essentially, MMDE, TCA and SSTCA are dimensionality reduction approaches for domain adaptation, while MVU and colored MVU are dimensionality reduction approaches for single-domain data visualization. Note that SSTCA reduces to TCA when $\gamma = 0$ and $\lambda = 0$. If we further drop the *objective 1* described in Chapter 3.6.1, TCA reduces to MVU[5]. Furthermore, TCA is a generalized version of MMDE. Finally, SSTCA can also be reduced to the semi-supervised Color MVU when we drop the distribution matching term and set $\gamma = 1$ and $\lambda > 0$.

In summary, MVU and Colored MVU learn a nonparametric kernel matrix by maximizing data variance or label dependence for data analysis in a single domain, while MMDE learns a nonparametric kernel matrix by minimizing domain distance for cross-domain learning. They are all transductive and computationally expensive. To balance prediction performance with computational complexity in cross-domain learning, the proposed TCA and SSTCA learn parametric kernel matrices by simultaneously minimizing distribution distance and maximizing data variance and label dependence, which then reduce the time complexity from $O(m(n_S + n_T)^{6.5})$ to $O(m(n_S + n_T)^2)$. Note that criterion (3.7) in the kernel learning problem of MMDE is similar to the recently proposed supervised dimensionality reduction method Colored MVU[174], in which a low-rank approximation is used to reduce the number of constraints and variables in the SDP. However, gradient descent is required to refine the embedding space and thus the solution can still get stuck in a local minimum. Last but not the least, MMDE and TCA are unsupervised, and do not utilize side information. In contrast, SSTCA is semi-supervised, and exploits side information in embedding learning.

As mentioned in Chapter 2.3, besides the embedding approaches, instance re-weighting methods, such as KMM, KLIEP, uLSIF, etc., have also been proposed to solve the transductive transfer learning problems by matching data distributions. The main difference between these methods and our proposed method is that we aim to match data distributions between domains in a latent space, where data properties can be preserved, instead of matching them in the original feature space. This is beneficial as the real-world data are often noisy, while the latent space has

[5]Note that MVU is a transductive dimensionality reduction method, while TCA can be generalized to out-of-sample patterns even in this restricted setting.

been de-noised. As a result, in practice, matching data distributions in the de-noised latent space may be more useful than matching them in the noisy original feature space for target learning tasks in the transductive transfer learning setting.

CHAPTER 4

APPLICATIONS TO WIFI LOCALIZATION

In this section, we apply the proposed dimensionality reduction framework to an application in wireless sensor networks: indoor WiFi localization.

4.1 WiFi Localization and Cross-Domain WiFi Localization

Location estimation is a major task of pervasive computing and AI applications that range from context-aware computing [80, 106] to robotics [12]. While GPS is widely used, in many places outdoor where high buildings can block signals, and in most indoor places, GPS cannot work well. With the increasing availability of 802.11 WiFi networks in cities and buildings, locating and tracking a user or cargo with wireless signal strength or received signal strength (RSS) is becoming a reality [74, 100, 132, 65]. The objective of indoor WiFi localization is to estimate the location y_i of a mobile device based on the RSS values $x_i = (x_{i_1}, x_{i_2}, \ldots, x_{i_k})$ received from k Access Points (APs), which periodically send out wireless signals to others. In the following, we consider the two-dimensional coordinates of a location[1], and indoor WiFi localization is intrinsically a regression problem.

In recent years, machine-learning-based methods have been applied to the indoor WiFi localization [134, 133, 132, 65, 100]. In general, these methods work in two phases. In an *offline* phase, a mobile device (or multiple ones) moving around the wireless environment is used to collect wireless signals from APs. Then, the RSS values (e.g. the signal vector $x_i = (x_{i_1}, x_{i_2}, \ldots, x_{i_k})$) together with the physical location information (i.e. the location coordinates in the 2D floor where the mobile device is), are used as labeled training data to learn a computational or statistical model.

As an example, Figure 4.1 shows an indoor 802.11 wireless environment of size about $60m \times 50m$. Five APs (AP_1,...,AP_5) are set up in the environment. A user with an IBM T42 laptop that is equipped with an Intel Pro/2200BG internal wireless card walks through the environment from the location A to F at time t_A,...,t_F. Then, six signal vectors are collected, each of which is 5-dimensional, as shown in Table 4.1. Note that the blank cells in the table denote the missing values, which we can fill in a small default value, e.g., $-100dBm$. The corresponding labels of the signal vectors x_{t_A},...,x_{t_F} are the 2D coordinates of the locations A,...,F in the building.

[1]Extension to three-dimensional localization is straightforward.

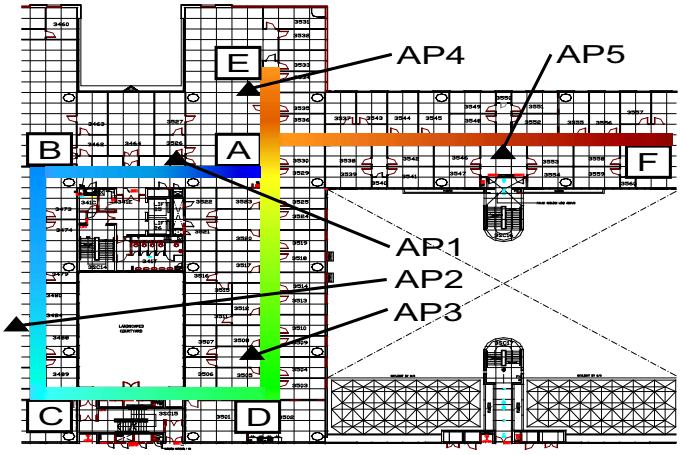


Figure 4.1: An indoor wireless environment example.

Table 4.1: An example of signal vectors (unit:dBm)

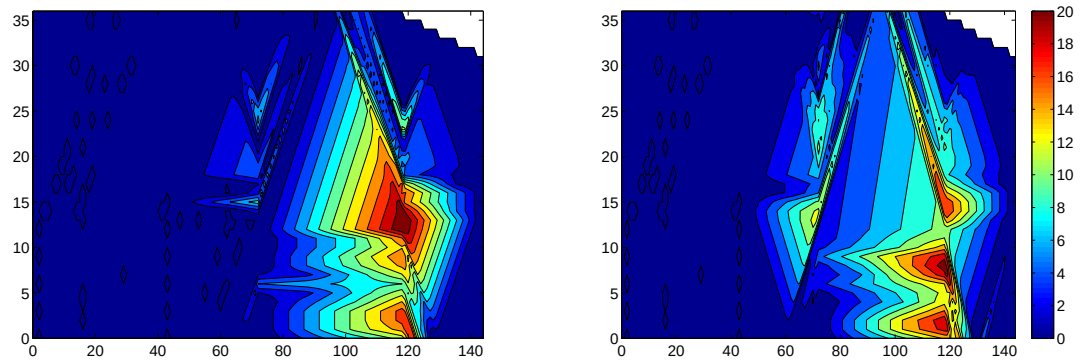
	AP_1	AP_2	AP_3	AP_4	AP_5
x_{t_A}	-40		-60	-40	-70
x_{t_B}	-50	-60		-80	
x_{t_C}		-40	-70		
x_{t_D}	-80		-40	-70	
x_{t_E}	-40		-70	-40	-80
x_{t_F}	-80			-80	-50

(All values are rounded for illustration)

However, most machine-learning methods rely on collecting a lot of labeled data to train an accurate localization model offline for use online, and assuming that the distributions of RSS data over different time periods are static. However, it is expensive to calibrate a localization model in a large environment. Moreover, the RSS values are noisy and can vary with time [217, 136]. As a result, even in the same environment, the RSS data collected in one time period may differ from those collected in another. An example is shown in Figure 4.2. As can be seen, the contours of the RSS values received from the same AP at different time periods are very different. Hence, domain adaptation is necessary for indoor WiFi localization.

4.2 Experimental Setup

We use a public data set from the 2007 IEEE ICDM Contest (the 2nd Task). This contains a few labeled WiFi data collected in time period T_1 (the source domain) and a large amount of unlabeled WiFi data collected in time period T_2 (the target domain). Here, "label" refers to the location information for which the WiFi data are received. WiFi data collected from different time periods are considered as different domains. The task is to predict the labels of the WiFi



(a) WiFi RSS received in T_1 from two APs (unit:dBm). (b) WiFi RSS received in T_2 from two APs (unit:dBm).

Figure 4.2: Contours of RSS values over a 2-dimensional environment collected from the same AP but in different time periods. Different colors denote different signal strength values (unit:dBm). Note that the original signal strength values are non-positive (the larger the stronger). Here, we shift them to positive values for visualization.

data collected in time period T_2. For more details on the data set, readers may refer to the contest report article [212].

Denote the data collected in time period T_1 and time period T_2 by $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively. In the experiments, we have $|\mathcal{D}_S| = 621$ and $|\mathcal{D}_T| = 3,128$. Furthermore, we randomly split $\mathcal{D}_T$ into $\mathcal{D}_T^u$ (the label information is removed in training) and $\mathcal{D}_T^o$. All the source domain data (621 instances in total) are used for training. As for the target domain data, 2,328 patterns are sampled to form D_T^o, and a variable number of patterns are sampled from the remaining 800 patterns to form D_T^u.

In the transductive evaluation setting, our goal is to learn a model from $\mathcal{D}_S$ and $\mathcal{D}_T^u$, and then evaluate the model on $\mathcal{D}_T^u$. In the out-of-sample evaluation setting, our goal is to learn a model from $\mathcal{D}_S$ and $\mathcal{D}_T^u$, and then evaluate the model on $\mathcal{D}_T^o$ (out-of-sample patterns). For each experiment, we repeat 10 times and then report the average performance using the Average Error Distance (AED):

$$AED = \frac{\sum_{(x_i,y_i)\in\mathcal{D}} |f(x_i) - y_i|}{N}.$$

Here, x_i is a vector of RSS values, $f(x_i)$ is the predicted location, y_i is the corresponding ground truth location, while $\mathcal{D} = \mathcal{D}_T^u$ in the transductive setting, and $\mathcal{D} = \mathcal{D}_T^o$ in the out-of-sample evaluation setting.

The following methods will be compared. For parameter tuning of TCA, SSTCA and all the other baseline methods, 50 labeled data are sampled from the source domain as validation set.

1. Traditional regression models that do not perform domain adaptation. These include the (supervised) regularized least square regression (RLSR), which is standard regression model. We train it on $\mathcal{D}_S$ only, and the (semi-supervised) Laplacian RLSR (LapRLSR) [14],

which is trained on both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ but without considering the difference in distributions. Note that the Laplacian RLSR has been applied to WiFi localization and is one of the state-of-the-art localization models [134].

2. A traditional dimensionality reduction method: Kernel PCA (KPCA) as introduced in Chapter 3.2.1. It first learns a projection from both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ via KPCA. RLSR is then applied on the projected $\mathcal{D}_S$ to learn a localization model.

3. Sample selection bias (or covariate shift) methods: KMM and KLIEP[2] as introduced in Chapter 2.3. They use both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ to learn weights of the patterns in $\mathcal{D}_S$, and then train a RLSR model on the weighted data. Following [82], we set the ϵ parameter in KMM as $B/\sqrt{n_1}$, where n_1 is the number of training data in the source domain. For KLIEP, we use the *likelihood cross-validation* method in [180] to automatically select the kernel width. Preliminary results suggest that the final performance of KLIEP can be sensitive to the initialization of the kernel width. Thus, its initial value is also tuned on the validation set.

4. A state-of-the-art domain adaptation method: SCL[3] as introduced in Chapter 2.3. It learns a set of new cross-domain features from both $\mathcal{D}_S$ and $\mathcal{D}_T^u$, and then augments features on the source domain data in $\mathcal{D}_S$ with the new features. A RLSR model is then trained.

5. The proposed TCA and SSTCA. First, we apply TCA / SSTCA on both $\mathcal{D}_S$ and $\mathcal{D}_T^u$ to learn transfer components, and map data in $\mathcal{D}_S$ to the latent space. Finally, a RLSR model is trained on the projected source domain data. There are two parameters in TCA, kernel width[4] σ and parameter μ. We first set $\mu = 1$ and search for the best σ value (based on the validation set) in the range $[10^{-5}, 10^5]$. Afterwards, we fix σ and search for the best μ value in $[10^{-3}, 10^3]$. For SSTCA, we use linear kernel for k_{yy} in (3.19) on the labels, and there are four tunable parameters (σ, μ, λ, and γ). We set σ and μ in the same manner as TCA. Then, we set $\gamma = 0.5$ and search for the best λ value in $[10^{-6}, 10^6]$. Afterwards, we fix λ and search for γ in $[0, 1]$. Note that this parameter tuning strategy may not get a global optimal combination of values of parameters. However, we find that the resulting performance is satisfactory in practice.

6. Methods that only perform distribution matching in a latent space: SSA[5] as introduced in 2.3 and TCAReduced, which replaces the constraint $W^\top KHKW = I$ in TCA by

[2]The code of KLIEP is downloaded from
`http://sugiyama-www.cs.titech.ac.jp/~sugi/software/KLIEP/index.html`

[3]Following [25], the pivot features are selected by mutual information while the number of pivots and other SCL parameters are determined by the validation data.

[4]We use the Laplace kernel $k(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|}{\sigma}\right)$, which has been shown to be a suitable kernel for the WiFi data [139].

[5]The code of SSA is provided by Paul von Bünau, the first author of [191].

$W^\top W = I$. Hence, TCAReduced aims to find a transformation W that minimizes the distance between different distributions without maximizing the variance in the latent space.

7. A closely related dimensionality reduction method: MMDE. This is a state-of-the-art method on the ICDM-07 contest data set [135, 139].

The first six methods will be compared in the out-of-sample setting in Chapters 4.3.1-4.3.3. Since MMDE (the last method) is transductive, we will compare the performance of TCA, SSTCA and MMDE in the transductive setting in Chapter 4.3.4.

4.3 Results

4.3.1 Comparison with Dimensionality Reduction Methods

We first compare TCA and SSTCA with some dimensionality reduction methods, including KPCA, SSA and TCAReduced in the out-of-sample setting. The number of unlabeled patterns in $\mathcal{D}_T^u$ is fixed at 400, while the dimensionality of the latent space varies from 5 to 50.

Figure 4.3 shows the results. As can be seen, TCA and SSTCA outperform all the other methods. Moreover, note that KPCA, though simple, can lead to significantly improved performance. This is because the WiFi data are highly noisy, and thus localization models learned in the de-noised latent space can be more accurate than those learned in the original input space. However, as mentioned in Chapter 3.3.2, KPCA can only de-noise but cannot ensure that the distance between data distributions in the two domains is reduced. Thus, TCA performs better than KPCA. In addition, though TCAReduced and SSA aim to reduce distance between domains, they may lose important information of the original data in the latent space, which in turn may hurt performance of the target learning tasks. Thus, they do not obtain good performance. Finally, we observe that SSTCA obtains better performance than TCA. As demonstrated in previous research [134], the manifold assumption holds on the WiFi data. Thus, the graph Laplacian term in SSTCA can effectively exploit label information from the labeled data to the unlabeled data across domains.

4.3.2 Comparison with Non-Adaptive Methods

In this experiment, we compare TCA and SSTCA with learning-based localization models that do not perform domain adaptation, including RLSR, LapRLSR and KPCA. The dimensionalities of the latent spaces for KPCA, TCA and SSTCA are fixed at 15. These values are determined based on the first experiment in Chapter 4.3.1.

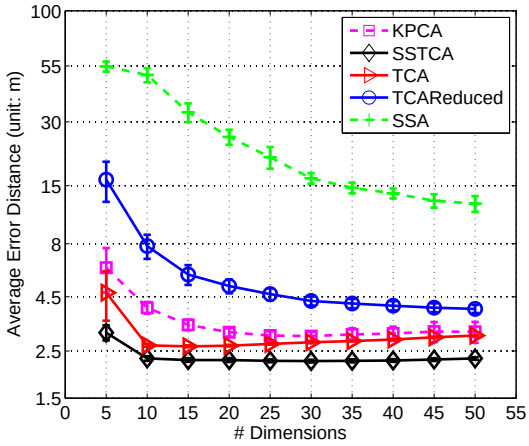


Figure 4.3: Comparison with dimensionality reduction methods.

Figure 4.4 shows the performance when the number of unlabeled patterns in $\mathcal{D}_T^u$ varies. As can be seen, even with only a few unlabeled data in the target domain, TCA and SSTCA can perform well for domain adaptation.

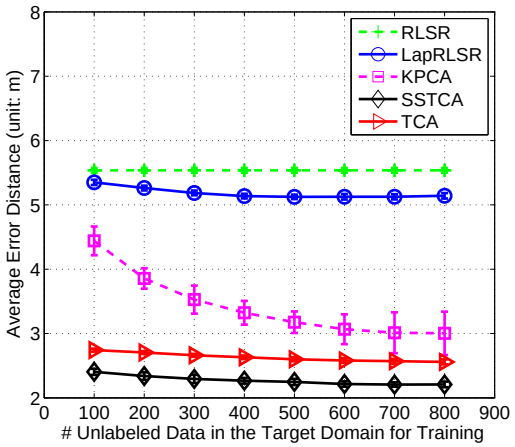


Figure 4.4: Comparison with localization methods that do not perform domain adaption.

4.3.3 Comparison with Domain Adaptation Methods

In this subchapter, we compare TCA and SSTCA with some state-of-the-art domain adaptation methods, including KMM, KLIEP, SCL and SSA. We fix the dimensionalities of the latent space in TCA and SSTCA at 15, while we fix the dimensionalities of the latent space in SSA and TCAReduced at 50. For training, all the source domain data are used and varying amount of the target domain data are sampled as $|D_T^u|$.

