Graph and Subspace Learning for Domain Adaptation

A Dissertation
Submitted to
the Temple University Graduate Board

---

in Partial Fulfillment
of the Requirements for the Degree of
DOCTOR OF PHILOSOPHY

---

by
Le Shu
October, 2015

Examining Committee Members:

Longin Jan Latecki, Advisory Chair, Computer and Information Sciences
Haibin Ling, Computer and Information Sciences
Slobodan Vucetic, Computer and Information Sciences
Ying Zhu, External Member, Electrical and Computer Engineering

Abstract

Graph and Subspace Learning for Domain Adaptation

by

Le Shu

Doctor of Philosophy in Computer and Information Sciences

Temple University in Philadelphia, October, 2015

Researcher Advisor: Longin Jan Latecki

In many practical problems, given that the instances in the training and test may be drawn from different distributions, traditional supervised learning can not achieve good performance on the new domain. Domain adaptation algorithms are therefore designed to bridge the distribution gap between training (source) data and test (target) data. In this thesis, I propose two graph learning and two subspace learning methods for domain adaptation.

Graph learning methods use a graph to model pairwise relations between instances and then minimize the domain discrepancy based on the graphs directly. The first effort we make is to propose a novel locality preserving projection method for domain adaptation task, which can find a linear mapping preserving the intrinsic structure for both source and target domains. We first construct two graphs encoding the neighborhood information for source and target domains separately. We then find linear projection coefficients which have the property of locality preserving for each graph. Instead of combing the two objective terms under compatibility assumption and requiring the user to decide the importance of each objective function, we propose a multi-objective formulation for this problem and solve it simultaneously using Pareto optimization. Pareto optimization allows multiple objectives to compete with each other in deciding the optimal trade-off. We use generalized eigen-decomposition to find the pareto frontier, which captures all possible good linear projection coefficients that are preferred by one or more objectives. The second effort is to

directly improve the pair-wise similarities between instances in the same domain as well as in different domains. We propose a novel method to solve domain adaptation task in a transductive setting. The proposed method bridges the distribution gap between source domain and target domain through affinity learning. It exploits the existence of a subset of data points in target domain which distribute similarly to the data points in the source domain. These data points act as the bridge that facilitates the data similarities propagation across domains. We also propose to control the relative importance of intra- and inter-domain similarities to boost the similarity propagation. In our approach, we first construct the similarity matrix which encodes both the intra- and inter- domain similarities. We then learn the true similarities among data points in joint manifold using graph diffusion. We demonstrate that with improved similarities between source and target data, spectral embedding provides a better data representation, which boosts the prediction accuracy.

Subspace learning methods aim to find a new coordinate system, in which the domain discrepancy is minimized. In this thesis, we refer to subspace-based method as those which model the domain shift between two subspaces directly. Our first effort is to propose a novel linear subspace learning approach for domain adaptation. Our key observation is that in many real world problems, such as image classification with blurred test images or cross domain text classification, domain shift can be modeled by a linear transformation between the source and target domain (intrinsically linear transformation between two subspaces underlying the source and target data). Motivated by this observation, our method explicitly aligns the data in two domains using a linear transformation while simultaneously finding a subspace which preserves the most data variance. With explicit data alignment, the subspace learning is formulated as minimizing of a PCA-like objective, which consists of two variables: the basis vectors of the common subspace and the linear transformation between two domains. We show that the optimization can be solved efficiently using an iterative algorithm based on alternating minimization, and prove its convergence to a local optimum. Our method can also integrate the label information of source data, which fur-

ther improves the robustness of the subspace learning and yields better prediction. Existing subspace based domain adaptation methods assume that data lie in a single low dimensional subspace. This assumption is too strong in many real world applications especially considering the domain could be a mixture of latent domains with significant inner-domain variations that should not be neglected. In our second approach, the key idea is to assume the data lie in a union of multiple low dimensional subspaces, which relaxes the common assumption above. We propose a novel two step subspace based domain adaptation algorithm: in subspaces discovery step, we cluster the source and target data using subspace clustering algorithm and estimate the subspace for each cluster using principal component analysis; in domain adaptation step, we propose a novel multiple subspace alignment (Multi-SA) algorithm, in which we identify one common subspace that aligns well with both source and target subspaces, and therefore, best preserves the variance for both domains. To solve this alignment problem jointly for multiple subspaces, we formulate this problem as solving an optimization problem that minimizes the weighted sum of multiple alignment costs. A higher weight is assigned to a source subspace if its label distribution has smaller distance, measured by KL divergence, compared to the overall label distribution. By putting more weights on those subspaces, the learned common subspace is able to to preserve the distinctive information.

Acknowledgements

It has been a wonderful journey for me, when i look over the past four years studying at Temple University. It is also of great joy reminding me all the friends and family who have helped and supported me along this long but fulfilling road.

My utmost gratitude goes to my advisor, Professor Longin Jan Latecki. I have learned so much from him over the past five years. His insights and suggestions help me to improve my research skills. His enthusiasm for research and encouragement have kept me continuing my work. Without the guidance and support of him, none of the work described in this dissertation would have been possible.

I would like to express my heartfelt gratitude to Professor Haibin Ling and Professor Slobodan Vucetic. They kindly served as my committee members and provided me with very useful comments. I would also like to thank my external committee member Professor Ying Zhu.

I would like to thank all group members, Xingwei Yang, Nan Li, Meng Yi, Zhuo Deng, Chen Shen, David Dobor, Ren-Hau Howard Liu, Xinggang Wang and Cong Yao for all discussions and suggestions. I will miss the time we spent together working toward the same deadline.

I would like to thank all my friends at Temple University, Erkang Cheng, Liya Ma, Mian Wang, Nian Shi, Liang Du, Chengliang Wang, Yi Wu, Yunsheng Wang, Yu Pang, Pengpeng Liang, who have made the past five years some of the most memorable years in my life.

I would also like to thank the mentors of my graduate study at Huazhong University of Science and Technology: Dr. Wenyu Liu and Dr. Hongbo Jiang. The study in the lab inspired me to pursue my PhD degree.

I would like to thank my parents for their unconditional love and support. They have done so much for me, and have been always supportive.

With great love, I thank my husband Tianyang Ma for sharing with me all the joys and supporting me through the rough time wholeheartedly.

Le Shu

Temple University

October 2015

To my parents and Tianyang.

Contents

List of Tables

List of Figures

Chapter 1

Introduciton

As human beings, we are able to adapt and apply efficiently our past experience to new scenarios, but how can we reproduce this skill for an artificial learning system?

1.1 Supervised Learning, Unsupervised Learning, Semi-Supervised Learning

The learning algorithms can be roughly divided into three main categories based on the type of information contained in the training data: supervised learning, unsupervised learning and semi-supervised learning. Supervised learning analyze the previously collected labeled training data and produces an inferred function, which can be used for mapping unlabeled test data [11, 19, 76, 64]. Unsupervised learning is to find hidden structure in unlabeled data [43, 57, 63, 73, 85]. Since the data given to the learner are unlabeled, there is no error or reward signal that can be used to evaluate a potential solution in unsupervised learning. Semi-supervised learning is halfway between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data) [14, 98, 4, 99]. Semi-supervised learning make use of a large amount of unlabeled data in conjunction with a small amount of labeled data, which can produce considerable improvement in learning accuracy.

The success of machine learning method usually rely on the existence of large amount of labeled data. It is unrealistic to assume the availability of labels with the increasing amount of data from various sources. it is also unrealistic and expensive to do manual annotation. Unsupervised learning and semi-supervised learning can solve the problem in a certain extent. However, unsupervised learning and semi-supervised learning still have the assumption: the training data, labeled or unlabeled, have the same distribution as that of the test data. Unfortunately, not all of these data, in fact, a majority part of these data does not necessarily follow the same distribution as the test data.

In this thesis, we focus on utilizing data comes from a different but closely related distribution as that of the test data, to aid learning the model. We are concerning two learning scenarios which are different from supervised learning, unsupervised learning and semi-supervised learning: 1) domain adaptation 2) learning from multiple domains or multiple latent domains. In Chapter 2, 3, 4, we are going to cover the first scenario with several strategies, locality preserving projection, affinity learning and data alignment. In Chapter 5, we are going to elaborate our strategy to cover the second scenario, subspace clustering with multiple subspace alignment.

1.2 Domain Adaptation

1.2.1 Single Source Domain Adaptation

Domain adaptation (DA) is a research field associated with machine learning and transfer learning[52, 75, 51, 79]. It aims to learn a well performing model from a source data distributions on a different but closely related target data distribution. Domain adaptation has gained significant attention in many areas of applied machine learning, including bio-informatics, speech and language processing, computer vision and etc.

For example, considering the problem of sentiment classification [9], where the task is to automatically classify the reviews on a product, such as newly released video games, into positive and negative views. For this classification task, we need to first collect many reviews of the product and manually annotate them. We would then train a classifier on the reviews with their corresponding labels. To achieve good classification accuracy, we need to manually annotate a large amount of reviews since the distribution of reviews on different video games can be very different. However, this manual annotation process can be very expensive to do. To reduce the effort for annotating reviews for video games, we may want to adapt a classification model that is trained on existing reviews on electronics to help learn classification models for reviews on the video games. In such cases, domain

adaptation can save a significant amount of labeling effort. We demonstrate sentimental classification for domain adaptation in Figure.(1.1).

As a second example, consider the problem of visual object recognition in computer vision. The goal is to recognize visual objects in each images. the labeled examples may be images downloaded from online merchants that are associated with category information obtained through previous manual-labeling efforts. For a classification task on the newly high-resolution images by a digital SLR camera 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 classifiers learned on the images downloaded from Amazon to images obtain with digital SLR camera. In such cases, it would be helpful if we could transfer the classification knowledge into the new domain. We demonstrate visual object recognition for domain adaptation in Figure. (1.2).

In these practical problems, given that the instances in the training and test domains may be drawn from different distributions, traditional supervised learning can not achieve good performance on the new domain. Given a new domain of interest, there may not be sufficient labeled data, and labeled data from a related domain need to be utilized. Domain adaptation algorithms are therefore designed to bridge the distribution gap between training (source) data and test (target) data.

There are several methods that have demonstrated improved performance under domain variations. Given the existence of label information from the source or target domain, these methods can be broadly classified into three groups:

- **Unsupervised Domain Adaptation:** all or part of source examples are labeled.
- **Semi-Supervised Domain Adaptation:** all or part of source examples and a small number of target examples are labeled.
- **Supervised Domain Adaptation:** all the examples in source domain and target domain are labeled.