Results are shown in Figure 4.5. As can be seen, domain adaptation methods that are based on feature extraction (including SCL, TCA and SSTCA) perform much better than instance re-weighting methods (including KMM and KLIEP). This is again because the WiFi data are

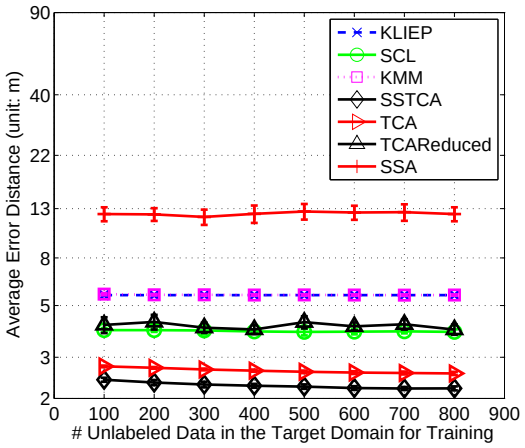
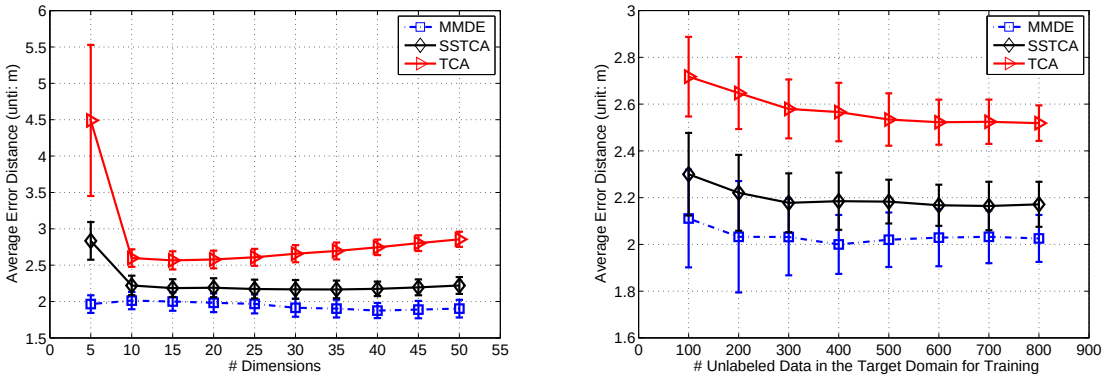


Figure 4.5: Comparison of TCA, SSTCA and the various baseline methods in the inductive setting on the WiFi data.

highly noisy, and so matching distributions directly based on the noisy observations may not be useful. Indeed, SCL may suffer from the bad choice of pivot features due to the noisy observations. On the other hand, TCA and SSTCA match distributions in the latent space, where the WiFi data have been implicitly de-noised.

4.3.4 Comparison with MMDE

In this subchapter, we compare TCA and SSTCA with MMDE in the transductive setting. The latent space is learned from $\mathcal{D}_S$ and a subset of the unlabeled target domain data sampled from $\mathcal{D}_T^u$. The performance is then measured on the same unlabeled data subset.



(a) Varying the dimensionality of the latent space.

(b) Varying the number of unlabeled data.

Figure 4.6: Comparison with MMDE in the transductive setting on the WiFi data.

Figure 4.6(a) shows the results for different dimensionalities of the latent space with $|\mathcal{D}_T^u| = 400$, while Figure 4.6(b) shows the results for different amounts of unlabeled target domain data, with the dimensionalities of MMDE, TCA and SSTCA fixed at 15. As can be seen, MMDE

outperforms TCA and SSTCA. This may be due to the limitation that the kernel matrix used in TCA / SSTCA is parametric. However, as mentioned in Chapter 3.4, MMDE is computationally expensive because it involves a SDP. This is confirmed in the training time comparison in Figure 4.7. In practice, TCA or SSTCA may be a better choice than MMDE for cross-domain adaptation.

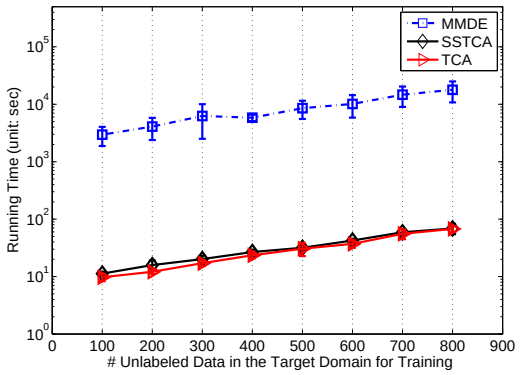


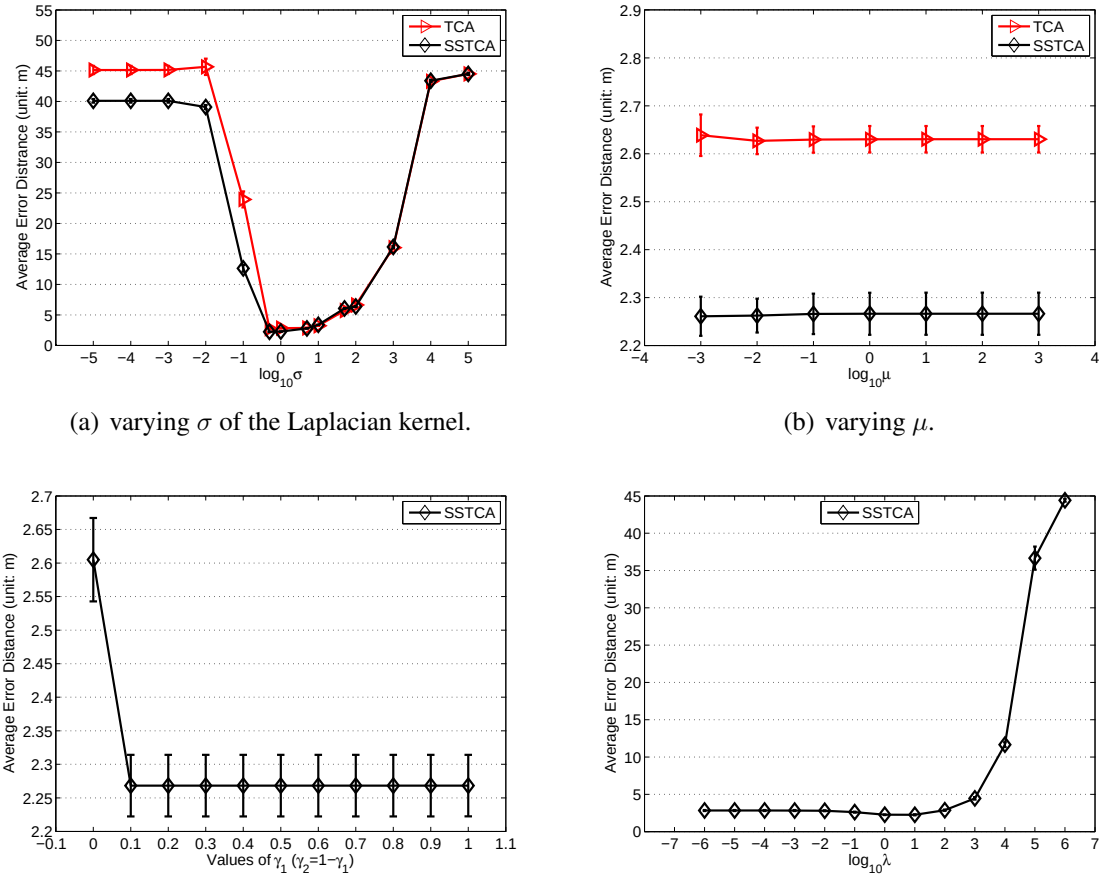
Figure 4.7: Training time with varying amount of unlabeled data for training.

4.3.5 Sensitivity to Model Parameters

In this subchapter, we investigate the effects of the parameters on the regression performance. These include the kernel width σ in the Laplacian kernel, tradeoff parameter μ, and for SSTCA, the two additional parameters γ and λ. The out-of-sample evaluation setting is used. All the source domain data are used, and we sample 2,328 samples from the target domain data to form $\mathcal{D}_T^o$, and another 400 samples to form $\mathcal{D}_T^u$. The dimensionalities of the latent spaces in TCA and SSTCA are fixed at 15. As can be seen from Figure 4.8, both TCA and SSTCA are insensitive to the settings of the various parameters.

4.4 Summary

In this section, we applied MMDE, TCA and SSTCA to the WiFi localization problem. Experimental results verified the effectiveness of our proposed methods in WiFi localization compared several cross-domain methods. From the results, we may observe that in the transductive experimental setting, MMDE performs best. The reason is that in TCA and SSTCA, we need tune the kernel parameters using cross-validation, while in MMDE, the kernel matrix is learned from the data automatically. As a result, the kernel matrix in MMDE can more fit the data, which is useful for learning tasks. However, MMDE may be not applicable in practice because of its expensive computational cost. Thus, for real-world applications, TCA and SSTCA are more desirable. Furthermore, based on the results in WiFi localization, SSTCA performs better than



(a) varying σ of the Laplacian kernel.

(b) varying μ.

(c) varying γ in SSTCA.

(d) varying λ in SSTCA.

Figure 4.8: Sensitivity analysis of the TCA / SSTCA parameters on the WiFi data.

TCA. The reason may be that the manifold assumption hold strongly on the WiFi data. As a result the manifold regularization term in SSTCA can be able to propagate label information from the source domain to the target domain effectively. Therefore, when the manifold assumption does hold on the source and target domain data, we suggest to use SSTCA for learning the latent space for transfer learning.

CHAPTER 5

APPLICATIONS TO TEXT CLASSIFICATION

In the previous section, we have showed the effectiveness of the proposed dimensionality reduction framework for the cross-domain WiFi localization problem. In this section, we apply the framework to another application: text classification.

5.1 Text Classification and Cross-domain Text Classification

Text classification, also known as document classification, aims to assign a document to one ore more categories based on its content. It is a fundamental task for Web mining and has been widely studied in the fields of machine learning, data mining and information retrieval [89, 215]. Traditional supervised learning approaches to text classification require sufficient labeled instances in a domain of interest in order to train a high-quality classifier. Test data are assumed to come from the same domain, following the same data distribution of the training domain data. Thus, most of these methods are unable to be applied to solve cross-domain text classification problems. However, in many real-world applications, labeled labeled are in short supply in the domain of interest. As a result, cross-domain learning algorithms are desirable in text classification.

In recent years, transfer learning techniques have been proposed to address the cross-domain text classification problems [48, 47, 49, 207, 204], which have been reviewed in Chapter 2. Most of these methods are designed for the text classification area only. For example, in [48] proposed a modified naive bayes algorithm for cross-domain text classification. The proposed method based on naive bayes, thus it is hard to be applied to other applications, such as WiFi localization. In contrast, we proposed dimensionality reduction framework can be used for general cross-domain regression or classification problems. Thus, it can be applied various diverse applications. In the following sections, we test the effectiveness of the proposed framework on a real-world cross-domain text classification dataset.

5.2 Experimental Setup

In this section, we perform cross-domain text classification experiments on a preprocessed data set of the 20-Newsgroups[1] [96]. This is a text collection of approximately 20,000 news-

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/

group documents hierarchically partitioned into 20 newsgroups. Documents from different sub-categories but under the same parent category are considered to be related domains.

In this experiment, we follow the preprocessing strategy in [47, 48] to create six data sets from this collection. For each data set, two top categories are chosen, one as positive and the other as negative. We then split the data based on sub-categories. Different sub-categories are considered as different domains, and the binary classification task is defined as top category classification. This splitting strategy ensures that the domains of labeled and unlabeled data are related, since they are under the same top categories. Besides, the domains are also ensured to be different, since they are drawn from different sub-categories. The six data sets created are "comp vs. sci", "rec vs. talk", "rec vs. sci", "sci vs. talk", "comp vs. rec" and "comp vs. talk" (Table 5.1). All the alphabets are converted to lower cases and stemmed using the Porter stemmer. Stop words are also removed and the document frequency feature is used to cut down the number of features.

Table 5.1: Summary of the six data sets constructed from the 20-Newsgroups data.

Task	# Fea.	# Doc.	Source Domain		Target Domain	
			# Pos.	# Neg.	# Pos.	# Neg.
Comp vs. Sci (**C** vs. **S**)	38,065	6,000	1,500	1,500	1,500	1,500
Rec vs. Talk (**R** vs. **T**)	30,165	6,000	1,500	1,500	1,500	1,500
Rec vs. Sci (**R** vs. **S**)	29,644	6,000	1,500	1,500	1,500	1,500
Sci vs. Talk (**S** vs. **T**)	33,151	6,000	1,500	1,500	1,500	1,500
Comp vs. Rec (**C** vs. **R**)	40,827	6,000	1,500	1,500	1,500	1,500
Comp vs. Talk (**C** vs. **T**)	45,514	6,000	1,500	1,500	1,500	1,500

From each of these six data sets, we randomly sample 40% of the documents from the source domain as $\mathcal{D}_S$, and sample 40% from the target domain to form the unlabeled subset $\mathcal{D}_T^u$, and the remaining 60% in the target domain to form the out-of-sample subset $\mathcal{D}_T^o$. Hence, in each cross-domain classification task, $|\mathcal{D}_S| = 1200$, $|\mathcal{D}_T^u| = 1200$ and $|\mathcal{D}_T^o| = 1800$.

We run 10 repetitions and report the average results. All experiments are performed in the out-of-sample setting. The evaluation criterion is the classification accuracy.

Note that we do not show the performance of MMDE in this experiment, because it results in "out of memory" on learning the kernel matrix using SDP solvers. Similar to Chapter 4, we perform a series of experiments to compare TCA and SSTCA with the following methods:

1. Linear support vector machine (SVM) in the original input space;
2. KPCA. A linear SVM is then trained in the latent space;
3. Three domain adaptation methods: KMM, KLIEP and SCL. Again, a linear SVM is used as the classifier in the latent space.

We experiment with the Laplace, RBF and linear kernels[2] for feature extraction or re-weighting in KPCA, KMM, TCA and SSTCA. Note that we do not compare with SSA because it results in "out of memory" on computing the covariance matrices.

For SSTCA, kernel k_{yy} in (3.19) is the linear kernel. The μ parameters in TCA and SSTCA are set to 1, and the λ parameter in SSTCA is set to 0.0001.

5.3 Results

5.3.1 Comparison to Other Methods

Results are shown in Tables 5.2 and 5.3. As can be seen, we can obtain a similar conclusion as in Chapter 4. Overall, feature extraction methods outperform instance-reweighting methods. In addition, on tasks such as "**R** vs. **T**", "**C** vs. **T**", "**C** vs. **S**" and "**C** vs. **R**", the performance of PCA is comparable to that of linear TCA. However, on tasks such as "**R** vs. **S**" and "**S** vs. **T**", linear TCA performs much better than PCA. This agrees with our motivation and the previous conclusion on the WiFi experiments, namely that mapping data from different domains to a latent space spanned by the principal components may not work well as PCA cannot guarantee a reduction in distance of the two domain distributions. In general, one may notice two main differences between the results on the WiFi data and those on the text data. First, the linear kernel performs better than the RBF and Laplacian kernels here. This agrees with the well-known observation that the linear kernel is often adequate for high-dimensional text data. Moreover, TCA performs better than SSTCA on the text data. This may be because the manifold assumption is weaker in the text domain than in the WiFi domain.

5.3.2 Sensitivity to Model Parameters

Figure 5.1 shows how the various parameters in TCA and SSTCA affect the classification performance. Here, we use linear kernels for both the inputs and outputs, and the dimensionalities of the latent spaces are fixed at 10. Thus, the remaining free parameters are μ for TCA; and μ, γ and λ for SSTCA. First, Figure 5.1(a) shows the sensitivity w.r.t. μ for TCA, and Figure 5.1(b) shows that for SSTCA (with $\gamma = 0.5$ and $\lambda = 10^{-4}$). As can be seen, both TCA and SSTCA are insensitive to the setting of μ. Next, Figure 5.1(c) shows the sensitivity of SSTCA w.r.t. γ (with $\mu = 1$ and $\lambda = 10^{-4}$). Again, there is a wide range of γ for which the performance of SSTCA is quite stable. Finally, Figure 5.1(d) shows the sensitivity w.r.t. λ (with $\gamma = 0.5$ and $\mu = 1$). Different from the WiFi results in Figure 4.8(d) where SSTCA performs well when $\lambda \leq 10^2$, here it performs well only when λ is very small ($\lambda \leq 10^{-4}$). This indicates that mani-

[2]RBF and linear kernels are defined as $k(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma}\right)$ and $k(x_i, x_j) = \|x_i - x_j\|^2$