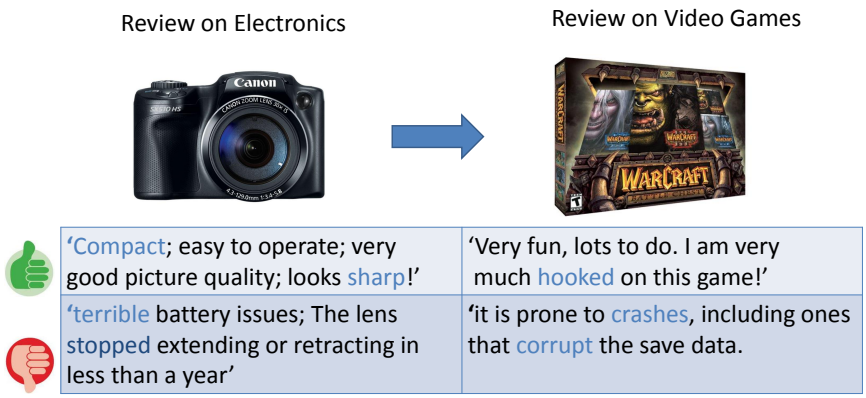

Figure 1.1: Domain Adaptation for Sentiment Classification: thinking about using models trained with reviews on electronics to reviews on video games.

All our approach belongs to unsupervised DA category, where there are no labels for target data. In unsupervised DA group, there are roughly two types of algorithms: The first type of methods can be referred to as instance-based approaches [48, 54, 62, 22, 21, 70, 95, 60, 6, 81, 31, 87, 86], which assumes that certain parts of the data in the source domain can distribute similarly to the data in the target domain by re-weighting. Instance reweighing and importance sampling are two major techniques in this context. These methods can be used to address one variation of domain adaptation problem: covariate shift, where the conditional distribution of the labels are the same for source and target domain but the marginal distribution are not the same. The second type of methods are based on feature representation learning, such as [9, 65, 66, 7, 78, 38, 8, 10, 32, 34, 26, 88, 94, 1]. The assumption is that although source and target data have different distributions, either there exists some general features which have similar conditional distributions in both domains, or it is possible to transform the original feature space into a new feature space which is predictive for the target domain. In this scenario, the knowledge used to transfer across domains is encoded into the newly learned feature representation. The performance on the target data is expected to improve significantly with the new feature representation. Our methods belongs to feature representation learning category.

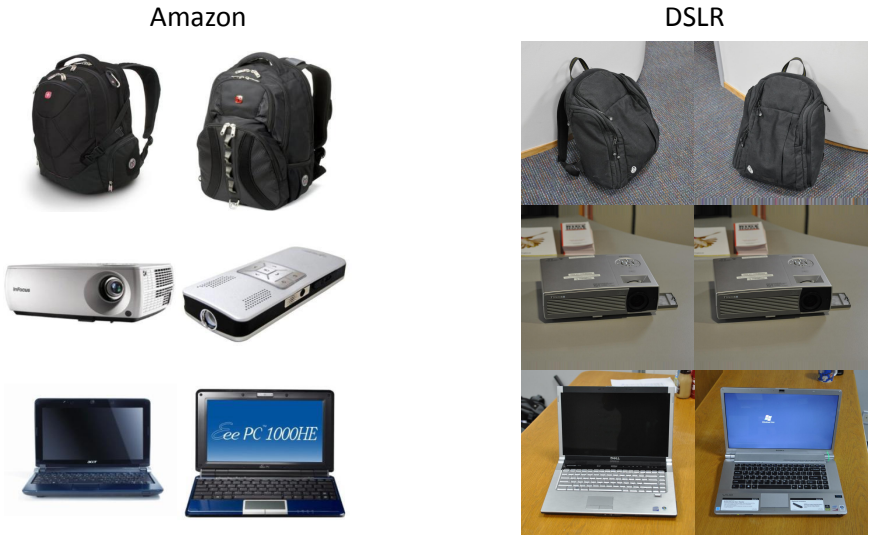

Figure 1.2: Domain Adaptation for Visual Object Recognition: think about using algorithms trained on clean Amazon images to annotate objects acquired with a digital SLR camera. Left: images collected from the amazon website, Right: images collected with a digital SLR camera.

1.2.2 Multiple Source or Multiple Latent Source Domain Adaptation

Domain adaptation with multiple sources has also received a lot of attention in many areas such as natural language processing and computer vision. For example, in sentiment classification, we have plenty of labeled data to train a model in some domains, such as movie reviews and book reviews. However, we may not have enough labeled data available for training in some domains, such as piano reviews. Multiple source domain adaptation algorithms can solve such problems by using the domains which have plenty labeled data as sources, and domains lack of labeled data as target domains. Assumed that we have access to the labeled training data for several source domain, it is wasteful if we only use one source for training. A natural solution is to combine the raw labeled data from each source domain and form a new sample more representative of the target distribution and use that to train a learning algorithm. There are several theoretical methods for multiple source domain adaptation, which give a general theorem which establishes a general bound [20, 62, 5]. There are well-developed algorithms to solve multiple source domain adap-

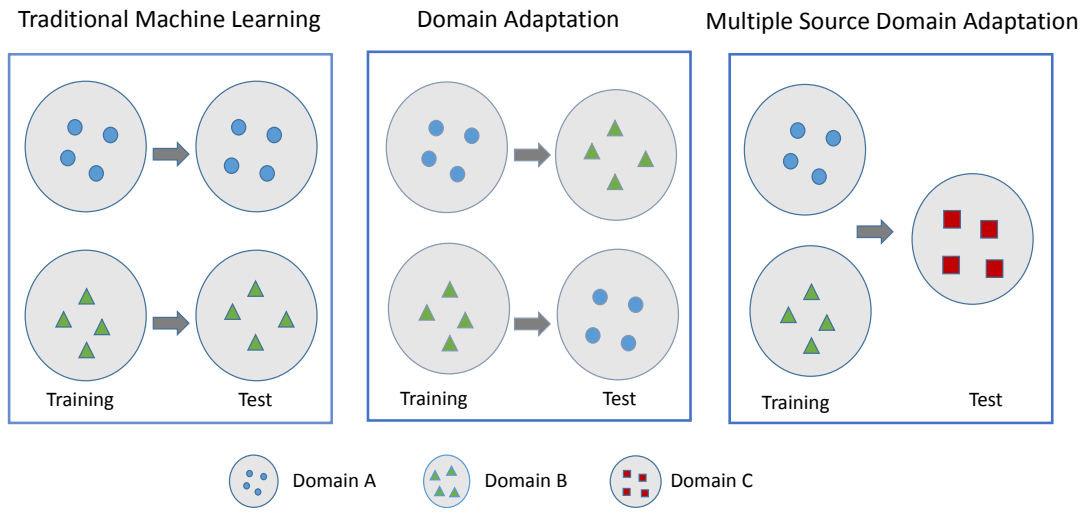

Figure 1.3: A Demonstration of the Difference between Traditional Machine Learning, Domain Adaptation, and Multiple Source Domain Adaptation.

tation problem. There are roughly two types of approaches: one is feature representation approaches [28, 15, 29, 82]; the other is based on the combination of pre-learned classifiers [77, 83, 92, 84].

Recently, [47, 61, 91] and [37] relax this assumption and assume that the domain label of the source data is not available for training. Both methods explicitly address the issue that source data may consist of multiple latent domains. In order to obtain the optimal domain invariant predictor, source data samples need to be first clustered into different groups, each of which corresponds to a latent domain.

1.3 Our Approach

To be specific, we propose graph and subspace learning methods to solve domain adaptation problem. The ultimate goal of all our approaches is to find a new feature representation in which the domain discrepancy is minimized. In graph learning domain adaptation approaches, we generate the new feature representation by directly exploring the properties

of graphs. In subspace learning domain adaptation approaches, we will first generate the subspaces for each domain, and then explore the relationship among these subspaces, and finally generate a domain-invariant new representation.

1.3.1 Graph Learning for Domain Adaptation

Our first approach is based on 'locality preserving' property of laplacian graph. The goal is to find a linear mapping which preserve the intrinsic structure for both source and target domains simultaneously. We proposes a novel locality preserving projection method for domain adaptation task. We first construct two graphs encoding the neighborhood information for source and target domains separately. We then find linear projection coefficients which have the property of locality preserving for each graph. Instead of combing the two objective terms under compatibility assumption and requiring the user to decide the importance of each objective function, we propose a multi-objective formulation for this problem and solve it simultaneously using Pareto optimization. The details of this approach can be find in Chapter 2.

Our second approach is based on the fact that graph diffusion can somehow improve the affinities in a graph. We propose a novel method to solve domain adaptation task in a transductive setting. The proposed method bridges the distribution gap between source domain and target domain through affinity learning. It exploits the existence of a subset of data points in target domain which distribute similarly to the data points in the source domain. These data points act as the bridge that facilitates the data similarities propagation across domains. We also propose to control the relative importance of intra- and inter-domain similarities to boost the similarity propagation. In our approach, we first construct the similarity matrix which encodes both the intra- and inter- domain similarities. We then learn the true similarities among data points in joint manifold using graph diffusion. We demonstrate that with improved similarities between source and target data, spectral

embedding provides a better data representation, which boosts the prediction accuracy. The details of domain adaptation via affinity learning is in Chapter 3.

1.3.2 Subspace Learning for Domain Adaptation

Our first approach propose a novel linear subspace learning methods for domain adaptation. Our key observation is that in many real world problems, such as image classification with blurred test images or cross domain text classification, domain shift can be modeled by a linear transformation between the source data and target data. Motivated by this observation, our method explicitly aligns the data in two domains using a linear transformation while simultaneously finding a subspace which preserves the most data variance. With explicit data alignment, the subspace learning is formulated as minimizing of a PCA-like objective, which consists of two variables: the basis vectors of the common subspace and the linear transformation between two domains. We show that the optimization can be solved efficiently using an iterative algorithm based on alternating minimization, and prove its convergence to a local optimum. The details of the approach is in Chapter 4.

Our second approach is to solve multiple latent domain adaptation problem. We would like to make an argument that single subspace assumption is too strong in many applications, especially considering the domain could be a mixture of latent domains with significant inner-domain variations that should not be neglected or data is from multiple sources. Our key idea is to assume the data lie in a union of multiple low dimensional subspaces, which relaxes the common assumption above. We propose a novel two step subspace based domain adaptation algorithm: in subspaces discovery step, we cluster the source and target data using subspace clustering algorithm and estimate the subspace for each cluster using principal component analysis; in domain adaptation step, we propose a novel multiple subspace alignment algorithm in which we seek a latent common subspace that aligns well to both source and target subspaces. We extensively evaluate our method on various domain adaption tasks for both single source domain and multiple source domains adaptation. Our

approach achieves favorable results compared to state-of-the-art domain adaptation methods. The details of our approach is in Chapter 5.

Chapter 2

Locality Preserving Projection for Domain Adaptation with Multi-Objective Learning

2.1 Introduction

In recent years, domain adaptation has gained significant attention in many areas of applied machine learning, including bio-informatics, speech and language processing, computer vision and etc. In many supervised machine learning and data mining tasks, it is usually assumed that both the labeled and unlabeled data are sampled from the same distribution. However, in many real-world tasks, this assumption does not hold. For example, in temporal domains, the feature distribution may be different from that of the former features over time. In clinical studies of disease, the selected samples may not be representative enough and have selection bias. Given a new domain of interest, there may not be sufficient labeled data, and labeled data from a related domain need to be utilized. In these practical problems, given that the instances in the training and test domains may be drawn from different distributions, traditional supervised learning can not achieve good performance on the new domain. Domain adaptation algorithms are therefore designed to bridge the distribution gap between training (source) data and test (target) data.

Most domain adaptation algorithms seeks to eliminate the difference between source and target distributions. They can be mainly categorized into two classes. The first class of methods seeks to make source distribution close to target distribution by re-weighting (importance sampling) source domain data. Such methods include [48, 54, 62]. The second class of methods are based on feature mapping or feature representation, such as [9, 65, 68]. The assumption is that although source and target data have different distributions, either there exists some general features which have similar conditional distributions in both domains, or it is possible to transform the original feature space into a new feature space which is predictive for the target domain.

In this paper, we propose a novel feature representation transfer method. Given labeled data from source domain and unlabeled data from target domain, locality preserving projections are learned simultaneously on both domains through a multi-objective optimization framework.

There are two key innovations in our method. First, we adopt locality preserving projections, a linear feature transformation method, to solve domain adaptation problem. Locality preserving projections (LPP) are first proposed in [45] as a dimension reduction method. Its key advantage compared to PCA and LDA is that it can discover the "intrinsic dimensionality" of the data, which could be much lower than the original feature space. It builds a graph incorporating neighborhood information of the data set and then computes a transformation which maps the data points to a subspace. The linear transformation optimally preserves local neighborhood information. Compared to other dimension reduction methods, higher classification accuracy can be achieved in the low dimensional space learned by LPP. Because of its good performance and simple implementation, there have been many works using LPP to solve different tasks where promising results are achieved. However, to the best of our knowledge, those methods do not attempt to solve the domain adaptation problem. In our work, in order to solve the domain adaptation problem, a discriminative low dimensional *common* space is discovered using LPP. LPP is learnt simultaneously on source and target domain. This promises that the source label can be transferred to target data in the learnt low dimensional common space.

To simultaneously learn LPP on both domains, we use a multi-objective learning framework, which is our second contribution. We first construct two graphs encoding the neighborhood information of source and target data. Intuitively, LPP needs to preserve local neighborhood information on both source and target data. Therefore, there are two objective functions to be optimized. A standard way to solve the above problem is to combine the two objective terms into a single objective with a trade-off parameter. The trade-off parameter is crucial, and can be obtained using cross-validation. However, in this work, we argue that such paradigm may not be suitable for domain adaptation task, which is simply because the labels of the target data are missing, so it is impossible to perform cross-validation. Therefore, we adopt the multi-objective learning framework. We use the

classic Pareto optimization, which allows multiple objectives to compete with each other in deciding the optimal trade-off. More details are introduced in the methodology section.

The rest of paper is organized as follows: We first review the related work. And then, we describe how to formulate LPP for domain adaptation via multi-objective framework. We further show how to solve the multi-objective optimization by finding the Pareto Frontier via generalized eigendecomposition. After that, experimental results on real world data sets are described in detail. Finally, we draw some conclusions.

2.2 Related Work and Discussion

Domain adaptation have been extensively studied in many research areas [66, 58, 50, 16, 17]. In this paper, we mainly consider the methods which assume that there are no labeled data in target domain (unsupervised domain adaptation). In particular, we review feature representation domain adaptation methods.

[9] proposed a heuristic method for domain adaptation which is called structural correspondence learning (SCL). SCL uses labeled data from both domains to induce the correspondence among features. SCL identify some domain invariant "pivot" features first, the other features are represented using their relative co-occurrence count with all pivot features. After that, SCL computes a projection matrix through the low rank approximation of the matrix. In [55], the main idea is to select features that are generalizable across domains. The method uses a regularized logistic regression classifier. During training, it allows the generalizable features to be less regularized, compared with the domain-specific features. However, their method for finding the generalizable features assumes that there are multiple source domains. Pan et al. [65] attempt to discover a latent feature representation across domains by minimizing the feature distribution difference, which is measured by the Maximum Mean Discrepancy statistic. The method solves a semi-definite programming (SDP) and directly gives the kernel matrix. In [67], an improved version is proposed,

which is called ”transfer component analysis”. The method reduces the distance between domain distributions dramatically by projecting the data onto the learned transfer components. The algorithm learns a kernel function that can be applied on new data sets. Gong et al. proposes geodesic flow kernel (GFK) to solve domain adaptation problems. [10]. The method embeds the source and target data into Grassmann manifolds and constructs geodesic flow between them to model domain shifts. GFK integrates an infinite number of subspaces that lie on the geodesic flow from the source subspace to the target one. and find new feature representations which is robust to changes of domains. In our work, we aim to learn a linear mapping matrix, which can preserve the local neighborhood structure of both source and target data. By learning the locality preserving projections on both source and target data simultaneously, we are able to discover a lower dimensional space which is domain independent.

Locality preserving projections [45] has been applied to solve many machine learning tasks. For example, LPP is adopted in [12] to perform document indexing. In [46], LPP is used to tackle face recognition problem in computer vision. Most recently, [42] proposed a feature selection method which incorporates LPP.

In this paper, we use Pareto optimization to learn LPP simultaneously on source and target domain. In Pareto optimization theory, the Pareto frontier captures all possible good solutions without requiring the users to set the correct parameter. Pareto optimization has not been widely used for the reason that it is NP-hard problem to compute the Pareto frontier in most cases. Recently, [24] show that by imposing orthogonal constraints and with some relaxation, the Pareto frontier of graph cut type objectives can be computed efficiently by solving a generalized eigendecomposition problem. In this paper, we follow the solution proposed in [24]. However, we aim to solve the domain adaptation problem, while [24] aim to tackle the multi-view clustering problem.

2.3 Problem Formulation

We assume that our data originate from two domains, Source (S) and Target (T). Source data is fully labeled, which is $(X_S, \mathbf{y_S}) = \{(x_S^1, y_S^1), (x_S^2, y_S^2), \cdots, (x_S^{n_s}, y_S^{n_s})\}$. Each pair of (x_S^i, y_S^i) lies in $R^d \times y$ space and samples from some distributions $P_S(X, Y)$. The target data has equal dimensionality d as source data, and is sampled from $P_T(X, Y)$. However we do not have any labels for the target domain data, i.e., $(X_T, ?) = \{(x_T^1, ?), (x_T^2, ?), \cdots, (x_T^{n_t}, ?)\}$. Given $(X_S, \mathbf{y_S})$ and $(X_T, ?)$, our goal is to learn linear projection coefficient $\mathbf{w} \in R^d$ such that the learned coefficients are discriminative for both domains.

If we consider $\mathbf{w}$ as coefficients in a linear projection function $y = \mathbf{w}^T x$, which maps data $x \in R^d$ to a continuous value y, then we think discriminative feature weights $\mathbf{w}$ should has the property of locality preserving, i.e., if two data x^i and x^j are "close" then $\mathbf{w}^T x^i$ and $\mathbf{w}^T x^j$ should be as close as well. The same insight has been used in many existing approaches, where locality preserving property shows merits in solving other tasks such as dimension reduction, document indexing and feature selection [45, 12, 44] .

In the rest of this section, we first describe how to construct the adjacency graphs for source and target data respectively. Given the two graphs, we show how to learn coefficients $\mathbf{w}$ simultaneously on both source and target data using a multi-objective optimization framework.

2.3.1 Graph Construction

Let A denote an adjacency graph, where each node represents a data point. We use A_S and A_T to denote the graph of source and target data respectively. When constructing A, an edge between nodes i and j exists if x_i and x_j are "close". The criteria for defining "close" can vary in different scenarios.

To construct A_T, since the labels of target data are not available, we define "close" in an unsupervised manner, i.e., nodes i and j are connected by an edge if i is among p nearest neighbors of j or j is among p nearest neighbors of i. Formally, we have:

$$A_T(i,j) = \begin{cases} \frac{x_T^j \cdot x_T^i}{\|x_T^j\| \cdot \|x_T^i\|} & \text{if } x_T^i \in N_p(x_T^j) \text{ or } x_T^j \in N_p(x_T^i) \\ 0 & \text{otherwise.} \end{cases}, \tag{2.1}$$

where $N_p(x_T^i)$ is the set of p nearest neighbors of x_T^i. Note that we compute the similarity matrix A_T with the cosine similarity measure. However, other similarity measures may be used.

For A_S, we take advantage of the available labels of source data, and define "close" in a supervised manner, i.e., nodes i and j are connected if x_i and x_j share the same label:

$$A_S(i,j) = \begin{cases} 1 & \text{if } x_S^i \text{ and } x_S^j \text{ share the same label} \\ 0 & \text{otherwise.} \end{cases}, \tag{2.2}$$

Note that unlike the weight computation for target data, the same weight 1 is used for all edges instead of computing cosine similarity [12].