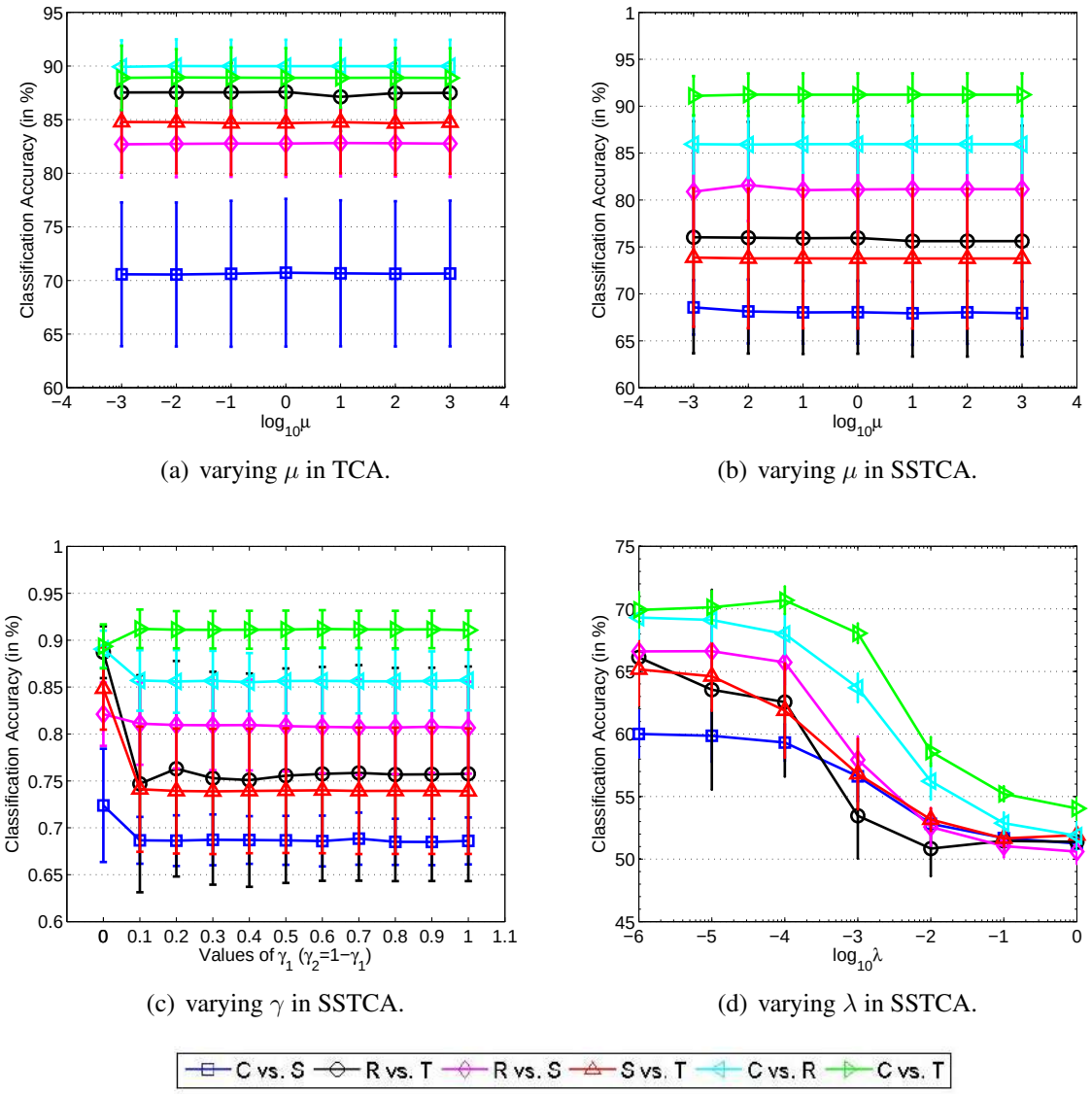
Table 5.2: Classification accuracies (%) of the various methods (the number inside parentheses is the standard deviation).

method		#dim	Task		
			S vs. **T**	**C** vs. **R**	**C** vs. **T**
SVM		all	76.70 (1.05)	81.59 (1.36)	90.51 (0.70)
TCA	linear kernel	5	70.31 (6.31)	77.19 (12.70)	83.57 (2.72)
		10	**84.51 (5.04)**	89.79 (2.54)	88.67 (3.01)
		20	82.23 (2.64)	89.73 (4.04)	92.03 (2.05)
		30	79.81 (4.20)	**91.81 (2.38)**	92.23 (2.41)
	Laplacian kernel	5	58.95 (1.14)	81.95 (0.82)	87.41 (1.41)
		10	69.02 (5.31)	82.29 (2.57)	90.01 (1.02)
		20	74.71 (3.01)	85.59 (1.59)	92.68 (1.12)
		30	74.40 (2.49)	87.89 (2.22)	92.63 (0.84)
	RBF kernel	5	63.31 (2.13)	78.14 (2.11)	86.05 (0.55)
		10	81.65 (4.08)	82.69 (2.24)	89.15 (0.69)
		20	79.54 (1.89)	83.51 (3.32)	90.77 (0.83)
		30	79.50 (1.91)	83.71 (2.27)	91.58 (0.64)
SSTCA	linear kernel	5	69.77 (5.14)	83.68 (2.64)	88.02 (3.06)
		10	73.75 (7.55)	85.99 (3.22)	91.38 (2.22)
		20	78.19 (4.17)	86.71 (3.36)	91.81 (2.13)
		30	72.79 (5.75)	85.81 (3.23)	**93.38 (2.02)**
	Laplacian kernel	5	71.46 (1.87)	79.09 (3.41)	89.31 (1.37)
		10	73.10 (2.10)	85.26 (2.25)	91.72 (0.64)
		20	74.94 (2.28)	84.28 (1.27)	92.47 (0.74)
		30	74.67 (1.79)	85.30 (1.80)	92.73 (0.76)
	RBF kernel	5	69.23 (1.79)	74.27 (.0287)	86.25 (1.32)
		10	77.61 (1.49)	83.09 (.0287)	90.35 (1.18)
		20	79.46 (1.27)	80.02 (.0287)	90.62 (0.83)
		30	79.88 (1.52)	81.30 (.0287)	90.21 (0.96)
KPCA	linear kernel	5	67.04 (9.62)	60.11 (12.98)	85.34 (2.78)
		10	81.42 (6.67)	87.33 (3.56)	91.24 (1.84)
		20	80.22 (3.81)	89.49 (3.34)	93.44 (1.92)
		30	77.92 (4.32)	**91.36 (1.51)**	**93.66 (1.81)**
	Laplacian kernel	5	58.90 (.0252)	67.12 (4.93)	68.77 (1.78)
		10	80.37 (.0252)	58.87 (4.97)	58.47 (2.35)
		20	72.67 (.0252)	75.71 (6.83)	73.94 (3.75)
		30	69.36 (.0252)	75.07 (10.64)	74.18 (4.24)
	RBF kernel	5	61.89 (.0252)	57.46 (3.26)	56.24 (1.06)
		10	79.92 (.0252)	57.84 (3.74)	56.82 (2.03)
		20	79.37 (.0252)	67.73 (5.53)	62.36 (3.84)
		30	72.31 (.0252)	67.66 (4.48)	64.76 (5.14)
SCL		all+50	76.73 (1.00)	81.60 (1.35)	90.61 (0.64)
KMM	linear kernel	all	76.38 (1.32)	78.17 (1.29)	88.06 (1.33)
	Laplacian kernel	all	75.83 (1.27)	77.81 (1.21)	85.92 (0.70)
	RBF kernel	all	76.43 (1.17)	77.28 (1.18)	84.30 (1.00)
KLIEP		all	75.11 (0.81)	77.94 (1.09)	85.12 (0.99)

Table 5.3: Classification accuracies (%) of the various methods (the number inside parentheses is the standard deviation).

method		#dim	Task		
			C vs. **S**	**R** vs. **T**	**R** vs. **S**
SVM		all	68.26 (1.23)	72.33 (2.32)	75.86 (1.55)
TCA	linear kernel	5	63.32 (8.03)	72.99 (20.73)	76.23 (5.29)
		10	70.41 (6.84)	**87.67 (2.12)**	82.83 (3.07)
		20	69.04 (5.07)	81.52 (8.86)	83.86 (3.24)
		30	69.01 (2.39)	77.26 (15.81)	**84.44 (2.29)**
	Laplacian kernel	5	56.47 (6.89)	71.01 (8.43)	77.79 (2.41)
		10	69.12 (13.27)	78.68 (8.23)	79.58 (8.12)
		20	69.37 (11.22)	79.57 (4.31)	78.71 (9.22)
		30	68.58 (11.21)	80.43 (4.64)	76.69 (9.52)
	RBF kernel	5	**74.44 (5.53)**	76.02 (8.99)	78.21 (3.43)
		10	74.88 (3.51)	82.51 (7.65)	78.34 (6.58)
		20	72.60 (5.60)	77.47 (2.62)	78.09 (6.88)
		30	71.64 (5.49)	77.62 (3.75)	80.11 (7.73)
SSTCA	linear kernel	5	65.07 (5.00)	73.44 (15.55)	76.76 (4.72)
		10	68.64 (3.00)	75.11 (11.93)	81.46 (3.59)
		20	64.28 (3.48)	60.69 (14.87)	77.45 (5.30)
		30	65.08 (3.12)	66.30 (16.74)	77.98 (4.19)
	Laplacian kernel	5	72.46 (2.80)	76.57 (3.67)	77.53 (1.59)
		10	**75.29 (3.92)**	79.84 (5.03)	72.70 (10.69)
		20	71.99 (4.73)	81.77 (3.44)	72.99 (9.99)
		30	69.71 (4.99)	82.09 (4.42)	72.34 (10.82)
	RBF kernel	5	75.73 (1.35)	84.61 (3.90)	70.36 (7.15)
		10	73.76 (3.25)	74.50 (7.85)	78.51 (7.50)
		20	70.87 (7.51)	75.49 (6.67)	79.28 (7.20)
		30	70.16 (5.98)	77.03 (5.56)	79.06 (7.60)
KPCA	linear kernel	5	63.63 (10.45)	70.34 (21.52)	67.08 (6.67)
		10	68.66 (6.59)	**88.26 (5.85)**	68.59 (10.00)
		20	69.18 (6.27)	82.59 (7.07)	71.46 (7.41)
		30	70.55 (2.81)	80.94 (11.63)	78.90 (8.33)
	Laplacian kernel	5	48.30 (10.07)	64.28 (4.89)	62.00 (7.68)
		10	44.43 (8.01)	81.52 (9.00)	54.42 (7.33)
		20	49.08 (10.46)	55.67 (6.35)	50.42 (1.01)
		30	45.24 (8.17)	63.13 (7.76)	50.43 (1.03)
	RBF kernel	5	53.87 (10.38)	67.24 (4.51)	62.23 (6.22)
		10	53.82 (6.23)	78.50 (4.23)	51.64 (2.11)
		20	47.66 (8.19)	60.94 (10.97)	50.49 (1.00)
		30	47.82 (8.37)	69.13 (9.66)	51.86 (3.82)
SCL		all+50	68.29 (1.22)	72.38 (2.36)	75.87 (1.48)
KMM	linear kernel	all	69.81 (1.27)	72.86 (1.53)	75.29 (1.85)
	Laplacian kernel	all	69.64 (1.27)	73.10 (1.67)	76.62 (1.23)
	RBF kernel	all	69.65 (1.24)	73.07 (1.48)	76.63 (1.14)
KLIEP		all	68.55 (1.36)	72.23 (1.20)	75.53 (1.17)

fold regularization may not be useful on the text data. In this case, using *unsupervised* TCA for domain adaptation may be a better choice than using SSTCA.



(a) varying μ in TCA.

(b) varying μ in SSTCA.

(c) varying γ in SSTCA.

(d) varying λ in SSTCA.

Figure 5.1: Sensitivity analysis of the TCA / SSTCA parameters on the text data.

5.4 Summary

In this section, we applied TCA and SSTCA to the text classification problem. Experimental results verified that our proposed methods based on the dimensionality reduction framework can achieve promising results in cross-domain text classification. However, different from the conclusion in WiFi localization, TCA outperforms SSTCA slightly in the text classification problem. The reason may be that the manifold assumption may not hold on the text data. As a result the manifold regularization term in SSTCA may not be able to propagate label information from the source domain to the target domain. In this case, SSTCA is not supposed

to outperform TCA. In a worse case, SSTCA may even perform worse than TCA if the manifold regularization hurt the learning performance. Therefore, when the manifold assumption does not hold on the source and target domain data, we suggest to use TCA for learning the latent space for transfer learning.

CHAPTER 6

DOMAIN-DRIVEN FEATURE SPACE TRANSFER FOR SENTIMENT CLASSIFICATION

In the previous chapters, we have considered transfer learning when the source and target domain have implicit overlap, which means the domain knowledge across these domains is hidden. Our strategy is to exploit a general dimensionality reduction framework to discover common latent factors that are useful for knowledge transfer across domains. We have proposed three methods based on the framework to learn the transfer latent space and applied them to two diverse real-world applications: indoor WiFi localization and text classification successfully. However, in some domain areas, we have some explicit domain knowledge that can help to design a more effective latent space for specific applications. In this chapter, we highlight one such application in sentiment classification, which we explain next, to show that domain knowledge can be encoded in feature space learning for transfer learning when it is available.

6.1 Sentiment Classification

With the explosion of Web 2.0 services, more and more user-generated resources have been available on the World Wide Web. They exist in the form of online user reviews on shopping or opinion sites, in posts of personal blogs or users feedback on news. This rich opinion information on the Web can help an individual make a decision to buy a product or plan a traveling route, or help a company to do a survey on its products or services, or even help the government to get the feedback on a policy. However, the sentiment data are in large-scale, and in many cases, opinions are hidden in long forum posts and blogs. As a result, how to find, extract or summarize useful information from opinion sources on the Web is an important and challenging task. Recently, *opinion mining*, also known as *sentiment analysis*, has attracted much attention in the natural language processing and information retrieval areas [110, 145, 111]. In sentiment analysis, there are some interesting tasks, for example, opinion extraction and summarization [81, 118, 184, 93], opinion integration [117], review spam identification [88], review quality prediction [116, 114] and opinion analysis in multiple languages [1], etc. Among these tasks, *Sentiment classification*, also known as *subjective classification* [145, 111], which aims at classifying text segment, e.g., text sentences and review articles, etc, into polarity categories (e.g., positive or negative), is an important task and has been widely studied because many users

do not explicitly indicate their sentiment polarity thus we need to predict it from the text data generated by users.

In sentiment classification, a domain D denotes a class of objects, events and their properties in the world. For example, different types of products, such as books, dvds and furniture, can be regarded as different domains. Sentiment data are the text segment containing user opinions about objects, events and their properties of the domain. User sentiment may exist in the form of a sentence, paragraph or article, which is denoted by x_j. In either case, it corresponds with a sequence of words $w_1 w_2 ... w_{x_j}$, where w_i is a word from a vocabulary W. Here, we represent user sentiment data with a bag-of-words method, with $c(w_i, x_j)$ to denote the frequency of word w_i in x_j. Without loss of generality, we use a unified vocabulary W for all domains and $|W| = m$. Furthermore, in sentiment classification tasks, either single word or NGram can be used as features to represent sentiment data, thus in the rest of this chapter, we will use *word* and *feature* interchangeably.

For each sentiment data x_j, there is a corresponding label y_j. $y_j = +1$ if the overall sentiment expressed in x_j is positive. $y_j = -1$ if the overall sentiment expressed in x_j is negative. A pair of sentiment text and its corresponding sentiment polarity $\{x_j, y_j\}$ is called the *labeled sentiment data*. If x_j has no polarity assigned, it is *unlabeled sentiment data*. Besides positive and negative sentiment, there are also neutral and mixed sentiment data in practical applications. *Mixed polarity* means user sentiment is positive in some aspects but negative in other ones. *Neutral polarity* means that there is no sentiment expressed by users. In this chapter, we only focus on positive and negative sentiment data, but it is not hard to extend the proposed solution to address multi-category sentiment classification problems.

6.2 Existing Works in Cross-Domain Sentiment Classification

In recent years, various machine learning techniques have been proposed for sentiment classification [145, 111]. In literature, compared to unsupervised learning methods [187], supervised learning algorithms [146] have been proven promising and widely used in sentiment classification. To date, there exist a lot of research work being proposed to improve the classification performance in the supervised setting [144, 128]. However, these methods rely heavily on manually labeled training data to train an accurate sentiment classifier. In order to reduce the human-annotating cost, Goldberg and Zhu adapted a graph-based semi-supervised learning method to make use of unlabeled data for sentiment classification. Sindhwani *et al.* [171] and Li *et al.* [105] proposed to incorporate lexical knowledge to the graph-based semi-supervised learning and non-negative matrix tri-factorization approaches to sentiment classification with a few labeled data.

However, these semi-supervised learning methods still require a few labeled data in the target domain to train an accurate sentiment classifier. In practice, we may have hundreds or thousands of domains at hand, it would be nice to annotate only several domain data to sentiment classifiers, which can be used to make predictions accurately in all other domains. Furthermore, similar to supervised learning methods, these approaches are domain dependent caused by changes in vocabulary. The reason is that users may use domain-specific words to express their sentiment in different domains. Table 6.1 shows several user review sentences from two domains: *electronics* and *video games*. In the *electronics* domain, we may use words like "compact", "sharp" to express our positive sentiment and use "blurry" to express our negative sentiment. While in the *video game* domain, words like "hooked", "realistic" indicate positive opinion and the word "boring" indicates negative opinion. Due to the mismatch between domain-specific words, a sentiment classifier trained in one domain may not work well when directly applied to other domains. Thus cross-domain sentiment classification algorithms are highly desirable to reduce domain dependency and manually labeling cost.

Table 6.1: Cross-domain sentiment classification examples: reviews of *electronics* and *video games* products. Boldfaces are domain-specific words, which are much more frequent in one domain than in the other one. Italic words are some domain-independent words, which occur frequently in both domains. "**+**" denotes positive sentiment, and "**-**" denotes negative sentiment.

	electronics	*video games*
+	**Compact**; easy to operate; very *good* picture quality; looks **sharp**!	A very *good* game! It is action packed and full of *excitement*. I am very much **hooked** on this game.
+	I purchased this unit from Circuit City and I was very *excited* about the quality of the picture. It is really *nice* and **sharp**.	Very **realistic** shooting action and *good* plots. We played this and were **hooked**.
-	It is also quite **blurry** in very dark settings. I will *never buy* HP again.	The game is so **boring**. I am extremely unhappy and will probably *never buy* UbiSoft again.

As a result, a sentiment classifier trained in one domain cannot be applied to another domain directly. To address this problem, Blitzer *et al.* [25] proposed the structural correspondence learning (SCL) algorithm, which has been introduced in Chapter 2.3, to exploit domain adaptation techniques for sentiment classification, which is a state-of-the-art method in cross-domain sentiment classification. SCL is motivated by a multi-task learning algorithm, alternating structural optimization (ASO), proposed by Ando and Zhang [4]. SCL tries to construct a set of related tasks to model the relationship between "pivot features" and "non-pivot features". Then "non-pivot features" with similar weights among tasks tend to be close with each other in a low-dimensional latent space. However, in practice, it is hard to construct a reasonable number of related tasks from data, which may limit the transfer ability of SCL for cross-domain sentiment classification. More recently, Li *et al.* [104] proposed to transfer common lexical knowledge across domains via matrix factorization techniques.