2.3.2 Multi-Objective Optimization

Given data X and its adjacency graph A, we are trying to find a discriminative feature weight $\mathbf{w}$ which can preserve the local structure of data X. Here we assume that $\mathbf{w}$ projects data points in X to vector $\hat{\mathbf{y}}$, that is $\hat{\mathbf{y}} = X\mathbf{w}$, where X can either be X_S or X_T. We optimize $\mathbf{w}$ from a locality preserving view.

$$
\begin{aligned}
\mathbf{w} &= \arg\min_{\mathbf{w}} \frac{1}{2} \sum_{i,j=1}^{n} (\frac{\hat{y^i}}{\sqrt{d^i}} - \frac{\hat{y^j}}{\sqrt{d^j}})^2 A(i,j) \\
&= \arg\min_{\mathbf{w}} \frac{1}{2} \sum_{i,j=1}^{n} (\frac{\mathbf{w}^T x^i}{\sqrt{d^i}} - \frac{\mathbf{w}^T x^j}{\sqrt{d^j}})^2 A(i,j) \\
&= \arg\min_{\mathbf{w}} \mathbf{w}^T X D^{-1/2} L D^{-1/2} X^T \mathbf{w} \\
&= \arg\min_{\mathbf{w}} \mathbf{w}^T X \bar{L} X^T \mathbf{w}
\end{aligned}
\tag{2.3}
$$

where $L = D - A$ is the graph Laplacian,and $d^i = \sum_j A(i,j)$ measures the local density around x^i. D is a diagonal matrix with $[d^1, d^2, \cdots, d^n]$ as its entries. The normalized graph Laplacian is denoted as $\bar{L} = D^{-1/2} L D^{-1/2}$.

The objective function in (2.3) incurs a heavy penalty if neighboring points x^i and x^j are mapped far away. Intuitively, to minimize (2.3) is to find $\mathbf{w}$ which can ensure that if x^i and x^j are "close" then $\mathbf{w}^T x^i$ and $\mathbf{w}^T x^j$ are close as well.

In order to optimize $\mathbf{w}$ on source and target graph simultaneously, it is clear that the optimization must involve two objective terms: $\mathbf{w}^T X_S \bar{L_S} X_S^T \mathbf{w}$ and $\mathbf{w}^T X_T \bar{L_T} X_T^T \mathbf{w}$. When combining two objective terms, a common practice is to convert two objective terms into a single objective term, by adding up two terms and using a parameter to control the trade-off, i.e.,

$$
\mathbf{w} = \arg\min_{\mathbf{w}} \{\mathbf{w}^T X_S \bar{L_S} X_S^T \mathbf{w} + \alpha \mathbf{w}^T X_T \bar{L_T} X_T^T \mathbf{w}\} \tag{2.4}
$$

The parameter α controls the trade-off between source and target graph. Therefore, it is critical to find a "good" parameter to guarantee that a feature coefficient is obtained by solving (2.4). A standard way to find the "good" parameter is through cross-validation. However, we argue that such paradigm may not be suitable for the unsupervised domain adaptation task, because the target data labels are unavailable, which makes it impossible to perform the cross-validation.

In our approach, instead of converging two separate objective terms into a single objective by introducing a trade-off parameter, we aim to directly solve the following multi-objective optimization, which is one of our main contributions.

$$\mathbf{w} = \arg\min_{\mathbf{w}} \{\mathbf{w}^T X_S \bar{L_S} X_S^T \mathbf{w}, \mathbf{w}^T X_T \bar{L_T} X_T^T \mathbf{w}\} \tag{2.5}$$

We add the following constraints where the last two constraints exclude the solution with eigenvalue 0.

$$\Omega \doteq \{\mathbf{w} \in R \mid \mathbf{w}^T \mathbf{w} = 1, X_S \mathbf{w} \perp D_S^{1/2} \mathbf{1}, X_T \mathbf{w} \perp D_T^{1/2} \mathbf{1}\} \tag{2.6}$$

To solve the above multi-objective optimization problem, we aim to find the Pareto frontier [24]. Before we introduce the concept of Pareto frontier, we first define Pareto improvement.

Pareto Improvement: We set $f_S(\mathbf{w}) = \mathbf{w}^T X_S \bar{L_S} X_S^T \mathbf{w}$ and $f_T(\mathbf{w}) = \mathbf{w}^T X_T \bar{L_T} X_T^T \mathbf{w}$. Given two coefficients $\mathbf{w}$ and $\mathbf{w}'$, we say $\mathbf{w}$ is a Pareto improvements over $\mathbf{w}'$ if and only if one of the following two conditions holds:

$$f_S(\mathbf{w}) < f_S(\mathbf{w}') \wedge f_T(\mathbf{w}) \leq f_T(\mathbf{w}')$$

or

$$f_S(\mathbf{w}) \leq f_S(\mathbf{w}') \wedge f_T(\mathbf{w}) < f_T(\mathbf{w}')$$

When $\mathbf{w}$ is a Pareto improvement over $\mathbf{w}'$, we say $\mathbf{w}$ is better than $\mathbf{w}'$.

Pareto frontier $\hat{P}$ refers to the optimal set of solutions, which satisfy the following three properties:

1. any $\mathbf{w}$ in $\hat{P}$ is better than that not in $\hat{P}$;
2. any two $\mathbf{w}$ in $\hat{P}$ are equally good;

3. for any $\mathbf{w}$ in $\hat{P}$, it is impossible to reduce the cost on one objective function without increasing its cost on the other objective function.

Therefore, the Pareto frontier is a complete set of equally "good" solutions that are superior to any other possible solutions. Despite this good property of Pareto frontier, computing Pareto frontier is unfortunately NP-hard in most cases. However, [24] show that if a multi-objective optimization problem has graph-cut objective terms, then its approximated Pareto frontier can be solved efficiently with a generalized eigendecomposition problem.

2.4 Computing the Pareto Frontier via Generalized Eigendecomposition

For the optimization problem defined in formula (2.5), its Pareto frontier contains infinite number of solutions. In order to make the computation efficient, we made an approximation to original optimization problem, by introducing additional constraints to narrow down the search space. Particularly, we aim to find a subset of solutions in Pareto frontier which is distinctive enough. Therefore, we apply an mutually orthogonal constraint, which is defined as:

$$\hat{\Omega} \doteq \{\mathbf{w} \in \Omega \mid \forall \mathbf{w} \neq \mathbf{w}', X_S\mathbf{w} \perp D_S^{1/2}\mathbf{1}, X_T\mathbf{w} \perp D_T^{1/2}\mathbf{1}\} \tag{2.7}$$

Under an assumption that the null space of $X_S\bar{L_S}X_S^T$ and $X_T\bar{L_T}X_T^T$ do not overlap, the optimization turns into solving a generalized Hermitian definite pencil problem [25]. Then $\hat{\Omega}$ is the set of N eigenvectors of the generalized eigenvalue problem [35].

$$X_S\bar{L_S}X_S^T\mathbf{w} = \lambda X_T\bar{L_T}X_T^T\mathbf{w} \tag{2.8}$$

However, in order to get a stable solution of the above eigen-problem, $X_T\bar{L_T}X_T^T$ is required to be non-singular [35]. Since in our applications, this does not always hold, in order to make the computation numerically stable, we adopt the SVD decomposition described as below.

2.4.1 SVD decomposition

Suppose we have the SVD decomposition of X_T as $X_T = U\Sigma V^T$. If we let $\bar{X_T} = U^T X_T = \Sigma V^T$ and multiply U^T to both sides of the equation, we can rewrite Eq. (2.8) as :

$$\begin{aligned} U^T X_S \bar{L_S} X_S^T \mathbf{w} &= \lambda U^T X_T \bar{L_T} {X_T}^T \mathbf{w} \\ &= \lambda \bar{X_T} \bar{L_T} {X_T}^T \mathbf{w} \end{aligned} \tag{2.9}$$

If we let $\mathbf{w} = U\mathbf{b}$, then we have:

$$\begin{aligned} U^T X_S L_S X_S^T U\mathbf{b} &= \lambda \bar{X_T} \bar{L_T} {X_T}^T U\mathbf{b} \\ &= \lambda \bar{X_T} \bar{L_T} {\bar{X_T}}^T \mathbf{b} \end{aligned} \tag{2.10}$$

Let $\bar{X_S} = U^T X_S$, then we rewrite Eq. (2.10) as:

$$\bar{X_S} \bar{L_S} {\bar{X_S}}^T \mathbf{b} = \lambda \bar{X_T} \bar{L_T} {\bar{X_T}}^T \mathbf{b} \tag{2.11}$$

whose optimal solution for $\mathbf{b}^*$'s can be still solved as the generalized eigenvalue problem. It is easy to check that $\bar{X_T}\bar{L_T}{\bar{X_T}}^T$ have a larger chance to be nonsingular so that the above eigen-problem has a stable closed form.

After we obtain $\mathbf{b}^*$, then $\mathbf{w}^*$ is obtained by solving a set of linear equations $\mathbf{w}^* = U\mathbf{b}^*$. The above function consists of $N-2$ orthogonal cuts in $\hat{\Omega}$. We further compute the Pareto frontier using the Algorithm 1.

2.4.2 Approximation Bound for Our Algorithms

As described above, we compute the orthogonal Pareto frontier as an approximation to the Pareto frontier. Here we create an upper bound on how far a point in the Pareto frontier can be to the orthogonal Pareto frontier. Let $\hat{\Omega} = \{\hat{\mathbf{b}}_i\}_{i=1}^{N-2}$ and $\hat{B} = (\hat{\mathbf{b}}_1, \cdots, \hat{\mathbf{b}}_{N-2})$. Any $\mathbf{b} \in \Omega$ can be represented by a linear combination of $\hat{\mathbf{b}}_i$'s: $\mathbf{b} = \hat{B}\mathbf{a}$, where $\mathbf{a} = (a_1, a_2, \cdots, a_{N-2})^T$. According to [24], we can derive a lower-bound for $\|\mathbf{a}\|$.

$$\|\mathbf{a}\|^2 \geq 1/\sigma_{max}^2(\hat{B}) \tag{2.12}$$

where $1/\sigma_{max}^2(\hat{B})$ is the largest singular value of $\hat{B}$. The larger $1/\sigma_{max}^2(\hat{B})$ is, the closer the two costs on the Pareto frontier and orthogonal Pareto frontier. This effectively bounds the difference between the costs of the cuts on the Pareto frontier and those on the orthogonal Pareto frontier.

2.5 Empirical Study

In this section, results of our analysis of locality preserving projection for domain adaptation with multi-objective learning are presented. First, the data sets and the experiment settings used in this analysis are briefly described. Second, we analyze the classification accuracy for different domain adaptation algorithms on real world data sets. We aim to answer the following questions: (1) how does our algorithms perform on data sets with different distributions on training and test? (2) how does it compare to other domain adaptation algorithms?