In this chapter, we target at finding an effective approach for the cross-domain sentiment classification problem. In particular, we propose a *spectral feature alignment* (SFA) algorithm to find a new representation for cross-domain sentiment data, such that the gap between domains can be reduced. SFA uses some *domain-independent words* as a bridge to construct a bipartite graph to model the co-occurrence relationship between *domain-specific words* and *domain-independent words*. The idea is that if two domain-specific words have connections to more common domain-independent words in the graph, they tend to be aligned together with higher probability. Similarly, if two domain-independent words have connections to more common domain-specific words in the graph, they tend to be aligned together with higher probability. We adapt a spectral clustering algorithm, which is based on the graph spectral theory [41], on the bipartite graph to co-align domain-specific and domain-independent words into a set of feature-clusters. In this way, the clusters can be used to reduce the mismatch between domain-specific words of both domains. Finally, we represent all data examples with these clusters and train sentiment classifiers based on the new representation.

6.3 Problem Statement and A Motivating Example

Problem Definition Assume we are given two specific domains D_S and D_T, where D_S and D_T are referred to as a source domain and a target domain respectively, suppose we have a set of labeled sentiment data $\mathcal{D}_S = \{(x_{S_i}, y_{S_i})\}_{i=1}^{n_S}$ in D_S, and some unlabeled sentiment data $\mathcal{D}_T = \{x_{T_j}\}_{j=1}^{n_T}$ in D_T. The task of cross-domain sentiment classification is to learn an accurate classifier to predict the polarity of unseen sentiment data from D_T.

In this section, we use an example to introduce the motivation of our solution to the cross-domain sentiment classification problem. First of all, we assume the sentiment classifier f is a linear function, which can be written as

$$y^* = f(x) = \mathbf{sgn}(xw^T),$$

where $x \in \mathbb{R}^{1\times m}$ and $\mathbf{sgn}(xw^T) = +1$ if $xw^T \geq 0$, otherwise, $\mathbf{sgn}(xw^T) = -1$. w is the weight vector of the classifier, which can be learned from a set of training data (pairs of sentiment data and their corresponding polarity labels).

Consider the example shown in Table 6.1 to illustrate our idea. We use a standard bag-of-words method to represent sentiment data of the *electronics* (**E**) and *video games* (**V**) domains. From Table 6.2, we can see that the difference between domains is caused by the frequency of the domain-specific words. Domain-specific words in the **E** domain, such as *compact*, *sharp*, *blurry*, do not occur in the **V** domain. On the other hand, domain-specific words in the **V** domain, such as *hooked*, *realistic*, *boring*, do not occur in the **E** domain. Suppose the **E** domain is the source domain and the **V** domain is the target domain, our goal is to train a vector of

weights w^* with labeled data from the **E** domain, and use it to predict sentiment polarity for the **V** domain data.[1] Based on the three training sentences in the **E** domain, the weights of features such as *compact* and *sharp* should be positive. The weight of features such as *blurry* should be negative and the weights of features such as *hooked*, *realistic* and *boring* can be arbitrary or zeros if an L_1 regularizer is applied on w for model training. However, an ideal weight vector in the **V** domain should have positive weights for features such as *hooked*, *realistic* and a negative weight for the feature *boring*, while the weights of features such as *compact*, *sharp* and *blurry* may take arbitrary values. That is why the classifier learned from the **E** domain may not work well in the **V** domain.

Table 6.2: Bag-of-words representations of *electronics* (**E**) and *video games* (**V**) reviews. Only domain-specific features are considered. "..." denotes all other words.

		...	compact	sharp	blurry	hooked	realistic	boring
E	**+**	...	1	1	0	0	0	0
	+	...	0	1	0	0	0	0
	-	...	0	0	1	0	0	0
V	**+**	...	0	0	0	1	0	0
	+	...	0	0	0	1	1	0
	-	...	0	0	0	0	0	1

In order to reduce the mismatch between features of the source and target domains, a straightforward solution is to make them more similar by adopting a new representation. Table 6.3 shows an ideal representation of domain-specific features. Here, *sharp_hooked* denotes a cluster consisting of *sharp* and *hooked*, *compact_realistic* denotes a cluster consisting of *compact* and *realistic*, and *blurry_boring* denotes a cluster consisting of *blurry* and *boring*. We can use these clusters as high-level features to represent domain-specific words. Based on the new representation, the weight vector w^* trained in the **E** domain should be also an ideal weight vector in the **V** domain. In this way, based on the new representation, the classifier learned from one domain can be easily adapted to another one.

Table 6.3: Ideal representations of domain-specific words.

		...	sharp_hooked	compact_realistic	blurry_boring
E	**+**	...	1	1	0
	+	...	1	0	0
	-	...	0	0	1
V	**+**	...	1	0	0
	+	...	1	1	0
	-	...	0	0	1

The problem is how to construct such an ideal representation as shown in Table 6.3. Clearly, if we directly apply traditional clustering algorithms such as k-means [76] on Table 6.2, we

[1]For simplicity, we only discuss domain-specific words here and ignore all other words.

are not able to align *sharp* and *hooked* into one cluster, since the *distance* between them is large. In order to reduce the gap and align domain-specific words from different domains, we can utilize domain-independent words as a bridge. As shown in Table 6.1, words such as *sharp*, *hooked compact* and *realistic* often co-occur with other words such as *good* and *exciting*, while words such as *blurry* and *boring* often co-occur with a word *never_buy*. Since the words like *good*, *exciting* and *never_buy* occur frequently in both the **E** and **V** domains, they can be treated as domain-independent features. Table 6.4 shows co-occurrences between domain-independent and domain-specific words. It is easy to find that, by applying clustering algorithms such as k-means on Table 6.4, we can get the feature clusters shown in Table 6.3: *sharp_hooked*, *blurry_boring* and *compact_realistic*.

Table 6.4: A co-occurrence matrix of domain-specific and domain-independent words.

	compact	realistic	sharp	hooked	blurry	boring
good	1	1	1	1	0	0
exciting	0	0	1	1	0	0
never_buy	0	0	0	0	1	1

The example described above motivates us that the co-occurrence relationship between domain-specific and domain-independent features is useful for feature alignment across different domains. We proposed to use a bipartite graph to represent this relationship and then adapt spectral clustering techniques to find a new representation for domain-specific features. In the following section, we will present spectral domain-specific feature alignment algorithm in detail.

6.4 Spectral Domain-Specific Feature Alignment

In this section, we describe our algorithm for adapting spectral clustering techniques to align domain-specific features from different domains for cross-domain sentiment classification.

As mentioned above, our proposed method consists of two steps: (1) to identify domain-independent features and (2) to align domain-specific features. In the first step, we aim to learn a feature selection function $\phi_{DI}(\cdot)$ to select l domain-independent features, which occur frequently and act similarly across domains D_S and D_T. These domain-independent features are used as a bridge to make knowledge transfer across domains possible. After identifying domain-independent features, we can use $\phi_{DS}(\cdot)$ to denote a feature selection function for selecting domain-specific features, which can be defined as the complement of domain-independent features. In the second step, we aims to to learn an alignment function $\varphi : \mathbb{R}^{(m-l)} \rightarrow \mathbb{R}^k$ to align domain-specific features from both domains into k predefined feature clusters $z_1, z_2, ..., z_k$, s.t. the difference between domain specific features from different domains on the new representation constructed by the learned clusters can be dramatically reduced.

For simplicity, we use W_{DI} and W_{DS} to denote the vocabulary of domain-independent and domain-specific features respectively. Then sentiment data x_i can be divided into two disjoint views. One view consists of features in W_{DI}, and the other is composed of features in W_{DS}. We use $\phi_{DI}(x_i)$ and $\phi_{DS}(x_i)$ to denote the two views respectively.

6.4.1 Domain-Independent Feature Selection

First of all, we need to identify which features are domain independent. As mentioned above, domain-independent features should occur frequently and act similarly in both the source and target domains. In this section, we present several strategies for selecting domain-independent features.

A first strategy is to select domain-independent features based on their frequency in both domains. More specifically, given the number l of domain-independent features to be selected, we choose features that occur more than k times in both the source and target domains. k is set to be the largest number such that we can get at least l such features.

A second strategy is based on the mutual dependence between features and labels on the source domain data. In [25], mutual information is applied on source domain labeled data to select features as "pivots", which can be referred to as domain-independent features in this papers. In information theory, mutual information is used to measure the mutual dependence between two random variables. Feature selection using mutual information, which is shown in (6.1) as follows, can help identify features relevant to source domain labels. But there is no guarantee that the selected features act similarly in both domains.

$$I(X^i; y) = \sum_{y \in \{+1,-1\}} \sum_{x \in X^i} p(x, y) log_2 \left(\frac{p(x, y)}{p(x)p(y)} \right), \tag{6.1}$$

where we denote X^i and y a feature and class label, respectively.

Here we propose a third strategy for selecting domain-independent features. Motivated by the supervised feature selection criteria, we can use mutual information to measure the dependence between features and domains. If a feature has high mutual value, then it is domain specific. Otherwise, it is domain independent. Furthermore, we require domain-independent features occur frequently. So, we modify the mutual information criterion between features and domains as follows,

$$I(X^i; D) = \sum_{d \in D} \sum_{x \in X^i, x \neq 0} p(x, d) log_2 \left(\frac{p(x, d)}{p(x)p(d)} \right), \tag{6.2}$$

where D is a domain variable and we only sum over non-zero values of a specific feature X^i. The smaller $I(X^i; D)$ is, the more likely that X^i can be treated as a domain-independent feature.

We still use the example shown in Table 6.1 to explain the proposed three strategies for domain-independent features selection. Using the first and third strategies, we may select the words such as "good", "never buy" and "very" as domain-independent features. Here "good" and "never buy" may be good domain-independent features for serving as a bridge across domains. However, the word "very" should not be used a bridge for aligning words from different domains. The reason is that in the *electronics* domain, we may say "this unit is very sharp", while in the *video games* domain, we may say "this game is very boring". If the word "very" is used as a bridge then the words "sharp" and "boring" may be aligned in a same clustering, which is not what we expect. In contrast, using the second strategy, we may be guaranteed to select sentiment words, which is relevant to class labels, such as "good", "nice" and "sharp" (assume the *electronics* domain is a source domain). Note that "sharp" should be selected because it is domain-dependent word. However, the word "sharp" has high mutual value to class labels in the *electronics* domain. Thus, domain-independent feature selection is a challenge task. In a worse case, some words may have opposing sentiment polarity in different domains, which makes the task more challenge. For example, the polarity of the word "thin" may be positive in the *electronics* domain but negative in the *furniture* domain. Hence, in this chapter, we focus more on addressing the problem of how to model the correlation between domain-independent and domain-specific words for transfer learning, which will be presented next. We leave the issue that how to develop a criteria to select domain-independent word precisely in our future work.

6.4.2 Bipartite Feature Graph Construction

Based on the above strategies for selecting domain-independent features, we can identify which features are domain independent and which ones are domain specific. Given domain-independent and domain-specific features, we can construct a bipartite graph $G = (V_{DS} \bigcup V_{DI}, E)$ between them, where we denote $V_{DS} \bigcup V_{DI}$ and E the vertices or edges of the graph G. In G, each vertex in V_{DS} corresponds to a domain-specific word in W_{DS}, and each vertex in V_{DI} corresponds to a domain-independent word in W_{DI}. An edge in E connects two vertexes in V_{DS} and V_{DI} respectively. Note that there is no intra-set edges linking two vertexes in V_{DS} or V_{DI}. Furthermore, each edge $e_{ij} \in E$ is associated with a non-negative weight m_{ij}. The score of m_{ij} measures the relationship between word $w_i \in W_{DS}$ and $w_j \in W_{DI}$ in $\mathcal{D}_S$ and $\mathcal{D}_T$ (e.g., the total number of co-occurrence of $w_i \in W_{DS}$ and $w_j \in W_{DI}$ in $\mathcal{D}_S$ and $\mathcal{D}_T$). A bipartite graph example is shown in Figure 6.1, which is constructed based on the example shown in Table 6.4. Thus, we can use the constructed bipartite graph to model the intrinsic relationship between domain-specific and domain-independent features.

Besides using the co-occurrence frequency of words within documents, we can also adopt more meaningful methods to estimate m_{ij}. For example, we can define a reasonable "window

size", by assuming that if the distance between two words exceeds the "window size", then the correlation between them is very week. Thus, if a domain-specific word and a domain-independent word co-occur within the "window size", then there is an edge connecting them. Furthermore, we can also use the distance between w_i and w_j to adjust the score of m_{ij}. The smaller is their distance, the larger the weight we can assign to the corresponding edge. In this chapter, for simplicity, we set the "window size" to be the maximum length of all documents. Also we do not consider word position to determine the weights for edges. We want to show that by constructing a simple bipartite graph and adapting spectral clustering techniques on it, we can algin domain-specific features effectively.

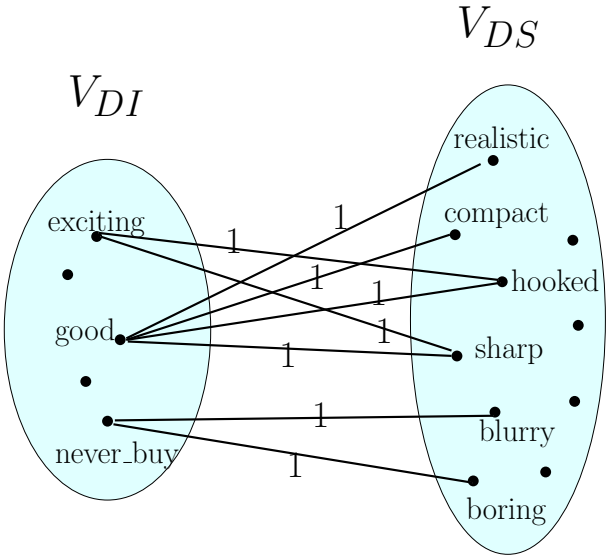


Figure 6.1: A bipartite graph example of domain-specific and domain-independent features.

6.4.3 Spectral Feature Clustering

In the previous section, we have presented how to construct a bipartite graph between domain-specific and domain-independent features. In this section, we show how to adapt a spectral clustering algorithm on the feature bipartite graph to align domain-specific features.

In spectral graph theory [41], there are two main assumptions: (1) if two nodes in a graph are connected to many common nodes, then these two nodes should be very similar (or quite related), (2) there is a low-dimensional latent space underlying a complex graph, where two nodes are similar to each other if they are similar in the original graph. Based on these two assumptions, spectral graph theory has been widely applied in many problems, e.g., dimensionality reduction and clustering [130, 13, 54]. In our case, we assume (1) if two domain-specific features are connected to many common domain-independent features, then they tend to be very related and will be aligned to a same cluster with high probability, (2) if two domain-independent features are connected to many common domain-specific features, then they tend

to be very related and will be aligned to a same cluster with high probability, (3) we can find a more compact and meaningful representation for domain-specific features, which can reduce the gap between domains. Therefore, with the above assumptions, we expect the mismatch problem between domain-specific features can be alleviated by applying graph spectral techniques on the feature bipartite graph to discover a new representation for domain-specific features.

Before we present how to adapt a spectral clustering algorithm to align domain-specific features, we first briefly introduce a standard spectral clustering algorithm [130] as follows,

Given a set of points $V = \{v_1, v_2, ..., v_n\}$ and their corresponding weighted graph G, the goal is to cluster the points into k clusters, where k is an input parameter.

1. Form an affinity matrix for V: $A \in \mathbb{R}^{n \times n}$, where $A_{ij} = m_{ij}$, if $i \neq j$; $A_{ii} = 0$.
2. Form a diagonal matrix D, where $D_{ii} = \sum_j A_{ij}$, and construct the matrix[2]
$$L = D^{-1/2} A D^{-1/2}.$$
3. Find the k largest eigenvectors of L, u_1, u_2, ..., u_k, and form the matrix
$$U = [u_1 u_2 ... u_k] \in \mathbb{R}^{n \times k}.$$
4. Normalize U, such that $U_{ij} = U_{ij} / (\sum_j U_{ij}^2)^{1/2}$.
5. Apply the k-means algorithm on U to cluster the n points into k clusters.

Based on the above description, the standard spectral clustering algorithm clusters n points to k discrete indicators, which can be referred to as "discrete clustering". Zha *et al.* [220] and Ding and He [55] have proven that the k principal components of a term-document co-occurrence matrix, which are referred to as the k largest eigenvectors u_1, u_2, ..., u_k in step 3, are actually the continuous solution of the cluster membership indicators of documents in the k-means clustering method. More specifically, the k principal components can automatically perform data clustering in the subspace spanned by the k principle components. This implies that a mapping function constructed from the k principal components can cluster original data and map them to a new space spanned by the clusters simultaneously. Motivated by this discovery, we show how to adapt the spectral clustering algorithm for cross-domain feature alignment.

Given the feature bipartite graph G, our goal is to learn a feature alignment mapping function $\varphi(\cdot) : \mathbb{R}^{m-l} \rightarrow \mathbb{R}^k$, where m is the number of all features, l is the number of domain-independent features and $m - l$ is the number of domain-specific features.

[2]In spectral graph theory [41] and Laplacian Eigenmaps [13], the Laplacian matrix $\widetilde{L} = I - L$, where I is an identity matrix. The changes in these forms of Laplacian matrix will only change the eigenvalues (from λ_i to $1 - \lambda_i$) but have no impact on eigenvectors. Thus selecting the k smallest eigenvectors of $\widetilde{L}$ in [41, 13] is equivalent to selecting the k largest eigenvectors of L in this paper.

1. Form a weight matrix $M \in \mathbb{R}^{(m-l)\times l}$, where M_{ij} corresponds to the co-occurrence relationship between a domain-specific word $w_i \in W_{DS}$ and a domain-independent word $w_j \in W_{DI}$.

2. Form an affinity matrix $A = \begin{bmatrix} \mathbf{0} & M \\ M^T & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{m\times m}$ of the bipartite graph, where the first $m-l$ rows and columns correspond to the $m-l$ domain-specific features, and the last l rows and columns correspond to the l domain-independent features.