2.5.1 Data description and experiment setup

The data set we evaluate first is the USPS handwritten digit database [49]. We extract two data sets from 9298 16x16 handwritten digit data sets. The data set 'USPS1' is constructed

Algorithm 1: Locality Preserving Projection for Domain Adaption with Multi-Objective Learning

input : Data Matrix: X_S, X_T, Label: y_S
output: The set of Pareto optimal weights: $\hat{P}$

Compute the normalized graph Laplacians $\bar{L_S}$, $\bar{L_T}$, and compute the SVD decomposition for $X_S = USV$.
Solve the generalized eigenvalue problem: $U^T X_S \bar{L_S} X_S^T U\mathbf{b} = \lambda \bar{X_T} \bar{L_T} \bar{X_T}^T \mathbf{b}$.
Let $\mathbf{w} = U^T\mathbf{b}$, Normalize all $\mathbf{w}$'s such that $\mathbf{w}^T\mathbf{w} = 1$.
Let $\hat{P}$ be the set of all the $\mathbf{w}$, excluding the two associated with eigenvalue 0 and ∞.
for *all* $\mathbf{w}$ *in* $\hat{P}$ **do**
    for *all* $\mathbf{w}'$ *in* $\hat{P}$ **do**
        if $\mathbf{w}$ *is a Pareto improvement over* $\mathbf{w}'$ **then**
            remove $\mathbf{w}'$ from $\hat{P}$;
            continue;
        end
        if $\mathbf{w}'$ *is a Pareto improvement over* $\mathbf{w}$ **then**
            remove $\mathbf{w}$ from $\hat{P}$;
            break;
        end
    end
end

as follows: the source domain contains all the handwritten digits of '1's which are labeled '+1', all the handwritten digits of '8's which are labeled '-1'. The target domain includes all handwritten digits '7's and '3's with no labels. The data set 'USPS2' is constructed in a similar way as that for 'USPS1'. The source domain contains all handwritten digits '7's with label '+1' and '8's with label '-1'. The target domain includes all handwritten digits '2's and '3's with no labels.

We then evaluate our algorithm on the 14 tumor data sets which were published by Ramaswamy et al. [80], and we downloaded them in the preprocessing version from Statnikov [80]. The data sets contain 14 different human tumor types and 12 normal types. Each type of tumor have only 10s order of subjects and 15009 genes. We extract three transfer learning data sets by coupling normal and tumor samples from the same tissue type together. The details of each data sets are as follows. For 'Bladder-Uterus', the source

domain contains all normal and disease samples with labels extracted from bladder tissue. The target domain includes all normal and disease samples without labels extracted from uterus tissue. For 'Prostate-Uterus', the source domain contains all normal and disease samples with labels extracted from prostate tissue. The target domain includes all normal and disease samples without labels extracted from uterus tissue. For 'Uterus-Pancreas', the source domain contains all normal and disease samples with labels extracted from uterus tissue. The target domain includes all normal and disease samples without labels extracted from pancreas tissue. We aim to predict whether a sample in the target domain is normal or disease given that samples in the source domain with labels.

At last, we evaluate our algorithm on the Lung tumor and brain tumor data sets downloaded from [80]. The source domain for 'Lung1' contains all samples in 'Adeno' and 'Squamous'. The target domain for 'Lung1' contains all samples in 'CIOD' and 'SMCL'. In the same way, the source domain for 'Lung2' contains all samples in 'Adeno' and 'SMCL'. The target domain for 'Lung2' contains all samples in 'CIOD' and 'Squamous'. The source domain for 'Brain' contains all samples in 'Medulloblastoma' and 'Malignant glima'. The target domain for 'Brain' contains all samples in 'AT/RT' and 'PNET'. In this part, we aim to adapt the feature space between source domain and target domain and aim to separate two types of cancers.

The details of each data sets are listed in Table 2.1. It is easy to observe that there are several data sets with extremely small sample size and high dimensional feature space. And for the USPS data sets, the distribution for some features in the training data sets is significantly different from that in the test data sets. We want to see how our algorithms perform on all these various kinds of transfer learning data sets. To make comparisons, we implemented several state-of-art domain adaptation algorithms. GFK embeds the datasets into Grassmann manifolds and constructs geodesic flows between them to model domain shifts [10]. TCA discovers a latent feature representation across domains by learning some transfer components in reproducing Kernel Hilbert space using maximum mean discrep-

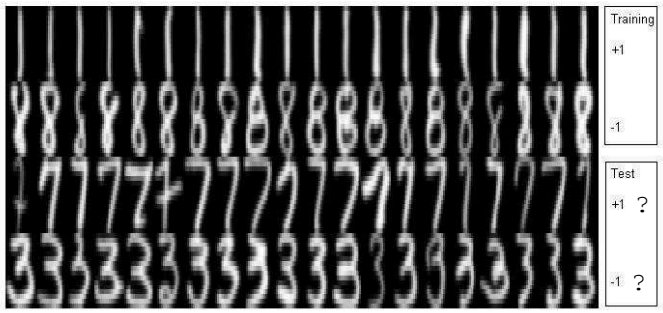

Figure 2.1: The training and test data sets for USPS handwritten digit: the first two rows represent the training data with labels, the third and fourth rows represent test data without labels.

ancy. Our baseline method 'Original' use the original features without learning a new representation for adaptation. We use 1-nearest neighbor classifier to do the classification and report the classification accuracy for each data set.

2.5.2 Experiment Results

The results are summarized in Table 2. From the table, it is easy to observe that our algorithm can achieve better classification on almost all data sets. Most importantly, our approach is more reliable in terms of performance than its competitors when the training and test data sets differ significantly.

For the gene expression data sets, which have very few of samples and high dimension of genes, our approach can find a linear projection which can enhance the classification accuracy. For the USPS data sets, there are quite a lot of features which have different distribution across the source and target domains. The new feature representation computed by GFK failed to adapt the source domain to the target domain.

Table 2.1: Summary of Tumor Data Sets

Datasets	Training Pos vs Neg	Testing Pos vs Neg	Features
$Lung_1$	20 : 6	17 : 21	12600
$Lung_2$	21 : 6	17 : 20	12600
$Brain$	7 : 14	14 : 15	10367
$USPS1$	664 : 731	1858 : 645	256
$USPS2$	644 : 731	542: 645	256
$Bladder - Uterus$	11 : 7	11 : 6	15009
$Prostate - Uterus$	14 : 9	11 : 6	15009
$Uterus - Pancreas$	11 : 6	11 : 10	15009

Datasets	Original	TCA	GFK	Our method
$Lung_1$	0.5263	0.6053	0.6053	**0.8158**
$Lung_2$	0.7027	0.6406	0.6406	**0.8919**
$Brain$	0.8966	0.8621	0.8966	**0.9655**
$USPS1$	0.8554	0.5610	0.6117	**0.9036**
$USPS2$	0.8569	0.7787	0.7889	**0.8833**
$Bladder - Uterus$	0.6534	0.6191	**0.7110**	0.7059
$Prostate - Uterus$	0.7059	0.7647	0.7647	**0.8235**
$Uterus - Pancreas$	0.7143	0.7143	0.7619	**0.8095**

Table 2.2: Classification Accuracy of Different Domain Adaptation Algorithms on Tumor Datasets. The best results of each data set are highlighted in bold.

2.6 Conclusion and Future Work

In this paper, we explore the locality preserving projection for domain adaptation with multi-objective learning. We propose multi-objective formulation for domain adaptation. The search space of our objective is the joint numerical range of two graphs. We find a relaxed mutually orthogonal optimal sets by using Pareto optimizations. The effectiveness of our approach is evaluated on the benchmark data sets with comparison to the state-of-the-art algorithms. The pragmatic benefits of our approach over existing domain adaptation algorithms are: $1)$ the users do not need to specify the trade-off parameters; $2)$ the training and test data sets do not need to be similar to each other. Our algorithm can find the new feature representation which can effectively preserve the local structure.

Chapter 3

Transductive Domain Adaptation with Affinity Learning

3.1 Introduction

In recent years, domain adaptation has gained significant attention in many areas of applied machine learning, including bioinformatics, speech and language processing, computer vision etc. In these practical problems, given that the instances in the training and testing domains may be drawn from different distributions, traditional learning method can not achieve good performance on the new domain. Domain adaptation algorithms are therefore designed to bridge the distribution gap between training (source) data and testing (target) data. Domain adaptation methods seek to eliminate the difference between source and target distributions.

In this paper, we propose a transductive method to explicitly improve intra- and inter-domain similarities. Our contribution is two-fold: first, we perform affinity learning via graph diffusion to bridge the distribution gap of source and target domain. The key idea is to exploit the existence of a subset of data points in the target domain which distributes similarly to the data points in the source domain. We denote this subset of data points as Bridge Points (BP). Through graph diffusion, we propagate the similarities between BP and other data points in the target domain, as well as the similarities between BP and data points in the source domain. Affinity learning is able to give robust pair-wise similarities of data points, since all paths between all pairs of data points are considered. In this way, affinity learning can bridge distribution gap of source and target domain. As our experimental results clearly demonstrate, our assumption that part of the target data is similar to part of the source data is often satisfied by real world data sets. Figure 3.1 illustrates our motivation.

Our second contribution is to adjust the intra- and inter- domain similarities. The intuition is that data points in the same domain are often more similar to each other than to those in different domain. In graph diffusion process, this makes the similarity propagation from data points in source domain to data points in target domain ineffective. Therefore, the proposed adjustment of the intra- and inter- domain similarities is a key step in making

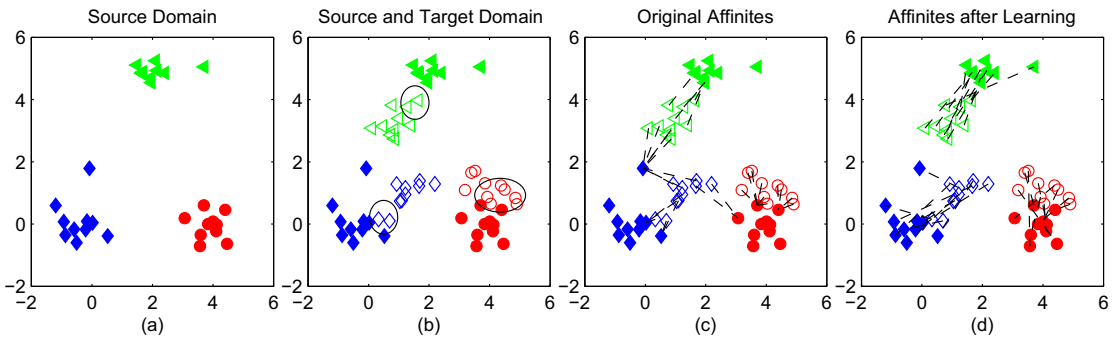

Figure 3.1: Schematic illustration of utilizing affinity learning for unsupervised domain adaptation. (a) Data points in source domain. Each color represents one class. (b) Data points in source and target domains. Solid shapes represent the data points in the source domain, hollow shapes represent the data points in the target domain. Black circles mark the Bridge Points, which is subset of data instances in the target domain and have similar distribution as data instances in the source domain. (c) The lines connect points in the target domain to their nearest neighbors in the source domain with the original similarities. (d) The lines connect points in the target domain to their nearest neighbors in the source domain after the affinity learning.

the affinity propagation successful. We balance the intra- and inter- domain edges by picking equal number of nearest neighbors in source and target domain for each data point and also re-weight intra- and inter-domain edges.

In summary, given the similarity matrix of source and target data, the procedure of our framework includes the following key steps:

Similarity Adjustment: re-weight intra- and inter- domain similarities.

Affinity Learning: iteratively learn similarities in joint geometric structure via Tensor Product Graph Diffusion (TPGD)[93].

Spectral Embedding: apply spectral embedding on diffusion matrix to get a low-dimensional representation.

In this paper, we use Tensor Product Graph Diffusion(TPGD) [93] to capture the joint manifold structure for the source and target domain. As demonstrated in [93], TPGD can robustly discover the true, underlying manifold structure in image retrieval. We utilize TPGD to learn joint geometric structure in the context of domain adaptation when training and testing are drawn from different distributions.

We examine the proposed method on several benchmark datasets which consists of visual object recognition and text classification tasks. The proposed method outperforms state-of-the-art methods. In particular, it achieves the best performance among all compared methods on 6 out of 8 visual object recognition tasks and 6 out of 6 text classification tasks.

The rest of the paper is organized as follows. We first give a brief review of related works in Section 3.2. In Section 3.3, we describe the proposed affinity learning for domain adaptation task. In particular, we describe how to construct the transition probability matrix with similarity adjustment. We also show how to perform graph diffusion on a tensor product graph to obtain robust similarities. In Section 3.4, we present our experimental results on benchmark datasets and compare it to several state-of-the-art methods. Finally, we come to the conclusion is in Section 3.5.

3.2 Related Work

Domain adaptation has been extensively studied in many research areas [68, 50, 16, 96]. Domain adaptation can be categorized into three types. The first type are self-labeling approaches, which include self-training [71] and co-training [16]. The second type of algorithms proposes to weight or select training instances to minimize the discrepancy distance [48, 54]. Our work belongs to the third type, which aims at finding "good" feature representations to minimize domain divergence and classification error, such as [9, 10, 66]. In particular, for object recognition application in computer vision, many works have been proposed to learn new feature representation, such as [59, 74, 36]. Compared to existing approaches, our method focus on affinity learning to bridge the distribution gap between source and target domain.

While our work share some common components compared to graph-based semi-supervised method, such as [99] where graph is used to propagate labels, the key

difference is that we aim to solve the domain adaptation problem and our goal is to use affinity learning to improve the noisy pairwise similarities due to domain shift. That motivates us to reweight the inter- and intra- domain edges, and use spectral embedding to obtain the low dimensional domain-invariant data representation.

There are also several works attempting to solve transfer learning in a transductive setting [72, 56, 33]. They apply label propagation to zero-shot and few-shot learning based on attribute graph or semantic graph. [90] exploits the mixture distribution to refine the classification labels. These work did not try to improve pair-wise similarities.

3.3 Proposed Approach

We assume that our data originate from two domains, Source (S) and Target (T). Source data $D_S = \{(x_S^1, y_S^1), (x_S^2, y_S^2), \cdots, (x_S^{N_S}, y_S^{N_S})\}$ is fully labeled, each pair (x_S^i, y_S^i) lies in $R^d \times y$ space. The source data are sampled from some distribution $P_S(X, Y)$. The target data has equal dimension d as the source data but is sampled from a different distribution $P_T(X, Y)$. We denote the target data as $D_T = \{(x_T^1, ?), (x_T^2, ?), \cdots, (x_T^{N_T}, ?)\}$, whose labels are unknown. Given D_S and D_T, our goal is to infer the class labels of data points in D_T.

In the rest of this section, we first describe how to construct the transition matrix P for source and target data jointly. We then iteratively learn the joint geometric structure and capture true similarities among data points in source and target domain. After we get the diffusion matrix, we compute the Laplacian graph and solve the smallest K eigenvectors to obtain a new feature representation of data points in source domain and target domain. After that, any classification approach can be adopted to predict labels for target data. In this work, we choose SVM classifier with linear kernel.

3.3.1 Cross-domain Graph Construction

The goal of this section is to construct the transition matrix of a graph G whose nodes consist of data points in both source and target domain. We use P_{SS} and P_{TT} to denote the transition probability matrices of data points in the source and target domains respectively. P_{ST} and P_{TS} denote the transition probability matrix of data points across domains. We construct the overall transition matrix as follows:

$$P = \begin{bmatrix} \beta P_{SS} & (1-\beta) P_{ST} \\ (1-\beta) P_{TS} & \beta P_{TT} \end{bmatrix} \tag{3.1}$$

where β controls the relative importance of the intra- and the inter- domain transition probabilities and $\beta \in [0, 1]$. Empirically, β can be set by solving

$$\frac{\beta}{1-\beta} = \frac{2N_S N_T}{N_S^2 + N_T^2} \tag{3.2}$$

which calibrates the average inter- and intra- domain edge weights to be close.

P_{SS} and P_{TT} are row-wise normalized similarity matrices which are computed as:

$$P_{SS} = D_{SS}^{-1} A_{SS} \quad P_{TT} = D_{TT}^{-1} A_{TT} \tag{3.3}$$

where A_{SS} is the similarity matrix of data points in source domain and A_{TT} encodes the similarities between data points in target domain. D_{SS} and D_{TT} are the diagonal matrices of the row sums of A_{SS} and A_{TT}.

To compute A_{SS}, we take advantage of the available labels of source data, and define "closeness" in a supervised manner, i.e., nodes i and j are connected if x_i and x_j share the

same label. The similarity matrix A_{SS} is defined as follows:

$$A_{SS}(i,j) = \begin{cases} \frac{x_S^j \cdot x_S^i}{\|x_S^j\| \cdot \|x_S^i\|} & \text{if } x_S^i \text{ and } x_S^j \text{ share the same label} \\ & \text{and } x_S^i \in N_p(x_S^j) \text{ or } x_S^j \in N_p(x_S^i) \\ 0 & \text{otherwise.} \end{cases}, \tag{3.4}$$

Here, we use N_p to denote the p nearest neighbors. In the source domain, the label information is embedded into the similarity matrix. While we compute the similarity matrix A_{SS} with the cosine similarity measure, other similarity measures may also be applicable.

To compute A_{TT}, since the labels of target data are not available, we define "closeness" in an unsupervised manner, i.e., nodes i and j are connected if i is among p nearest neighbors of j or j is among p nearest neighbors of i. Formally, we have:

$$A_{TT}(i,j) = \begin{cases} \frac{x_T^j \cdot x_T^i}{\|x_T^j\| \cdot \|x_T^i\|} & \text{if } x_T^i \in N_p(x_T^j) \text{ or } x_T^j \in N_p(x_T^i) \\ 0 & \text{otherwise.} \end{cases}, \tag{3.5}$$

where $N_p(x_T^i)$ is the set of p nearest neighbors of x_T^i.

Similarly cross-domain transition probability matrices P_{ST} and P_{TS} are computed as follows:

$$P_{ST} = D_{ST}^{-1} A_{ST} \quad P_{TS} = D_{TS}^{-1} A_{TS} \tag{3.6}$$

where A_{ST} denotes the cross-domain similarities. D_{ST} and D_{TS} are the diagonal matrices of the row sums of A_{ST} and A_{TS}. A_{ST} is computed as follows:

$$A_{ST}(i,j) = \begin{cases} \frac{x_S^i \cdot x_T^j}{\|x_S^i\| \cdot \|x_T^j\|} & \text{if } x_S^i \in N_p(x_T^j) \text{ or } x_T^j \in N_p(x_S^i) \\ 0 & \text{otherwise.} \end{cases}, \tag{3.7}$$

To summarize, we introduce two major differences in the transition matrix construction process which are tailored for unsupervised domain adaptation task: First, we add supervised information to the similarity matrix A_{SS} to remove noisy entries in A_{SS}. Second, we

control the relative importance of intra-domain and inter-domain transition probabilities. When building KNN connected graph, we balance the intra- and inter- domain edges by picking equal number of nearest neighbors in source and target domain for each data point. We also perform a reweighting on the intra- and inter- domain similarities. As demonstrated by our experiments in Section 3.4.2, these two steps greatly boost the performance of affinity learning which results in a better data representation, and therefore a higher prediction accuracy.

3.3.2 Diffusion Process on Tensor Product Graph

In this section we review tensor product graph diffusion process introduced in [93]. Given the edge (transition probability matrix) P, we define $Q^{(1)} = P$ and

$$Q^{(t+1)} = P\, Q^{(t)}\, P^T + I, \tag{3.8}$$

where I is the identity matrix. We iterate (3.8) until convergence. Let us denote the limit matrix by $Q^* = \lim_{t\to\infty} Q^{(t)}$. A closed form expression for Q^* is as follows:

$$\lim_{t\to\infty} Q^{(t)} = Q^* = P^* = \mathrm{vec}^{-1}(\sum_{i=0}^{t} \mathbb{P}^i)\ \mathrm{vec}(I)). \tag{3.9}$$

The proof of the convergence of (3.8) and closed form equation can be found in [93], where $\mathbb{P}$ is the tensor product of P with itself. Since $Q^* = P^*$, we obtain that the iterative algorithm on Q defined by (3.8) yields the same similarities as the TPG diffusion process on $\mathbb{P}$ for a sufficient number of iterations.

3.4 Empirical Study

In this section, we present our experimental results on visual object recognition tasks. We set K and p in our method through cross-validation based on classification error of data

Table 3.1: Performance Gain Analysis on Visual Object Recognition , where: C : Caltech, A : Amazon, W : Webcam, D : DSLR. SE: spectral embedding; SA: similarity adjustment; TPGD: tensor product graph diffusion; Our Method = SA+TPGD+SE.