3. Form a diagonal matrix D, where $D_{ii} = \sum_j A_{ij}$, and construct the matrix

$$L = D^{-1/2} A D^{-1/2}.$$

4. Find the k largest eigenvectors of L, u_1, u_2, ..., u_k, and form the matrix

$$U = [u_1 u_2 ... u_k] \in \mathbb{R}^{m\times k}.$$

5. Define the feature alignment mapping function as

$$\varphi(x) = x U_{[1:m-l,:]},$$

where $U_{[1:m-l,:]}$ denotes the first $m-l$ rows of U and $x \in \mathbb{R}^{1\times(m-l)}$.

Given a feature alignment mapping function $\varphi(\cdot)$, for a data example x_i in either a source domain or target domain, we can first apply $\phi_{DS}(\cdot)$ to identify the view associated with domain-specific features of x_i, and then apply $\varphi(\cdot)$ to find a new representation $\varphi(\phi_{DS}(x_i))$ of the view of domain-specific features of x_i. Note that the affinity matrix A constructed in Step 2 is similar to the affinity matrix of a term-document bipartite graph proposed in [54], which is used for spectral co-clustering terms and documents simultaneously. Though our goal is only to cluster domain-specific features, it is proved that clustering two related sets of points simultaneously can often get better results than only clustering one single set of points [54].

6.4.4 Feature Augmentation

If we have selected domain-independent features and aligned domain-specific features perfectly, then we can simply augment domain-independent features with features via feature alignment to generate a perfect representation for cross-domain sentiment classification. However, in practice, we may not be able to identify domain-independent features correctly and thus fail to perform feature alignment perfectly. Similar to the strategy used in [4, 25], we augment all original features with features learned by feature alignment to construct a new representation. A tradeoff parameter γ is used in this feature augmentation to balance the effect of original

features and new features. Thus, for each data example x_i, the new feature representation is defined as

$$\widetilde{x}_i = [x_i,\ \gamma\varphi(\phi_{DS}(x_i))],$$

where $x_i \in \mathbb{R}^{1\times m}$, $\widetilde{x}_i \in \mathbb{R}^{1\times m+k}$ and $0 \leq \gamma \leq 1$. In practice, the value of λ can be determined by evaluation on some heldout data. The whole process of our proposed framework for cross-domain sentiment classification is presented in Algorithm 6.1.

Algorithm 6.1 Spectral Feature Alignment (SFA) for cross-domain sentiment classification

Require: A labeled source domain data set $\mathcal{D}_S = \{(x_{S_i}, y_{S_i})\}_{i=1}^{n_S}$, an unlabeled target domain data set $\mathcal{D}_T = \{x_{T_j}\}_{j=1}^{n_T}$, the number of clusters K and the number of domain-independent features m.

Ensure: Predicted labels Y_T of the unlabeled data X_T in the target domain.

1: Apply the criteria mentioned in Chapter 6.4.1 on $\mathcal{D}_S$ and $\mathcal{D}_T$ to select l domain-independent features. The remaining $m - l$ features are treated as domain-specific features.

$$\Phi_{DI} = \begin{bmatrix} \phi_{DI}(x_S) \\ \phi_{DI}(x_T) \end{bmatrix},\ \Phi_{DS} = \begin{bmatrix} \phi_{DS}(x_S) \\ \phi_{DS}(x_T) \end{bmatrix}.$$

2: By using Φ_{DI} and Φ_{DS}, calculate (*DI-word*)-(*DS-word*) co-occurrence matrix $M \in \mathbb{R}^{(m-l)\times l}$.

3: Construct matrix $L = D^{-1/2}AD^{-1/2}$, where $A = \begin{bmatrix} \mathbf{0} & M \\ M^T & \mathbf{0} \end{bmatrix}$.

4: Find the K largest eigenvectors of L, u_1, u_2, ..., u_K, and form the matrix

$$U = [u_1 u_2 ... u_K] \in \mathbb{R}^{m\times K}.$$

Let mapping $\varphi(x_i) = x_i U_{[1:m-l,:]}$, where $x_i \in \mathbb{R}^{m-l}$

5: Train a classifier f on

$$\{(\widetilde{x}_{S_i}, y_{S_i})\}_{i=1}^{n_S} = \{([x_{S_i},\ \gamma\varphi(\phi_{DS}(x_{S_i}))], y_{S_i})\}_{i=1}^{n_S}$$

6: **return** $f(\widetilde{x}_{T_i})$'s.

As can be seen in the algorithm, in the first step, we select a set of domain-independent features using one of the three strategies introduced in Chapter 6.4.1. In the second step, we calculate the corresponding co-occurrence matrix M and then construct a normalized matrix L corresponding to the bipartite graph of the domain-independent and domain-specific features. Eigen-decomposition is performed on L to find the K leading eigenvectors to construct a mapping ϕ. Finally training and testing are both performed on augmented representations.

6.5 Computational Issues

Note the computational cost of the SFA algorithm is maintained by an eigen-decomposition of the matrix $L \in \mathbb{R}^{m\times m}$. In general, it takes $O(Km^2)$ [175], where m is the total number

of features and K is the dimensionality in the latent space. If m is large, then it would be computationally expensive. However, if the matrix L is sparse, then we can still apply the Implicitly Restarted Arnoldi method to solve the eigen-decomposition of L iteratively and efficiently [175]. The computational time is $O(Kpm)$ approximately, where p is the number iterations in the Implicitly Restarted Arnoldi method defined by users. In practice, the bipartite feature graph defined in Chapter 6.4.2 is extremely sparse. As a result, the matrix L is sparse as well. Hence, the eigen-decomposition of L can be solved efficiently.

6.6 Connection to Other Methods

Different from the state-of-the-art cross-domain sentiment classification algorithms such as the SCLalgorithm, which only learns common structures underlying domain-specific words without fully exploiting the relationship between domain-independent and domain-specific words, SFA can better capture the intrinsic relationship between domain-independent and domain-specific words via the bipartite graph representation and learn a more compact and meaningful representation underlying the graph via co-aligning domain-independent and domain-specific words. Experiments in two real world domains indicate that SFA is indeed promising in obtaining better performance than several baselines including SCL in terms of the accuracy for cross-domain sentiment classification.

6.7 Experiments

In this section, we describe our experiments on two real-world datasets and show the effectiveness of our proposed SFA for cross-domain sentiment classification.

6.7.1 Experimental Setup

Datasets

In this section, we first describe the datasets used in our experiments. The first dataset is from Blitzer *et al.* [25]. It contains a collection of product reviews from Amazon.com. The reviews are about four product domains: *books* (**B**), *dvds* (**D**), *electronics* (**E**) and *kitchen appliances* (**K**). Each review is assigned a sentiment label, -1 (negative review) or $+1$ (positive review), based on the rating score given by the review author. In each domain, there are $1,000$ positive reviews and $1,000$ negative ones. In this dataset, we can construct 12 cross-domain sentiment classification tasks from with source: **D** $\rightarrow$ **B**, **E** $\rightarrow$ **B**, **K** $\rightarrow$ **B**, **K** $\rightarrow$ **E**, **D** $\rightarrow$ **E**, **B** $\rightarrow$ **E**, **B** $\rightarrow$ **D**, **K** $\rightarrow$ **D**, **E** $\rightarrow$ **D**, **B** $\rightarrow$ **K**, **D** $\rightarrow$ **K**, **E** $\rightarrow$ **K**, where the word before an arrow corresponds with the source domain and the word after an arrow corresponds with the target domain. We use

$RevDat$ to denote this dataset. The sentiment classification task on this dataset is document-level sentiment classification.

The other dataset is collected by us for experiment purpose. We have crawled a set of reviews from Amazon[3] and TripAdvisor[4] websites. The reviews from Amazon are about three product domains: *video game* (**V**), *electronics* (**E**) and *software* (**S**). The TripAdvisor reviews are about the *hotel* (**H**) domain. Instead of assigning each review with a label, we split these reviews into sentences and manually assign a polarity label for each sentence. In each domain, we randomly select $1,500$ positive sentences and $1,500$ negative ones for experiment. Similarly, we also construct 12 cross-domain sentiment classification tasks: **V** $\rightarrow$ **H**, **V** $\rightarrow$ **E**, **V** $\rightarrow$ **S**, **S** $\rightarrow$ **E**, **S** $\rightarrow$ **V**, **S** $\rightarrow$ **H**, **E** $\rightarrow$ **V**, **E** $\rightarrow$ **H**, **E** $\rightarrow$ **S**, **H** $\rightarrow$ **S**, **H** $\rightarrow$ **E**, **S** $\rightarrow$ **V**. We use $SentDat$ to denote this dataset. Sentiment classification task on this dataset is sentence-level sentiment classification. For both datasets, we use Unigram and Bigram features to represent each data example (a review in $RevDat$ and a sentence in $SentDat$). The summary of the datasets is described in Table 6.5.

Table 6.5: Summary of datasets for evaluation.

Dataset	Domain	# Reviews	# Pos	# Neg	# Features
$RevDat$	dvds	$2,000$	$1,000$	$1,000$	$473,856$
	kitchen	$2,000$	$1,000$	$1,000$	
	electronics	$2,000$	$1,000$	$1,000$	
	books	$2,000$	$1,000$	$1,000$	
$SentDat$	video game	$3,000$	$1,500$	$1,500$	$287,504$
	hotel	$3,000$	$1,500$	$1,500$	
	software	$3,000$	$1,500$	$1,500$	
	electronics	$3,000$	$1,500$	$1,500$	

Baselines

In order to investigate the effectiveness of our method, we have compared it with several algorithms. In this section, we describe some baseline algorithms with which we compare SFA. One baseline method denoted by NoTransf, is a classifier trained directly with the source domain training data. The gold standard (denoted by upperBound) is an in-domain classifier trained with labeled data from the target domain. For example, for **D** $\rightarrow$ **B** task, NoTransf means that we train a classifier with labeled data of **D** domain. upperBound corresponds with a classifier trained with the labeled data from **B** domain. So, the performance of upperBound in **D** $\rightarrow$ **B** task can be also regarded as an upper bound of **E** $\rightarrow$ **B** and **K** $\rightarrow$ **B** tasks. Another baseline method denoted by PCA is a classifier trained on a new representation which augments original

[3]http://www.amazon.com/

[4]http://www.tripadvisor.com/

features with new features which are learned by applying latent semantic analysis (also can be referred to as principal component analysis) [53] on the original view of domain-specific features (as shown in Table 6.2). A third baseline method denoted by FALSA is a classifier trained on a new representation which augments original features with new features which are learned by applying latent semantic analysis on the co-occurrence matrix of domain-independent and domain-specific features. We compare our method with PCA and FALSA in order to investigate if spectral feature clustering is effective in aligning domain-specific features. We have also compared our algorithm with a method: structural correspondence learning (SCL) proposed in [25]. We follow the details described in Blitzer's thesis [23] to implement SCL with logistic regression to construct auxiliary tasks. Note that SCL, PCA, FALSA and our proposed SFA all use unlabeled data from the source and target domains to learn a new representation and train classifiers using the labeled source domain data with new representations.

Parameter Settings & Evaluation Criteria

For NoTransf, upperBound, PCA, FALSA and SFA, we use logistic regression as the basic sentiment classifier. The library implemented in [64] is used in all our experiments. The tradeoff parameter C in logistic regression [64] is set to be 10000, which is equivalent to set $\lambda = 0.0001$ in [23]. The parameters of each model are tuned on some heldout data in **E** $\rightarrow$ **B** task of $RevDat$ and **H** $\rightarrow$ **S** task of $SentDat$, and are fixed to be used in all experiments. We use accuracy to evaluate the sentiment classification result: the percentage of correctly classified examples over all testing examples. The definition of accuracy is given as follows,

$$Accuracy = \frac{|\{x|x \in \mathcal{D}_{tst} \cap f(x) = y\}|}{|\{x|x \in \mathcal{D}_{tst}\}|},$$

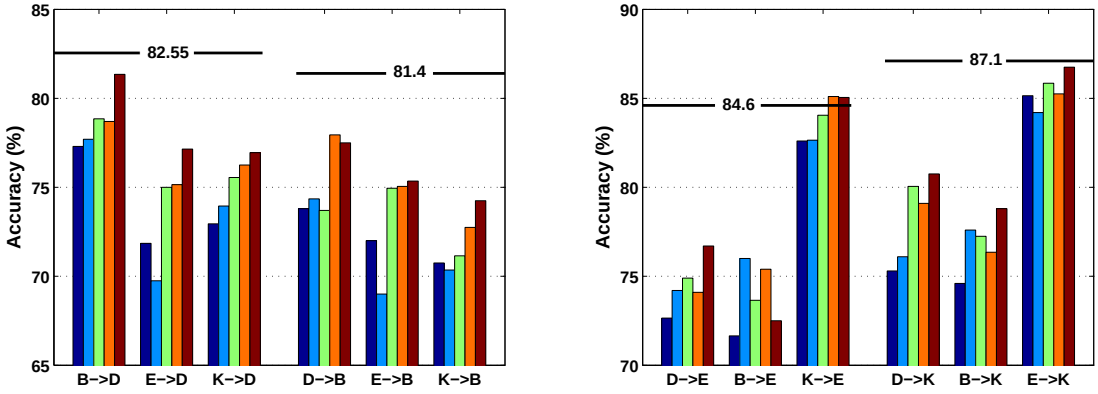
where $\mathcal{D}_{tst}$ denotes the test data, y is the ground truth sentiment polarity and $f(x)$ is the predicted sentiment polarity. For all experiments on $RevDat$, we randomly split each domain data into a training set of 1,600 instances and a test set of 400 instances. For all experiments on $SentDat$, we randomly split each domain data into a training set of 2,000 instances and a test set of 1,000 instances. The evaluation of cross-domain sentiment classification methods is conducted on the test set in the target domain without labeled training data in the same domain. We report the average results of 5 random trails.

6.7.2 Results

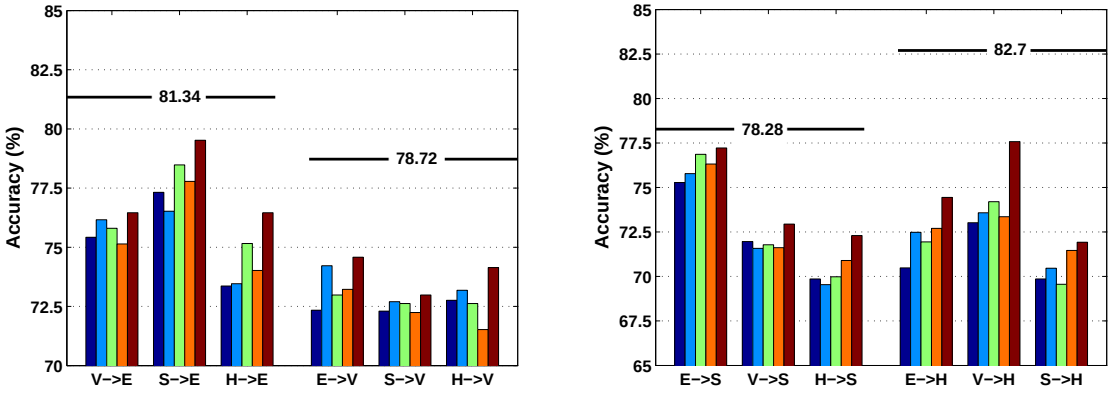
Overall Comparison Results between SFA and Baselines

In this section we compare the accuracy of SFA with NoTransf, PCA, FALSA and SCL by 24 tasks on two datasets. For PCA, FALSA and SFA, we use Eqn.(6.2) defined in Chapter 6.4.1

to identity domain-independent and domain-specific features. We adopt the following settings: the number of domain-independent features $l = 500$, the number of domain-specific features clusters $k = 100$ and the parameter in feature augmentation $\gamma = 0.6$. Studies of the SFA parameters are presented in Chapters 6.7.2 and 6.7.2. For SCL, we use mutual information to select "pivots". The number of "pivots" is set to be 500, and the number of dimensionality h in [25] is set to be 50. All these parameters and domain-independent feature (or "pivot") selection methods are determined based on results on the heldout data mentioned in the previous section.



(a) Comparison Results on $RevDat$.



(b) Comparison Results on $SentDat$.

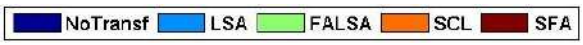


Figure 6.2: Comparison results (unit: %) on two datasets.

Figure 6.2(a) shows the comparison results of different methods on $RevDat$. In the figure, each group of bars represents a cross-domain sentiment classification task. Each bar in specific color represents a specific method. The horizontal lines are accuracies of upperBound. From the figure, we can observe that the four domains of $RevDat$ can be roughly classified into two groups: **B** and **D** domains are similar to each other, as are **K** and **E**, but the two groups are different from each other. Adapting a classifier from **K** domain to **E** domain is much easier than adapting it from **B** domain. Clearly, our proposed SFA performs better than other methods

including state-of-the-art method SCL in most tasks. As mentioned in Chapter 6.3, clustering domain-specific features with bag-of-words representation may fail to find a meaningful new representation for cross-domain sentiment classification. Thus PCA only outperforms NoTransf slightly in some tasks, but its performance may even drop on other tasks. It is not surprising to find that FALSA gets significant improvement compared to NoTransf and PCA. The reason is that representing domain-specific features via domain-independent features can reduce the gap between domains and thus find a reasonable representation for cross-domain sentiment classification. Our proposed SFA can not only utilize the co-occurrence relationship between domain-independent and domain-specific features to reduce the gap between domains, but also use graph spectral clustering techniques to co-align both kinds of features to discover meaningful clusters for domain-specific features. Though our goal is only to cluster domain-specific features, it has been proved that clustering two related sets of points simultaneously can often get better results than clustering one single set of points only [54].