%	A-D	A-W	D-A	D-W	W-A	W-C	C-D	C-W	Average
Baseline	40.8	41.7	32.3	73.6	34.1	29.9	42.0	40.7	41.9
SE	40.1	40.7	34.4	66.4	35.7	28.7	47.1	43.7	42.1
SA + SE	45.8	44.4	41.1	89.5	38.7	35.0	52.8	49.1	49.6
TPGD + SE	43.9	41.0	34.8	74.9	**39.8**	35.2	54.1	48.5	46.5
Our Method	**50.3**	**49.0**	**40.5**	**92.2**	39.7	**36.8**	**55.4**	**52.9**	**52.1**

samples in source domains. We first compare to the baseline approach and evaluate the performance gain at each step and give detailed analysis. This provides clear insight about the merits of the proposed method. Our results on benchmark datasets are also favorable when compared to several state-of-the-art domain adaptation methods.

We first describe our experiment settings, and then validate our method on visual object recognition and text classification task. The obtained results clearly demonstrate that our method outperforms several state-of-the-art domain adaptation methods on most of the datasets.

3.4.1 Performance Gain Analysis

We perform experiments using 4 object recognition datasets, which includes: Amazon, Webcam, DSLR. These three datasets are first introduced in [74]. Additionally, we use Caltech-256 in [41] as the fourth dataset to further evaluate the proposed methods. Each dataset is treated as a domain and 10 common object categories are extracted. We downloaded the processed datasets with SURF features from [10]. We conduct each experiment using every pair of source and target dataset. We report the recognition accuracy on every pair of source and target dataset.

Table 3.2: Recognition Accuracy of Benchmark Domain Adaptation Method for Visual Object Recognition, where C : Caltech, A : Amazon, W : Webcam, D : DSLR.

%	A-D	A-W	D-A	D-W	W-A	W-C	C-D	C-W
Baseline	40.8	41.7	32.3	73.6	34.1	29.9	42.0	40.7
TCA[66]	36.3	27.8	28.7	82.0	24.2	22.5	45.2	32.5
KMM[48]	42.7	42.4	36.0	83.0	31.9	29.0	53.5	45.8
GFK[10]	42.7	40.7	36.2	76.3	31.8	30.9	43.3	44.7
LandMark[36]	47.1	46.1	33.4	78.0	**40.2**	35.4	**57.3**	49.5
Our Method	**50.3**	**49.0**	**40.5**	**92.2**	39.7	**36.8**	55.4	**52.9**

3.4.2 Visual Object Recogonition

As the baseline approach, we adopt the original features and train a linear SVM model on source domain. To illustrate the significance of performance gain using affinity learning to facilitate domain adaptation, we study four variants of our method. In the first variant, we apply spectral embedding directly to the original similarity matrix. In the second variant, we add intra- and inter- domain similarities reweighting before applying spectral embedding. In the third variant, we apply graph diffusion to the original similarity matrix before applying spectral embedding. In the final variant, we put all components together which is the proposed approach.

We compare the recognition accuracy of the baseline approach and the 4 variants in Tabel 3.1. We can see that low-dimensional feature representation obtained by spectral embedding can preserve most of the information for each dataset, whose accuracy is comparable to the baseline, but with no improvement. If we adjust intra- and inter- domain similarities and apply spectral embedding, the average recognition accuracy improves 7.5% compared to that of the baseline. If we apply affinity learning and spectral embedding together, the average recognition accuracy improves 4.6% compared to that of the baseline. If we combine adjusting intra- and inter- domain similarities and affinity learning through graph diffusion, the performance improves 10.0% compared to that of the baseline method. Overall, these results demonstrate that adjusting intra- and inter- domain similarities can

facilitate the affinity learning, and affinity learning can provide more reliable affinities for data points in joint manifold.

We compare the proposed method to several state-of-the-art methods: KMM [48], TCA [66], GFK [10], LandMark [36]. Table 3.2 summarizes accuracy of object recognition on 8 pairs of source and target domains obtained from the four datasets. For the compared methods, most results are quoted from [36], except for D-A and D-W which we generated using the code downloaded from authors' websites. The average recognition accuracy of our method improves 3.7% when compared to that of the second best method 'LandMark'. Our method performs the best on 6 out of 8 pairs of domains.

3.4.3 Cross-domain Text Classification

In this section, we test the proposed approach on cross-domain text classification tasks. We use Reuters-21578 Processed dataset[1]. Reuters dataset consists of 3 different domains: Orgs, People and Places, and has 6570 instances in total. We conduct 6 unsupervised domain adaptation experiments, which are Orgs to People, Orgs to Places, People to Orgs, People to Places, Places to Orgs, Places to People. We also compare our method to KMM [48], TCA [66], GFK [10], LandMark [36]. Table 4.3 reports classification accuracy for 6 unsupervised cross-domain text classification tasks. Our method achieves better results than the other state-of-the-art methods. The average classification accuracy of our method on 6 domain adaptation tasks improves 10% when compared to the baseline method. Compared to KMM and LandMark, the classification accuracy of our method is 8.9% and 7.6% higher respectively.

We also conduct another experiment in which we randomly pick target data together with their labels and treat them as source data for feature representation learning and training the prediction model. We then test the model on the rest of unlabeled data in the target domain. We repeat this 10 times and report the average classification accuracy. We com-

[1] http://www.cse.ust.hk/TL/index.html

Table 3.3: Classification Accuracy on Cross-Domain Text Classification, where Pe is short for People and Pl is short for Places. The proposed method performs the best on all 6 pairs.

%	Orgs-Pe	Orgs-Pl	Pe-Orgs	Pe-Pl	Pl-Orgs	Pl-Pe
Baseline	69.3	65.5	70.2	51.5	65.9	56.1
TCA [66]	67.9	63.5	73.2	52.8	59.2	55.4
KMM[48]	64.3	69.7	74.3	53.7	65.5	57.6
GFK [10]	68.1	64.8	71.4	56.9	60.9	56.1
LandMark[36]	68.6	64.5	74.9	60.5	64.6	61.3
Ours	**79.1**	**73.1**	**84.6**	**67.8**	**73.4**	**62.0**

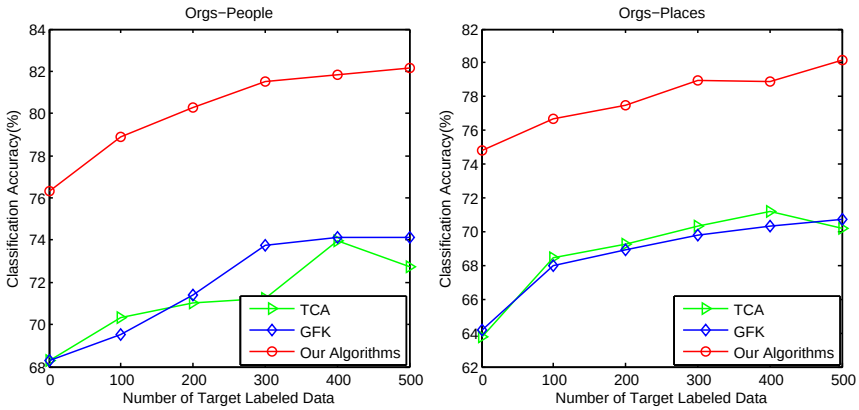

Figure 3.2: Results on two domain adaptation tasks under varying amounts of labeled target data. The graphs show the average classification accuracy averaged over 10 runs (with randomly selected labeled instances).

pare our method to KMM [48] and LandMark [36]. Figure 3.2 shows the results on two domain adaptation tasks: Orgs to People and People to Places. The curves illustrate the classification accuracy using different numbers of labeled target data. As we can see from the figure, our method is able to learn domain-invariant feature representation even when there is no labeled target data used (left-most point where the number of labeled target data equals to 0). As the number of labeled target data increase, the classification accuracy for domain adaptation tasks increase for all three methods, while our method can consistently achieve better performance than the other two compared methods.

3.5 Conclusions

We demonstrate that affinity learning can be a very successful tool for domain adaptation. Our method is able to learn the joint geometric structure of source and target domain based on the preservation of intra-domain and across domain information. We first construct the similarity matrix which encodes the intra- and inter- domain similarities. We then iteratively learn 'true' similarities for data points in joint manifold with Tensor Product Graph Diffusion (TPGD). At last, we apply spectral embedding on diffusion matrix to get the low-dimensional feature representation. The effectiveness of our method is validated on standard benchmark datasets for visual object recognition and text classification. Empirical results show that the proposed method achieves robust performance and outperforms state-of-the-art methods.

Chapter 4

Subspace Learning with Data Alignment for Domain Adaptation

4.1 Introduction

A typical assumption for supervised methods is that the training and test data have the same distribution. However, in real world tasks, this assumption does not always hold. Domain adaptation has been widely studied in recent years, which aims to solve the supervised learning problem when there exists domain shift between source (training) and target (test) data. Many existing domain adaptation approaches are based on subspace learning, such as [32, 38, 10, 8]. A common philosophy of those approaches is to find a subspace in which the distribution discrepancy of source and target domain is reduced. In one of the pioneer works that explores the subspace learning for domain adaptation, [9] proved that if the subspace that preserves the most variation of the source data overlaps with that of the target data, then the predictive model trained for source domain can be effectively transferred to target domain. On the other hand, if the subspace of source domain is orthogonal to the subspace of target domain, then the knowledge learned from source data can not be transferred to target domain. This clearly demonstrates the importance of finding a shared subspace in which the data variation of both the source and target data is preserved.

In this paper, we propose a novel linear subspace learning approach for domain adaptation. Like [32, 10, 8] the linear subspace is defined using a set of basis vectors. Unlike the existing approaches, our method finds the subspace using both source and target data while explicitly aligning the target subspace to the source subspace. Our observation is that: to obtain a subspace good for domain adaptation, simply computing principal components using both source and target data might be problematic, due to the domain shift. For instance, in the digits image classification, the test images could be rotated, shifted or blurred compared to the training images. In that case, the basis of the training space is not compatible to those of the test space. Therefore, it is impossible to find a set of basis vectors which are optimal for both training and test images. The other example is the cross domain text classification in which the texts in two languages may have the same intrinsic subspace because the words in two vocabularies may have one to one correspondences. However,

due to the difference in two dictionaries, the subspace may also not be compatible. It is easy to notice that all the domain shifts mentioned above, such as image rotation, shifting, blurring, or the permutation in bag of word representations, can be in fact modeled as a linear transformation between the source data and target data. We demonstrate our motivation using a toy example in Figure 4.1.