From the comparison results on $SentDat$ shown in Figure 6.2(b), we can get similar conclusion: SFA outperforms other methods in most tasks. One interesting observation from the results is that SCL does not work well compared to its performance on $RevDat$. One reason may be that in sentence-level sentiment classification, the data are quite sparse. In this case, it is hard to construct a reasonable number of auxiliary tasks that are useful to model the relationship between "pivots" and "non-pivots". The performance of SCL highly relies on the auxiliary tasks. Thus in this dataset, SCL even performs worse than FALSA in some tasks. We do **t-test** on the comparison results of the two datasets and find that SFA outperforms other methods with 0.95 confidence intervals.

Effect of Domain Independent Features of SFA

In this section, we conduct two experiments to study the effect of domain-independent features on the performance of SFA. The first experiment is to test the effect of domain-independent features identified by different methods on the overall performance of SFA. The second one is to test the effect of different numbers of domain-independent features on SFA performance. As mentioned in Chapter 6.4.1, besides using Eqn. (6.2) to identify domain-independent and domain-specific features, we can also use the other two strategies to identify them. In Table 6.6, we summarize the comparison results of SFA using different methods to identify domain-independent features. We use SFA$_{DI}$, SFA$_{FQ}$ and SFA$_{MI}$ to denote SFA using Eqn. (6.2), frequency of features in both domains and mutual information between features and labels in the source domain respectively. From the table, we can observe that SFA$_{DI}$ and SFA$_{FQ}$ achieve comparable results and they are stable in most tasks. While SFA$_{MI}$ may work very well in some tasks such as **K** $\rightarrow$ **D** and **E** $\rightarrow$ **B** of $RevDat$, but work very bad in some tasks such as **E** $\rightarrow$ **D** and **D** $\rightarrow$ **E** of $RevDat$. The reason is that applying mutual information on source domain data

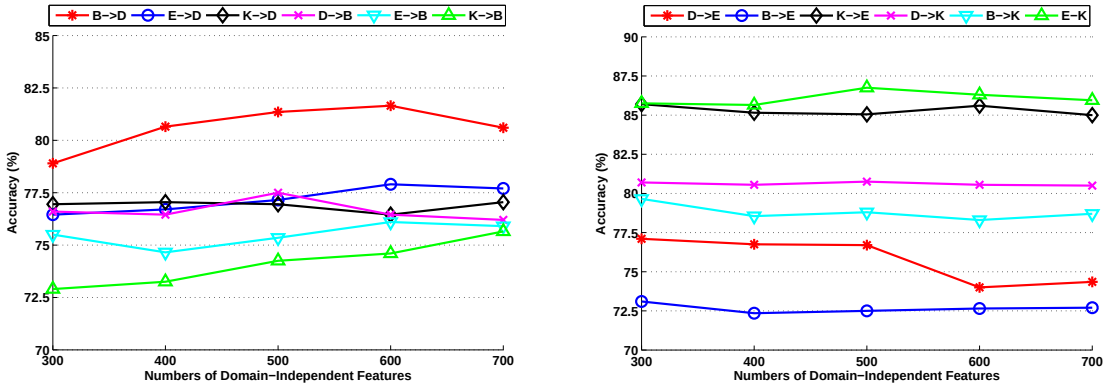
can find features that are relevant to the source domain labels but cannot guarantee the selected features to be domain independent. In addition, the selected features may be irrelevant to the labels of the target domain. To test the effect of the number of domain-independent features on the performance of SFA, we apply SFA on all 24 tasks in the two datasets, and fix $k = 100$, $\gamma = 0.6$. The value of l is changed from 300 to 700 with step length 100. The results are shown in Figure 6.3(a) and 6.3(b). From the figures, we can find that when l is in the range of $[400, 700]$, SFA performs well and stably in most tasks. Thus SFA is robust to the quality and numbers of domain-independent features.

Table 6.6: Experiments with different domain-independent feature selection methods. Numbers in the table are accuracies in percentage.

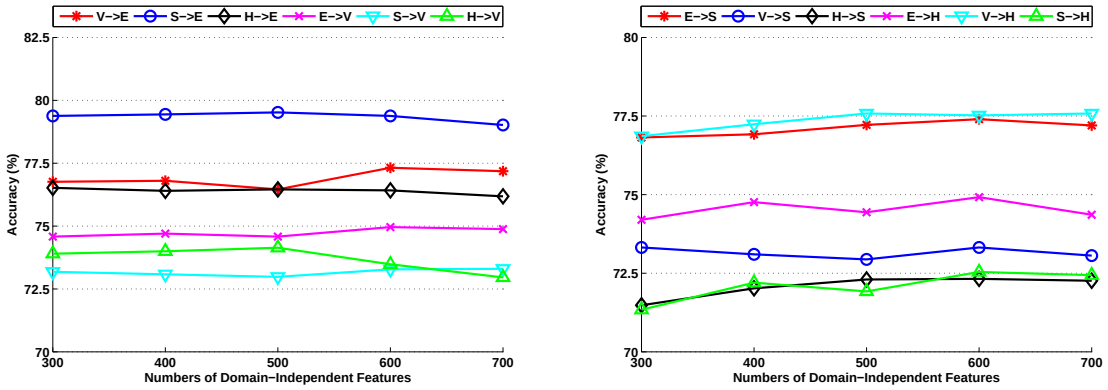
$RevDat$						
	B→D	**E→D**	**K→D**	**D→B**	**E→B**	**K→B**
SFA$_{DI}$	81.35	77.15	76.95	77.5	75.65	74.8
SFA$_{FQ}$	81.25	77	76.6	78.25	75.35	74.25
SFA$_{MI}$	80.1	70.4	78.45	79.8	78.25	75.15
SFA$_{DI+MI}$	81.55	78.90	77.85	78.65	77.80	76.35
SFA$_{FQ+MI}$	81.45	79.55	78.45	78.50	77.55	75.75
	D→E	**B→E**	**K→E**	**D→K**	**B→K**	**E→K**
SFA$_{DI}$	76.7	72.5	85.05	80.75	78.8	86.75
SFA$_{FQ}$	76.05	73.45	84.9	80.6	79.05	85.8
SFA$_{MI}$	70.85	73	82.05	78.9	78.8	86.75
SFA$_{DI+MI}$	75.65	77.15	85.90	81.05	79.80	87.55
SFA$_{FQ+MI}$	74.80	77.10	85.95	81.15	79.80	86.55
$SentDat$						
	V→E	**S→E**	**H→E**	**E→V**	**S→V**	**H→V**
SFA$_{DI}$	76.64	79.52	76.46	74.58	72.98	74.14
SFA$_{FQ}$	76.62	79.5	76.64	74.38	73.16	74.54
SFA$_{MI}$	76.96	79.08	76.46	75.06	73.86	74.88
SFA$_{DI+MI}$	76.64	79.52	76.46	74.58	72.98	74.14
SFA$_{FQ+MI}$	76.62	79.5	76.64	74.38	73.16	74.54
	E→S	**V→S**	**H→S**	**E→H**	**V→H**	**S→H**
SFA$_{DI}$	77.22	72.94	72.3	74.44	77.58	71.92
SFA$_{FQ}$	77.5	72.96	72.22	75	77.46	71.62
SFA$_{MI}$	77.48	73.22	72.38	75.98	77.08	72.46
SFA$_{DI+MI}$	77.22	72.94	72.3	74.44	77.58	71.92
SFA$_{FQ+MI}$	77.5	72.96	72.22	75	77.46	71.62

Model Parameter Sensitivity of SFA

Besides the number of domain-independent features l, there are two other parameters in SFA: one of them is the number of domain-specific feature-clusters k and the other is the tradeoff parameter λ in feature augmentation. In this section, we further test the sensitivity of these two parameters on the overall performance of SFA.



(a) Results on $RevDat$.

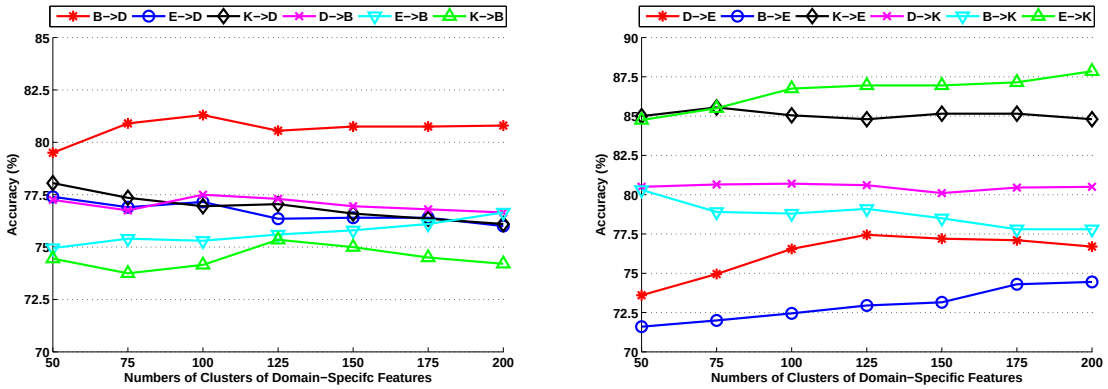


(b) Results on $SentDat$.

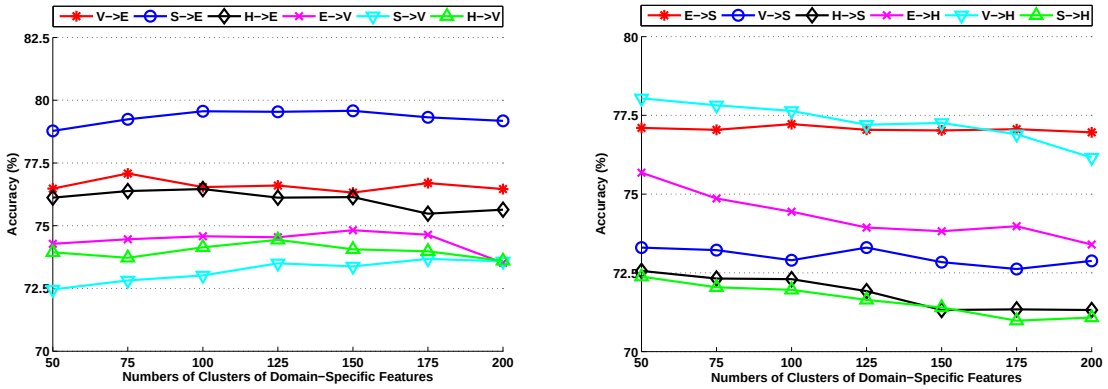
Figure 6.3: Study on varying numbers of domain-independent features of SFA

We first test the sensitivity of the parameter k. In this experiment, we fix $l = 500$, $\gamma = 0.6$ and change the value of k from 50 to 200 with step length 25. Figure 6.4(a) and 6.4(b) show the results of SFA under varying values of k. Note that though for some tasks such as $\mathbf{V} \rightarrow \mathbf{H}$, $\mathbf{H} \rightarrow \mathbf{S}$ and $\mathbf{E} \rightarrow \mathbf{H}$, SFA gets worse performance when the value of k increases, in most task, when the cluster number k falls in the range from 75 to 175, SFA performs well and stably.

Finally, we test the sensitivity of the parameter γ. In this experiment, we fix $l = 500$, $k = 100$ and change the values of γ from 0.1 to 1 with step length 0.1. Results are shown in Figure 6.5(a) and 6.5(b). Apparently, when $\gamma < 0.2$, which means the original features dominate the effect of the new feature representation, SFA works badly in most tasks. However, when $\gamma \geq 0.3$, SFA works stably in most tasks, which implies that the features learned by the spectral feature alignment algorithm do boost the performance in cross-domain sentiment classification.



(a) Results on $RevDat$ under Varying Values of k.

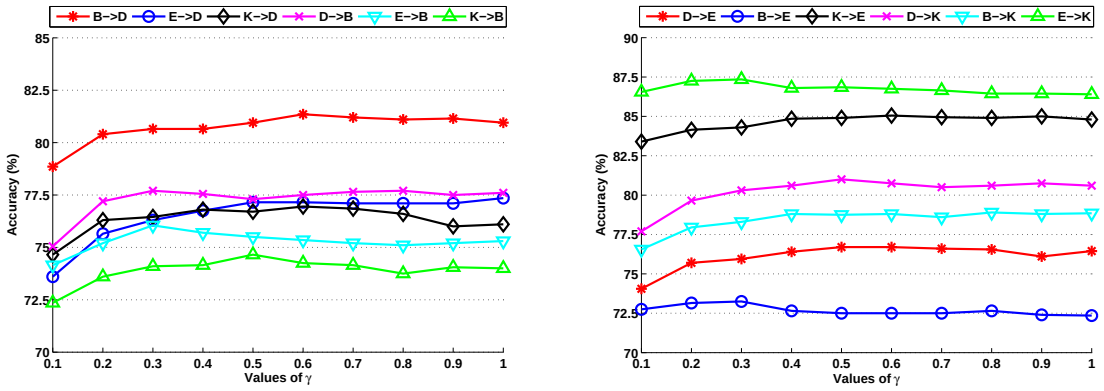


(b) Results on $SentDat$ under Varying Values of k.

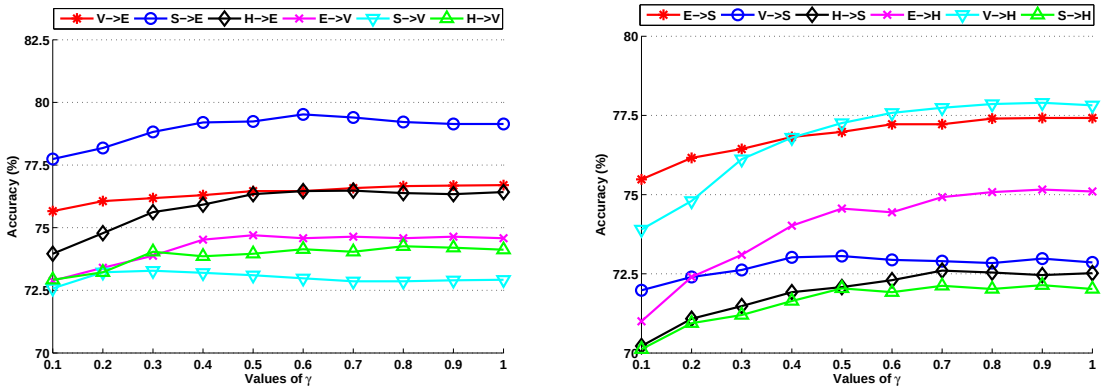
Figure 6.4: Model parameter sensitivity study of k on two datasets.

6.8 Summary

In this section, we have studied the problem of how to develop a latent space for transfer learning when explicit domain knowledge is available in some specific applications. We proposed a spectral feature alignment (SFA) algorithm for sentiment classification. Experimental results showed that our proposed method outperforms a state-of-the-art method in cross-domain sentiment classification. One limitation of SFA is that it is domain-driven, which means it is hard to be applied to other applications. However, the success of SFA in sentiment classification suggests that in some domain areas, where domain knowledge is available or can be easy to obtain, to develop a domain-driven method may be more effective than general solutions for learning the latent space for transfer learning.



(a) Results on $RevDat$ under Varying Values of γ.



(b) Results on $SentDat$ under Varying Values of γ.

Figure 6.5: Model parameter sensitivity study of γ on two datasets.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

Transfer learning is still at an early but promising stage. There exist many research issues needed to be addressed. In this thesis, we first surveyed the field of transfer learning, where we give some definitions of transfer learning, summarize transfer learning into different settings, categorize transfer learning approaches into four contexts, analyze the relationship between transfer learning and other related areas, discuss some interesting research issues in transfer learning and introduce some applications of transfer learning.

In this thesis, we focus on the transductive transfer learning setting, and propose two different feature-based transfer learning frameworks based different situations: (1) domain knowledge is hidden or hard to capture, and (2) domain knowledge can be observed or easy to encode into learning. In the first situation, we proposed a novel and general dimensionality reduction framework for transfer learning, which aims to learn a latent space to reduce the distance between domains and preserve properties of original data simultaneously. Based on this framework, we propose three different algorithms to learn the latent space, known as Maximum Mean Discrepancy Embedding (MMDE) [135], Transfer Component Analysis (TCA) [139] and Semi-supervised Transfer Component Analysis (SSTCA) [140]. MMDE learns the latent by learning a non-parametric kernel matrix, which may be more precise. However, the non-parametric kernel matrix learning results in a need to solve a semi-definite program (SDP), which is extremely expensive ($O((n_S + n_T)^{6.5})$). In contrast, TCA and SSTCA learn a parametric kernel matrix instead, which may make the latent space less precise, but avoid the use of a SDP solver, which is much more efficient ($O(m(n_S + n_T)^{6.5})$, where $m \ll n_S + n_T$). Hence, in practice, TCA and SSTCA are more applicable. We apply these three algorithms to two diverse applications: WiFi localization and text classification, and achieve promising results.

In order to capture and apply the domain knowledge inherent in many application areas, we also propose a Spectral Feature Alignment (SFA) [137] algorithm for cross-domain sentiment classification, which aims at aligning domain-specific words from different domains in a latent space by modeling the correlation between the domain-independent and domain-specific words in a bipartite graph and using the domain-specific features as a bridge. Experimental results show the great classification performance of SFA in cross-domain sentiment classification.

7.2 Future Work

In the future, we plan to continue to explore our work on transfer learning along the following directions.

- We plan to study the proposed dimensionality reduction framework for transfer learning theoretically. Research issues include: (1) how to determine the number of the reduced dimensionalities adaptively? (2) How to get a generalization error bound of the proposed framework? (3) How to develop an efficient algorithm for kernel choice of TCA and SSTCA based on the recent theoretic study in [176].

- We plan to extend the dimensionality reduction framework in a relational learning manner, which means that data in source and target domains can be relational instead of being independent distributed. Most previous transfer learning methods assumed data from different domains to be independent distributed. However, in many real-world applications, such as link prediction in social networks or classifying Web pages within a Web site, data are often intrinsically relational, which present a major challenge to transfer learning [52]. We plan to develop a relational dimensionality reduction framework, which can encode the relational information among data into the latent space learning, while can also reduce the difference between domains and preserve properties of the original data.

- We plan to study the negative transfer learning issue. It has been proven that when the source and target tasks are unrelated, the knowledge extracted from a source task may not help, and even hurt, the performance of the target task [159]. Thus, how to avoid negative transfer and then ensure safe transfer of knowledge is crucial in transfer learning. Recently, in [30], we have proposed an Adaptive Transfer learning algorithm based on Gaussian Processes (AT-GP), which can be used to adapt the transfer learning schemes by automatically estimating the similarity between a source and a target task. Preliminary experimental results show that the proposed algorithm is promising to avoid negative transfer in transfer learning. In the future, we plan to develop a criterion to measure whether there exists a latent space, onto which knowledge can be safely transferred across domain/tasks. Based on the criterion, we can develop a general framework to automatically identify whether the source and target tasks are related or not, and determine whether we should project the data from the source and target domains onto the latent space.

REFERENCES

[1] Ahmed Abbasi, Hsinchun Chen, and Arab Salem. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Transaction Information System*, 26(3):1–34, June 2008.

[2] Eneko Agirre and Oier Lopez de Lacalle. On robustness and domain adaptation using svd for word sense disambiguation. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 17–24. ACL, June 2008.

[3] Morteza Alamgir, Moritz Grosse-Wentrup, and Yasemin Altun. Multitask learning for brain-computer interfaces. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, volume 9, pages 17–24. JMLR W&CP, May 2010.

[4] Rie K. Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[5] Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 1–9. ACL, June 2005.

[6] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems 19*, pages 41–48. MIT Press, 2007.

[7] Andreas Argyriou, Andreas Maurer, and Massimiliano Pontil. An algorithm for transfer learning in a heterogeneous environment. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 71–85. Springer, September 2008.

[8] Andreas Argyriou, Charles A. Micchelli, Massimiliano Pontil, and Yiming Ying. A spectral regularization framework for multi-task structure learning. In *Annual Conference on Neural Information Processing Systems 20*, pages 25–32. MIT Press, 2008.

[9] Andrew Arnold, Ramesh Nallapati, and William W. Cohen. A comparative study of methods for transductive transfer learning. In *Proceedings of the 7th IEEE International Conference on Data Mining Workshops*, pages 77–82. IEEE Computer Society, 2007.

[10] Jing Bai, Ke Zhou, Guirong Xue, Hongyuan Zha, Gordon Sun, Belle Tseng, Zhaohui Zheng, and Yi Chang. Multi-task learning for learning to rank in web search. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1549–1552. ACM, 2009.

[11] Bart Bakker and Tom Heskes. Task clustering and gating for bayesian multitask learning. *Journal of Machine Learning Reserch*, 4:83–99, 2003.

[12] Maxim Batalin, Gaurav Sukhatme, and Myron Hattig. Mobile robot navigation using a sensor network. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 636–642, April 2003.

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

[14] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, November 2006.

[15] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jenn Wortman. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.

[16] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Annual Conference on Neural Information Processing Systems 19*, pages 137–144. MIT Press, 2007.

[17] Shai Ben-David, Tyler Lu, Teresa Luu, and David Pal. Impossibility theorems for domain adaptation. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, volume 9, pages 129–136. JMLR W&CP, May 2010.

[18] Shai Ben-David and Reba Schuller. Exploiting task relatedness for multiple task learning. In *Proceedings of the Sixteenth Annual Conference on Learning Theory*, pages 825–830. Morgan Kaufmann Publishers Inc., August 2003.

[19] Indrajit Bhattacharya, Shantanu Godbole, Sachindra Joshi, and Ashish Verma. Cross-guided clustering: Transfer of relevant supervision across domains for improved clustering. In *Proceedings of the 9th IEEE International Conference on Data Mining*, pages 41–50. IEEE Computer Society, December 2009.

[20] Steffen Bickel, Jasmina Bogojeska, Thomas Lengauer, and Tobias Scheffer. Multi-task learning for HIV therapy screening. In *Proceedings of the 25th International Conference on Machine learning*, pages 56–63. ACM, July 2008.

[21] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning*, pages 81–88. ACM, June 2007.

[22] Steffen Bickel, Christoph Sawade, and Tobias Scheffer. Transfer learning by distribution matching for targeted advertising. In *Advances in Neural Information Processing Systems 21*, pages 145–152. 2009.

[23] John Blitzer. *Domain Adaptation of Natural Language Processing Systems*. PhD thesis, The University of Pennsylvania, 2007.

[24] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jenn Wortman. Learning bounds for domain adaptation. In *Annual Conference on Neural Information Processing Systems 20*, pages 129–136. MIT Press, 2008.

[25] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 432–439. ACL, June 2007.

[26] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the Conference on Empirical Methods in Natural Language*, pages 120–128. ACL, July 2006.

[27] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Learning Theory*, pages 92–100, July 1998.

[28] Edwin Bonilla, Kian Ming Chai, and Chris Williams. Multi-task gaussian process prediction. In *Annual Conference on Neural Information Processing Systems 20*, pages 153–160. MIT Press, 2008.

[29] Bin Cao, Nathan N. Liu, and Qiang Yang. Transfer learning for collective link prediction in multiple heterogenous domains. In *Proceedings of the 27th International Conference on Machine learning*. ACM, June 2010.

[30] Bin Cao, Sinno Jialin Pan, Yu Zhang, Dit-Yan Yeung, and Qiang Yang. Adaptive transfer learning. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*. AAAI Press, July 2010.

[31] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[32] Kian Ming A. Chai, Christopher K. I. Williams, Stefan Klanke, and Sethu Vijayakumar. Multi-task gaussian process learning of robot inverse dynamics. In *Advances in Neural Information Processing Systems 21*, pages 265–272. 2009.

[33] Yee Seng Chan and Hwee Tou Ng. Domain adaptation with active learning for word sense disambiguation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 49–56. ACL, June 2007.

[34] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, USA, 2006.

[35] Olivier Chapelle, Pannagadatta Shivaswamy, Srinivas Vadrevu, Kilian Weinberger, Ya Zhang, and Belle Tseng. Multi-task learning for boosting with application to web search ranking. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1189–1198. ACM, July 2010.

[36] Bo Chen, Wai Lam, Ivor W. Tsang, and Tak-Lam Wong. Extracting discriminative concepts for domain adaptation in text mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 179–188. ACM, June 2009.

[37] Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. A convex formulation for learning shared structures from multiple tasks. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 137–144. ACM, June 2009.

[38] Lin Chen, Dong Xu, Ivor W. Tsang, and Jiebo Luo. Tag-based web photo retrieval improved by batch mode re-tagging. In *Proceedings of the 23rd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010.

[39] Tianqi Chen, Jun Yan, Gui-Rong Xue, and Zheng Chen. Transfer learning for behavioral targeting. In *Proceedings of the 19th International Conference on World Wide Web*, pages 1077–1078. ACM, April 2010.

[40] Yuqiang Chen, Ou Jin, Gui-Rong Xue, Jia Chen, and Qiang Yang. Visual contextual advertising: Bringing textual advertisements to images. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*. AAAI Press, July 2010.

[41] Fan R. K. Chung. *Spectral Graph Theory*. Number 92 in CBMS Regional Conference Series in Mathematics. American Mathematical Society, 1997.

[42] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, July 2008.

[43] Nello Cristianini, Jaz Kandola, Andre Elisseeff, and John Shawe-Taylor. On kernel-target alignment. In *Advances in Neural Information Processing Systems 14*, pages 367–373. MIT Press, 2002.

[44] W. Bruce Croft and John Lafferty. *Language Modeling for Information Retrieval*. Kluwer Academic Publishers, Norwell, MA, USA, 2003.

[45] Wenyuan Dai, Yuqiang Chen, Gui-Rong Xue, Qiang Yang, and Yong Yu. Translated learning: Transfer learning across different feature spaces. In *Annual Conference on Neural Information Processing Systems 21*, pages 353–360. MIT Press, 2009.

[46] Wenyuan Dai, Ou Jin, Gui-Rong Xue, Qiang Yang, and Yong Yu. Eigentransfer: a unified framework for transfer learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 25–31. ACM, June 2009.

[47] Wenyuan Dai, Guirong Xue, Qiang Yang, and Yong Yu. Co-clustering based classification for out-of-domain documents. In *Proceedings of the 13th ACM International Conference on Knowledge Discovery and Data Mining*. ACM, August 2007.

[48] Wenyuan Dai, Guirong Xue, Qiang Yang, and Yong Yu. Transferring naive bayes classifiers for text classification. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 540–545. AAAI Press, July 2007.

[49] Wenyuan Dai, Qiang Yang, Guirong Xue, and Yong Yu. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning*, pages 193–200. ACM, June 2007.

[50] Wenyuan Dai, Qiang Yang, Guirong Xue, and Yong Yu. Self-taught clustering. In *Proceedings of the 25th International Conference of Machine Learning*, pages 200–207. ACM, July 2008.

[51] Hal Daumé III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263. ACL, June 2007.

[52] Jesse Davis and Pedro Domingos. Deep transfer via second-order markov logic. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 217–224. ACM, June 2009.

[53] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[54] Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274. ACM, July 2001.

[55] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, pages 225–232. ACM, July 2004.

[56] Harris Drucker. Improving regressors using boosting techniques. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 107–115. Morgan Kaufmann Publishers Inc., July 1997.

[57] Lixin Duan, Ivor W. Tsang, Dong Xu, and Tat-Seng Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the 26th International Conference on Machine Learning*, pages 289–296. ACM, June 2009.

[58] Lixin Duan, Ivor W. Tsang, Dong Xu, and Stephen J. Maybank. Domain transfer svm for video concept detection. In *Proceedings of the 22nd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1375–1381. IEEE, June 2009.

[59] Lixin Duan, Dong Xu, Ivor W. Tsang, and Jiebo Luo. Visual event recognition in videos by learning from web data. In *Proceedings of the 23rd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010.

[60] Eric Eaton, Marie desJardins, and Terran Lane. Modeling transfer relationships between learning tasks for improved inductive transfer. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 317–332. Springer, September 2008.

[61] Henry C. Ellis. *The Transfer of Learning*. The Macmillan Company, New York, 1965.

[62] Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[63] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi-task learning. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117. ACM, August 2004.

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

[65] Brian Ferris, Dieter Fox, and Neil Lawrence. WiFi-SLAM using gaussian process latent variable models. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2480–2485, January 2007.

[66] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37. Springer-Verlag, 1995.

[67] Jing Gao, Wei Fan, Jing Jiang, and Jiawei Han. Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 283–291. ACM, August 2008.

[68] Wei Gao, Peng Cai, Kam-Fai Wong, and Aoying Zhou. Learning to rank only using training data from related domain. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 162–169. ACM, July 2010.

[69] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel method for the two-sample problem. In *Advances in Neural Information Processing Systems 19*, pages 513–520. MIT Press, 2007.

[70] Arthur Gretton, Olivier Bousquet, Alex J. Smola, and Bernhard Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *Proceedings of the 16th International Conference on Algorithmic Learning Theory*, volume 3734 of *Lecture Notes in Computer Science*, pages 63–77. Springer, October 2005.

[71] Hong Lei Guo, Li Zhang, and Zhong Su. Empirical study on the performance stability of named entity recognition model across domains. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 509–516. ACL, July 2006.

[72] Rakesh Gupta and Lev Ratinov. Text categorization with knowledge transfer from heterogeneous data sources. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 842–847. AAAI Press, July 2008.

[73] Sunil Kumar Gupta, Dinh Phung, Brett Adams, Truyen Tran, and Svetha Venkatesh. Nonnegative shared subspace learning and its application to social media retrieval. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1169–1178. ACM, July 2010.

[74] Andreas Haeberlen, Eliot Flannery, Andrew M. Ladd, Algis Rudys, Dan S. Wallach, and Lydia E. Kavraki. Practical robust localization over large-scale 802.11 wireless networks. In *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, pages 70–84, September 2004.

[75] Abhay Harpale and Yiming Yang. Active learning for multi-task adaptive filtering. In *Proceedings of the 27th International Conference on Machine learning*. ACM, June 2010.

[76] John A. Hartigan and M. Anthony Wong. A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[77] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition, 2009.

[78] Shohei Hido, Yuta Tsuboi, Hisashi Kashima, Masashi Sugiyama, and Takafumi Kanamori. Inlier-based outlier detection via direct density ratio estimation. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 223–232, December 2008.

[79] Jean Honorio and Dimitris Samaras. Multi-task learning of gaussian graphical models. In *Proceedings of the 27th International Conference on Machine learning*, Haifa, Israel, June 2010. ACM.

[80] Derek H. Hu, Sinno Jialin Pan, Vincent W. Zheng, Nathan N. Liu, and Qiang Yang. Real world activity recognition with multiple goals. In *Proceedings of 10th ACM International Conference on Ubiquitous Computing*, pages 30–39. ACM, September 2008.

[81] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, August 2004.

[82] Jiayuan Huang, Alex Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems 19*, pages 601–608. MIT Press, 2007.

[83] Laurent Jacob, Francis Bach, and Jean-Philippe Vert. Clustered multi-task learning: A convex formulation. In *Advances in Neural Information Processing Systems 21*, pages 745–752. 2009.

[84] Tony Jebara. Multi-task feature and kernel selection for SVMs. In *Proceedings of the 21st International Conference on Machine Learning*, volume 69. ACM, July 2004.

[85] Jing Jiang. *Domain Adaptation in Natural Language Processing*. PhD thesis, The University of Illinois at Urbana-Champaign, 2008.

[86] Jing Jiang. Multi-task transfer learning for weakly-supervised relation extraction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th*

International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, volume 2, pages 1012–1020. ACL, 2009.

[87] Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271. ACL, June 2007.

[88] Nitin Jindal and Bing Liu. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, pages 219–230. ACM, April 2008.

[89] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, Lecture Notes in Computer Science, pages 137–142. Springer, April 1998.

[90] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209. Morgan Kaufmann Pulishers Inc., June 1999.

[91] Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. A least-squares approach to direct importance estimation. *Journal of Machine Learning Research*, 10:1391–1445, 2009.

[92] Yoshinobu Kawahara and Masashi Sugiyama. Change-point detection in time-series data by direct density-ratio estimation. In *Proceedings of 2009 SIAM International Conference on Data Mining*, pages 389–400, April 2009.

[93] Lun-Wei Ku, Yu-Ting Liang, and Hsin-Hsi Chen. Opinion extraction, summarization and tracking in news and blog corpora. In *Proceedings of the AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*, 2006.

[94] Christoph H. Lampert and Oliver Krömer. Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning. In *Proceedings of the 11th European Conference on Computer Vision*, September 2010.

[95] Gert R. G. Lanckriet, Nello Cristianini, Peter L. Bartlett, Laurent El Ghaoui, and Michael I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[96] Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339, 1995.

[97] Neil D. Lawrence and John C. Platt. Learning to learn with the informative vector machine. In *Proceedings of the 21st International Conference on Machine Learning*, Banff, Alberta, Canada, July 2004. ACM.

[98] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *Annual Conference on Neural Information Processing Systems 19*, pages 801–808. MIT Press, 2007.

[99] Su-In Lee, Vassil Chatalbashev, David Vickrey, and Daphne Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th International Conference on Machine Learning*, pages 489–496. ACM, July 2007.

[100] Julie Letchner, Dieter Fox, and Anthony LaMarca. Large-scale localization from wireless signal strength. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 15–20. AAAI Press / The MIT Press, July 2005.

[101] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. ACM / Springer, July 1994.

[102] Bin Li, Qiang Yang, and Xiangyang Xue. Can movies and books collaborate?: cross-domain collaborative filtering for sparsity reduction. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pages 2052–2057. Morgan Kaufmann Publishers Inc., July 2009.

[103] Bin Li, Qiang Yang, and Xiangyang Xue. Transfer learning for collaborative filtering via a rating-matrix generative model. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 617–624. ACM, June 2009.

[104] Tao Li, Vikas Sindhwani, Chris Ding, and Yi Zhang. Knowledge transformation for cross-domain sentiment classification. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 716–717. ACM, July 2009.

[105] Tao Li, Yi Zhang, and Vikas Sindhwani. A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association of Computational Linguistics*, pages 244–252. ACL, August 2009.

[106] Lin Liao, Donald J. Patterson, Dieter Fox, and Henry A. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311–331, 2007.

[107] Xuejun Liao, Ya Xue, and Lawrence Carin. Logistic regression with an auxiliary data source. In *Proceedings of the 21st International Conference on Machine Learning*, pages 505–512. ACM, August 2005.

[108] Xiao Ling, Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. Spectral domain-transfer learning. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 488–496. ACM, August 2008.

[109] Xiao Ling, Gui-Rong Xue, Wenyuan Dai, Yun Jiang, Qiang Yang, and Yong Yu. Can chinese web pages be classified with english data source? In *Proceedings of the 17th International Conference on World Wide Web*, pages 969–978. ACM, April 2008.

[110] Bing Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Springer, January 2007.

[111] Bing Liu. Sentiment analysis and subjectivity. *Handbook of Natural Language Processing, Second Edition*, 2010.

[112] Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient $l_{2,1}$-norm minimization. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, June 2009.

[113] Kang Liu and Jun Zhao. Cross-domain sentiment classification using a two-stage method. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1717–1720. ACM, November 2009.

[114] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. Modeling and predicting the helpfulness of online reviews. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 443–452. IEEE Computer Society, December 2008.

[115] Yiming Liu, Dong Xu, Ivor W. Tsang, and Jiebo Luo. Using large-scale web data to facilitate textual query based retrieval of consumer photos. In *Proceedings of the seventeen ACM international conference on Multimedia*, pages 55–64. ACM, October 2009.

[116] Yue Lu, Panayiotis Tsaparas, Alexandros Ntoulas, and Livia Polanyi. Exploiting social context for review quality prediction. In *Proceedings of the 19th international conference on Wrld wide web*, pages 691–700. ACM, April 2010.

[117] Yue Lu and Chengxiang Zhai. Opinion integration through semi-supervised topic modeling. In *Proceedings of the 17th International Conference on World Wide Web*, pages 121–130. ACM, April 2008.

[118] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. Rated aspect summarization of short comments. In *Proceedings of the 18th international conference on World wide web*, 2009.

[119] Ping Luo, Fuzhen Zhuang, Hui Xiong, Yuhong Xiong, and Qing He. Transfer learning from multiple source domains via consensus regularization. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 103–112. ACM, October 2008.

[120] M. M. Hassan Mahmud and Sylvian R. Ray. Transfer learning using kolmogorov complexity: Basic theory and empirical evaluations. In *Annual Conference on Neural Information Processing Systems 20*, pages 985–992. MIT Press, 2008.

[121] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009.

[122] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems 21*, pages 1041–1048. 2009.

[123] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Multiple source adaptation and the renyi divergence. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, June 2009.

[124] Lilyana Mihalkova, Tuyen Huynh, and Raymond J. Mooney. Mapping and revising markov logic networks for transfer learning. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 608–614. AAAI Press, July 2007.

[125] Lilyana Mihalkova and Raymond J. Mooney. Transfer learning from minimal target data by mapping across relational domains. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pages 1163–1168. Morgan Kaufmann Publishers Inc., July 2009.

[126] Sebastian Mika, Gunnar Rätsch, Jason Weston, Bernhard Schölkopf, and Klaus-Robert Müller. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX*, pages 41–48. 1999.

[127] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[128] Tony Mullen and Nigel Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 412–418, July 2004.

[129] Yurii Nesterov and Arkadii Nemirovskii. *Interior-point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1994.

[130] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, 2001.

[131] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2-3):103–134, 2000.

[132] Jeffrey J. Pan, James T. Kwok, Qiang Yang, and Yiqiang Chen. Multidimensional vector regression for accurate and low-cost location estimation in pervasive computing. *IEEE Transactions on Knowledge and Data Engineering*, 18(9):1181–1193, September 2006.

[133] Jeffrey J. Pan and Qiang Yang. Co-localization from labeled and unlabeled data using graph laplacian. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2166–2171, January 2007.

[134] Jeffrey J. Pan, Qiang Yang, Hong Chang, and Dit-Yan Yeung. A manifold regularization approach to calibration reduction for sensor-network based tracking. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 988–993. AAAI Press, July 2006.

[135] Sinno Jialin Pan, James T. Kwok, and Qiang Yang. Transfer learning via dimensionality reduction. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pages 677–682. AAAI Press, July 2008.

[136] Sinno Jialin Pan, James T. Kwok, Qiang Yang, and Jeffrey J. Pan. Adaptive localization in a dynamic WiFi environment through multi-view learning. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 1108–1113. AAAI Press, July 2007.

[137] Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Chen Zheng. Cross-domain sentiment classification via spectral feauture alignment. In *Proceedings of the 19th International Conference on World Wide Web*, pages 751–760. ACM, April 2010.

[138] Sinno Jialin Pan, Dou Shen, Qiang Yang, and James T. Kwok. Transferring localization models across space. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pages 1383–1388. AAAI Press, July 2008.

[139] Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1187–1192, July 2009.

[140] Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 2010. Submitted.

[141] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.

[142] Sinno Jialin Pan, Vincent W. Zheng, Qiang Yang, and Derek H. Hu. Transfer learning for WiFi-based indoor localization. In *Proceedings of the Workshop on Transfer Learning for Complex Task of the 23rd AAAI Conference on Artificial Intelligence*, July 2008.

[143] Weike Pan, Evan W. Xiang, Nathan N. Liu, and Qiang Yang. Transfer learning in collaborative filtering for sparsity reduction. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*. AAAI Press, July 2010.

[144] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association of Computational Linguistics*, pages 271–278. ACL, July 2004.

[145] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[146] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86. ACL, July 2002.

[147] David Pardoe and Peter Stone. Boosting for regression transfer. In *Proceedings of the 27th International Conference on Machine learning*. ACM, June 2010.

[148] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. MIT Press, 2009.

[149] Piyush Rai and Hal DauméIII. Infinite predictor subspace models for multitask learning. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, July 2010.

[150] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning*, pages 759–766. ACM, June 2007.

[151] Rajat Raina, Andrew Y. Ng, and Daphne Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 713–720. ACM, June 2006.

[152] Sowmya Ramachandran and Raymond J. Mooney. Theory refinement of bayesian networks with hidden variables. In *Proceedings of the 14th International Conference on Machine Learning*, pages 454–462, July 1998.

[153] Parisa Rashidi and Diane J. Cook. Activity recognition based on home to home transfer learning. In *Proceedings of the Workshop on Plan, Activity, and Intent Recognition of the 24th AAAI Conference on Artificial Intelligence*. AAAI Press, July 2010.

[154] Vikas C. Raykar, Balaji Krishnapuram, Jinbo Bi, Murat Dundar, and R. Bharat Rao. Bayesian multiple instance learning: automatic feature selection and inductive transfer. In *Proceedings of the 25th International Conference on Machine learning*, pages 808–815. ACM, July 2008.

[155] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning Journal*, 62(1-2):107–136, 2006.

[156] Alexander E. Richman and Patrick Schone. Mining Wiki resources for multilingual named entity recognition. In *Proceedings of 46th Annual Meeting of the Association of Computational Linguistics: Human Language Technology*, pages 1–9. ACL, June 2008.

[157] Stephen Robertson, Stephen Robertson, and Ian Soboroff. The trec 2002 filtering track report. In *Text REtrieval Conference 2001*, 2001.

[158] Marcus Rohrbach, Michael Stark, György Szarvas, Iryna Gurevych, and Bernt Schiele. What helps where - and why? semantic relatedness for knowledge transfer. In *Proceedings of the 23rd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010.

[159] Michael T. Rosenstein, Zvika Marx, and Leslie Pack Kaelbling. To transfer or not to transfer. In *a NIPS-05 Workshop on Inductive Transfer: 10 Years Later*, December 2005.

[160] Ulrich Rückert and Stefan Kramer. Kernel-based inductive transfer. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 220–233. Springer, September 2008.

[161] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Proceedings of the 11th European Conference on Computer Vision*, September 2010.

[162] B. Schölkopf, S. Mika, C.J.C. Burges, P. Knirsch, Klaus-Robert Müller, G. Rätsch, and A.J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, September 1999.

[163] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[164] Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[165] Anton Schwaighofer, Volker Tresp, and Kai Yu. Learning gaussian process kernels via hierarchical bayes. In *Annual Conference on Neural Information Processing Systems 17*, pages 1209–1216. MIT Press, 2005.

[166] Gabriele Schweikert, Christian Widmer, Bernhard Schölkopf, and Gunnar Rätsch. An empirical analysis of domain adaptation algorithms for genomic sequence analysis. In *Advances in Neural Information Processing Systems 21*, pages 1433–1440. 2009.

[167] Chun-Wei Seah, Yew-Soon Ong Ivor W. Tsang, and Kee-Khoon Lee. Predictive distribution matching svm for multi-domain learning. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases 2010*, September 2010.

[168] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[169] Xiaoxiao Shi, Wei Fan, and Jiangtao Ren. Actively transfer domain knowledge. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 342–357. Springer, September 2008.

[170] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90:227–244, 2000.

[171] Vikas Sindhwani and Prem Melville. Document-word co-regularization for semi-supervised sentiment analysis. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 1025–1030. IEEE Computer Society, 2008.

[172] Alex J. Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A Hilbert space embedding for distributions. In *Proceedings of the 18th International Conference on Algorithmic Learning Theory*, pages 13–31, October 2007.

[173] Le Song. *Learning via Hilbert Space Embedding of Distributions*. PhD thesis, The University of Sydney, December 2007.

[174] Le Song, Alex Smola, Karsten Borgwardt, and Arthur Gretton. Colored maximum variance unfolding. In *Annual Conference on Neural Information Processing Systems 20*, pages 1385–1392. MIT Press, 2008.

[175] Danny C. Sorensen. Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations. Technical Report TR-96-40, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1996.

[176] Bharath Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Gert Lanckriet, and Bernhard Schölkopf. Kernel choice and classifiability for RKHS embeddings of probability distributions. In *Advances in Neural Information Processing Systems 22*, pages 1750–1758. 2009.

[177] Michael Stark, Michael Goesele, and Bernt Schiele. A shape-based object class model for knowledge transfer. In *Proceedings of 12th IEEE International Conference on Computer Vision*, September 2009.

[178] Ingo Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2001.

[179] Masashi Sugiyama, Motoaki Kawanabe, and Pui Ling Chui. Dimensionality reduction for density ratio estimation in high-dimensional spaces. *Neural Network*, 23(1):44–59, 2010.

[180] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul Von Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in Neural Information Processing Systems 20*, pages 1433–1440. MIT Press, 2008.

[181] Taiji Suzuki and Masashi Sugiyama. Estimating squared-loss mutual information for independent component analysis. In *Proceedings of the 8th International Conference on Independent Component Analysis and Signal Separation*, pages 130–137, March 2009.

[182] Matthew E. Taylor and Peter Stone. *Journal of Machine Learning Research*, 10(1):1633–1685, 2009.

[183] Sebastian Thrun and Lorien Pratt, editors. *Learning to learn*. Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[184] Ivan Titov and Ryan McDonald. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of 46th Annual Meeting of the Association of Computational Linguistics: Human Language Technology*, pages 308–316. ACL, June 2008.

[185] Tatiana Tommasi, Francesco Orabona, and Barbara Caputo. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *Proceedings of the 23rd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010.

[186] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[187] Peter Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting fo the Association for Computational Linguistics*, pages 417–424, 2002.

[188] Laurens van der Maaten, Eric Postma, and Jaap van den Herik. Dimensionality reduction: A comparative review. Technical Report TiCC-TR 2009-005, Tilburg University, 2009.

[189] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, September 1998.

[190] Alexander Vezhnevets and Joachim Buhmann. Towards weakly supervised semantic segmentation by means of multiple instance and multitask learning. In *Proceedings of the 23rd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010.

[191] Paul von Bünau, Frank C. Meinecke, Franz C. Kirúly, and Klaus R. Müller. Finding stationary subspaces in multivariate time series. *Physical Review Letters*, 103(21), September 2009.

[192] Bo Wang, Jie Tang, Wei Fan, Songcan Chen, Zi Yang, and Yanzhu Liu. Heterogeneous cross domain ranking in latent space. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 987–996. ACM, November 2009.

[193] Chang Wang and Sridhar Mahadevan. Manifold alignment using procrustes analysis. In *Proceedings of the 25th International Conference on Machine learning*, pages 1120–1127. ACM, July 2008.

[194] Hua-Yan Wang, Vincent W. Zheng, Junhui Zhao, and Qiang Yang. Indoor localization in multi-floor environments with reduced effort. In *Proceedings of the 8th Annual IEEE International Conference on Pervasive Computing and Communications*, pages 244–252. IEEE Computer Society, March 2010.

[195] Pu Wang, Carlotta Domeniconi, and Jian Hu. Using Wikipedia for co-clustering based cross-domain text classification. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 1085–1090. IEEE Computer Society, 2008.

[196] Xiaogang Wang, Cha Zhang, and Zhengyou Zhang. Boosted multi-task learning for face verification with applications to web image and video search. In *Proceedings of the 22nd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 142–149. IEEE, June 2009.

[197] Zheng Wang, Yangqiu Song, and Changshui Zhang. Transferred dimensionality reduction. In *Proceedings of the 2008 European Conference on European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 550–565. Springer, September 2008.

[198] Kilian Q. Weinberger, Fei Sha, and Lawrence K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the 21st International Conference on Machine Learning*, July 2004.

[199] Christian Widmer, Jose Leiva, Yasemin Altun, and Gunnar Rätsch. Leveraging sequence classification by taxonomy-based multitask learning. In *Proceedings of 14th Annual International Conference on Research in Computational Molecular Biology*, volume 6044 of *Lecture Notes in Computer Science*, pages 522–534. Springer, April 2010.

[200] Tak-Lam Wong, Wai Lam, and Bo Chen. Mining employment market via text block detection and adaptive cross-domain information extraction. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 283–290. ACM, 2009.

[201] Dan Wu, Wee Sun Lee, Nan Ye, and Hai Leong Chieu. Domain adaptive bootstrapping for named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1523–1532. ACL, August 2009.

[202] Pengcheng Wu and Thomas G. Dietterich. Improving svm accuracy by training on auxiliary data sources. In *Proceedings of the 21st International Conference on Machine Learning*, Banff, Alberta, Canada, July 2004. ACM.

[203] Evan W. Xiang, Bin Cao, Derek H. Hu, and Qiang Yang. Bridging domains using world wide knowledge for transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:770–783, June 2010.

[204] Sihong Xie, Wei Fan, Jing Peng, Olivier Verscheure, and Jiangtao Ren. Latent space domain transfer between high dimensional overlapping distributions. In *18th International World Wide Web Conference*, pages 91–100. ACM, April 2009.

[205] Dikan Xing, Wenyuan Dai, Gui-Rong Xue, and Yong Yu. Bridged refinement for transfer learning. In *11th European Conference on Principles and Practice of Knowledge*

Discovery in Databases, Lecture Notes in Computer Science, pages 324–335. Springer, September 2007.

[206] Qian Xu, Sinno Jialin Pan, Hannah Hong Xue, and Qiang Yang. Multitask learning for protein subcellular location prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 99(PrePrints), 2010.

[207] Gui-Rong Xue, Wenyuan Dai, Qiang Yang, and Yong Yu. Topic-bridged plsa for cross-domain text classification. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 627–634. ACM, July 2008.

[208] Rong Yan and Jian Zhang. Transfer learning using task-level features with application to information retrieval. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pages 1315–1320. Morgan Kaufmann Publishers Inc., 2009.

[209] Jun Yang, Rong Yan, and Alexander G. Hauptmann. Cross-domain video concept detection using adaptive svms. In *Proceedings of the 15th international conference on Multimedia*, pages 188–197. ACM, 2007.

[210] Qiang Yang. Activity recognition: Linking low-level sensors to high-level intelligence. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 20–25. Morgan Kaufmann Publishers Inc., July 2009.

[211] Qiang Yang, Yuqiang Chen, Gui-Rong Xue, Wenyuan Dai, and Yong Yu. Heterogeneous transfer learning for image clustering via the social web. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, volume 1, pages 1–9. ACL, 2009.

[212] Qiang Yang, Sinno Jialin Pan, and Vincent W. Zheng. Estimating location using Wi-Fi. *IEEE Intelligent Systems*, 23(1):8–13, 2008.

[213] Tianbao Yang, Rong Jin, Anil K. Jain, Yang Zhou, and Wei Tong. Unsupervised transfer classification: application to text categorization. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1159–1168. ACM, July 2010.

[214] Xiaolin Yang, Seyoung Kim, and Eric Xing. Heterogeneous multitask learning with joint sparsity constraints. In *Advances in Neural Information Processing Systems 22*, pages 2151–2159. 2009.

[215] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420. Morgan Kaufmann Publishers Inc., July 1997.

[216] Jieping Ye. Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems. *Journal of Machine Learning Research*, 6:483–502, 2005.

[217] Jie Yin, Qiang Yang, and L.M. Ni. Learning adaptive temporal radio maps for signal-strength-based location estimation. *IEEE Transactions on Mobile Computing*, 7(7):869–883, July 2008.

[218] Xiao-Tong Yuan and Shuicheng Yan. Visual classification with multi-task joint sparse representation. In *Proceedings of the 23rd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010.

[219] Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the 21st International Conference on Machine Learning*, July 2004.

[220] Hongyuan Zha, Xiaofeng He, Chris Ding, Horst Simon, and Ming Gu. Spectral relaxation for k-means clustering. In *Advances in Neural Information Processing Systems 14*, pages 1057–1064. MIT Press, 2001.

[221] Zheng-Jun Zha, Tao Mei, Meng Wang, Zengfu Wang, and Xian-Sheng Hua. Robust distance metric learning with auxiliary knowledge. In *Proceedings of the 21st international jont conference on Artifical intelligence*, pages 1327–1332, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[222] Kai Zhang, Joe W. Gray, and Bahram Parvin. Sparse multitask regression for identifying common mechanism of response to therapeutic targets. *Bioinformatics*, 26(12):i97–i105, 2010.

[223] Yu Zhang, Bin Cao, and Dit-Yan Yeung. Multi-domain collaborative filtering. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, July 2010.

[224] Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, July 2010.

[225] Yu Zhang and Dit-Yan Yeung. Multi-task warped gaussian process for personalized age estimation. In *Proceedings of the 23rd IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010.

[226] Yu Zhang and Dit-Yan Yeung. Transfer metric learning by learning task relationships. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1199–1208. ACM, July 2010.

[227] Peilin Zhao and Steven C.H. Hoi. OTL: A framework of online transfer learning. In *Proceedings of the 27th International Conference on Machine learning*. ACM, June 2010.

[228] Vincent W. Zheng, Derek H. Hu, and Qiang Yang. Cross-domain activity recognition. In *Proceedings of the 11th International Conference on Ubiquitous Computing*, September 2009.

[229] Vincent W. Zheng, Sinno Jialin Pan, Qiang Yang, and Jeffrey J. Pan. Transferring multi-device localization models using latent multi-task learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pages 1427–1432. AAAI Press, July 2008.

[230] Vincent W. Zheng, Qiang Yang, Evan W. Xiang, and Dou Shen. Transferring localization models over time. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pages 1421–1426. AAAI Press, July 2008.

[231] Erheng Zhong, Wei Fan, Jing Peng, Kun Zhang, Jiangtao Ren, Deepak Turaga, and Olivier Verscheure. Cross domain distribution adaptation via kernel mapping. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1027–1036. ACM, June 2009.

[232] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*, pages 321–328. MIT Press, 2004.

[233] Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[234] Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceeding of The 22nd International Conference on Machine Learning*, pages 912–919, August 2003.

[235] Hankui Zhuo, Qiang Yang, Derek H. Hu, and Lei Li. Transferring knowledge from another domain for learning action models. In *Proceedings of 10th Pacific Rim International Conference on Artificial Intelligence*, pages 1110–1115. Springer-Verlag, December 2008.