We make two key contributions in the proposed approach. Our first contribution is to perform data alignment and subspace learning simultaneously. We introduce a novel PCA-like objective function which consists of two variables, the subspace basis, and the linear transformation between source basis and target basis. To jointly optimize the objective function over these two variables, we propose an iterative algorithm based on alternative minimization, and prove that it converges to a local optimum. Our second contribution is to exploit labels available for source data to obtain a better subspace for prediction. We aim to incorporate label information to subspace based domain adaptation algorithms. While the unsupervised subspace learning, such as PCA, is able to preserve the data variance, it fails to consider the label information therefore may not be robust in some cases. We use Figure 5.1 to demonstrate this. While the source data has similar variance along both directions U_s and U_s', U_s is clearly more suitable for building a predictive model, since the positive and negative data samples are better separated. We show that the label information of source data can be naturally integrated into our objective function, and the same algorithm can be used to solve both unsupervised and supervised version of our objective functions. We examine the proposed method on both synthetic dataset as well as benchmark datasets for real world domain adaptation problems. Our method achieves state-of-the-art results.

The rest of the paper is organized as follows: First, we discuss several works which are related to the proposed approach. Second, we define the subspace learning problem formally, describe the alternative minimization algorithm and prove its convergence. Then, we describe the experimental settings and discuss the empirical results. At last, we give the conclusion.

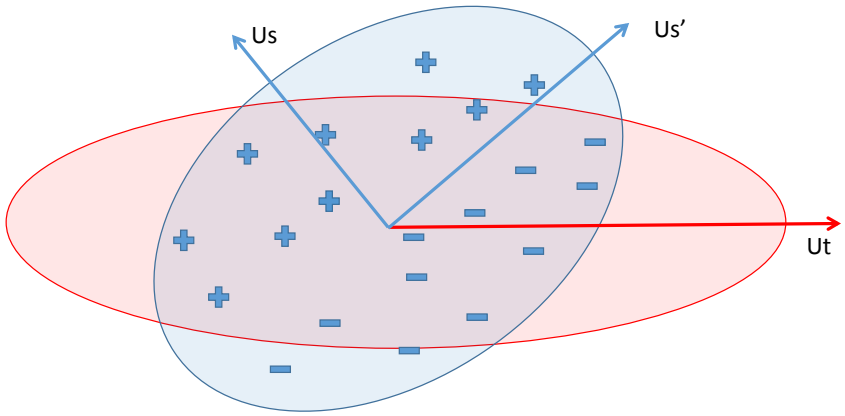

Figure 4.1: A toy example to demonstrate our motivation for subspace learning with data alignment. Both source (blue circle) and target (red circle) data are sampled from a gaussian distribution. U_t is the subspace that preserves the most variance for target data. U_s and U_s' are two subspace candidates for source domain, but U_s is better for label prediction than U_s'. A rotation of target data U_t would align it with U_s and reduce the domain discrepancy.

4.2 Related work

Domain adaptation have gained significant attention in recent years [54, 48, 23, 66, 17, 13], [89]. Domain adaptation can be categorized into three types: The first type of method are self-labeling approaches, which include self-training [71] and co-training [16]. The second type of approaches propose to select or reweight source domain instances to minimize the distribution discrepancy between source and target domain, such as [48, 18, 54]. Our work is along with the third type, which aims at finding common subspace to minimize domain divergence. As a result, features used to build the prediction model in source domain also have support in target domain, and the trained model can be transferred to target domain more effectively. Subspace based domain adaptation gains a lot of popularity due to its promising results on many real world applications, such as computer vision [10, 32, 78, 38] and natural language processing [9, 8, 67]. In general statistical modeling, a good subspace is the one that preserves most of the data variance. A subspace can be uniquely characterized by a set of basis vectors, which are often identified as the eigenvectors with principal component analysis (PCA). Subspace based domain adaptation approaches assume that

source data and target data lie in two different intrinsic low dimensional subspaces. [8] propose a coupled subspace learning algorithm in which the subspaces of source and target domain are first identified independently. Labeled source data is projected into both the source subspace and the target subspace, and a linear prediction model is learned using features in both subspaces. They showed that only the weights on the features in the target subspace are transferred during testing. Similar to [8], both [10] and [38] assume that source data and target data lie in two different low dimensional subspaces. The key idea is to learn domain-invariant features by using intermediate subspaces. They show that both source and target subspaces are on the Grassmannian manifold, and a geodesic flow curve connecting these two subspaces captures the incremental changes. [65] attempt to discover a good latent feature representation across domains by minimizing the feature distribution difference, which is measured by the Maximum Mean Discrepancy statistic. The approach solves a semi-definite programming (SDP) and directly gives the kernel matrix. In [67], an improved version was proposed, which is called ”transfer component analysis”. The method can reduce the distance between domain distributions dramatically by projecting the data onto the learned transfer components. This algorithm learns a kernel function that can be applied on new data sets.

The closest work to ours is [32], which proposes a subspace alignment algorithm that explicitly aligns the bases of the subspaces to reduce the domain shift. There subspaces for source and target domain are first computed independently followed by a subspace alignment step. This means that the subspace for target domain only depends on target data samples. The goal of subspace alignment is only to align the coordinates, so that the source samples can be projected into the target subspace. Our method explicitly aligns the data in two domains using a linear transformation while simultaneously finding a subspace which preserves the most data variance The key difference in our approach is that our goal is to find a shared intrinsic subspace for both domains. In our formulation, the source samples

also help to determine the subspace of the target data. The other difference is that we perform the data alignment step in the original space rather than in the subspace like [32].

4.3 Methodology

We assume there are N_s labeled source data samples (X_s, Y_s) drawn from distribution P_s and N_t unlabeled target data samples $(X_t, ?)$ drawn from another distribution P_t. We have $X_s \in R^{p \times N_s}$ and $X_t \in R^{p \times N_t}$. Both source and target data have the same dimension p. Our goal is to find a common subspace, in which the domain divergence is minimized, so we are able to infer the correct labels of the target data.

4.3.1 Problem Formulation

A good common subspace should be able to preserve the variance, or equivalently minimize the reconstruction error, for both source and target domain. To find such a common subspace, one can simply solve the following PCA problem:

$$\begin{aligned} &\arg\min_{U} \|X_s - UU^T X_s\|_F^2 + \|X_t - UU^T X_t\|_F^2 \\ &s.t. \quad U^T U = I \end{aligned} \tag{4.1}$$

where $U \in R^{p \times K}$ of which each column is a basis vector. Solving Eq. (4.1) only makes sense when the true subspace for both domains agree or just differ slightly. However, if there exists dramatic domain shift between source and target domains, the solution of Eq. (4.1) does not solve the domain adaptation problem. As we point out in the introduction section, in many real world applications, such as image classification or cross language text classification, the true subspace between two domains could have significant domain shift, which should not be neglected.

Therefore, we use a linear transformation $M \in R^{p \times p}$ to explicitly model the domain shift between the true subspace of two domains. In particular, we use U_s and U_t to denote the subspace of source and target domain respectively, and assume that the number their basis vectors are the same, i.e., $U_s \in R^{p \times K}$ and $U_t \in R^{p \times K}$, then $U_s = MU_t$.

By incorporating $U_s = MU_t$ into Eq. (4.1), we aim to solve:

$$\begin{aligned} &\arg\min_{U_t, M} \|X_s - (MU_t)(MU_t)^T X_s\|_F^2 + \|X_t - U_t U_t^T X_t\|_F^2 \\ &s.t.\ U_t^T U_t = I, \quad M^T M = I \end{aligned} \tag{4.2}$$

Eq. (4.2) is a non-convex function of U_t and M, therefore in polynomial time, it is only possible to find a locally optimal solution. In the next section, we present an efficient iterative algorithm to solve Eq. (4.2).

4.3.2 Alternating Optimization

Since it is difficult to optimize over U_t and M simultaneously, we adopt an alternating minimization approach. At each step, we alternatively optimize over U_t and M with the other one fixed. The details are described below.

(1) Initialization

Since the iterative alternating optimization procedures efficiency is greatly affected by the initialization step, in this paper, we initialize $M = I$ rather than random allocation.

(2) Optimization over U_t with fixed M

If M is constant, solving Eq. (4.2) turns into solving:

$$\begin{aligned} &\arg\max_{U_t} \mathbf{tr}(U_t^T (M^T X_s X_s^T M + X_t X_t^T) U_t) \\ &s.t. \quad U_t^T U_t = I \end{aligned} \tag{4.3}$$

It is easy to see that this is similar to the objective in the PCA problem. Therefore the columns of the optimal solution U_t of Eq. (4.3) correspond to the top K eigenvectors of

$M^T X_s X_s^T M + X_t X_t^T$. We also want to point out that in the original formulation Eq. (4.2), the linear transformation M is applied to the target basis U_t. Eq. (4.3) provides us another perspective: we first transform the source samples by applying M to X_s then solve the PCA problem jointly using target samples and transformed source samples.

(3) Optimization over M with fixed U_t

If U_t is constant, we can drop the second term in Eq. (4.2), as well as the constraints with respect to U_t. Then Eq. (4.2) turns into:

$$\begin{aligned} &\arg\min_{M} \|X_s - (MU_t)(MU_t)^T X_s\|_F^2 \\ &s.t. \quad M^T M = I \end{aligned} \tag{4.4}$$

We use the method of Lagrange multipliers to find the local minimum of Eq. (4.4) . We study the Lagrange function defined by

$$\mathbf{tr}(U_t^T M^T X_s X_s^T M U_t) - \lambda(M^T M) \tag{4.5}$$

If we take the derivative of Eq. (4.5) over M and set the derivative to 0, we have

$$X_s X_s^T M U_t U_t^T - \lambda M = 0 \tag{4.6}$$

After multiplying by U_t, we get

$$X_s X_s^T M U_t - \lambda M U_t = 0 \tag{4.7}$$

From Eq. (4.7), this is the same problem as we solved for U_t, which is again a PCA problem. Therefore, to minimize Eq. (4.5), MU_t should be the top K eigenvectors of $X_s X_s^T$. In order to obtain a closed form solution exists for Eq. (4.5) which always decreases

the objective, we first compute $U_s = MU_t$ as the first K eigenvectors of $X_s^T X_s$. We then multiply the pseudo inverse of $U_t^+ = (U_t^T U_t)^{-1} U_t^T = U_t^T$, to both sides of the equation, we then get the closed form solution for M as follows:

$$M = U_s U_t^T \tag{4.8}$$

After we get U_t and M, during training, we let $X_s' \leftarrow U_t M X_s$, and train a classifier (linear SVM in our experiments) using X_s' and Y_s. During test, we apply the classifier to $X_t' \leftarrow U_t X_t$ to predict the labels for target data. We summarize our approach in Algorithm (3).

(4) Convergence Analysis

We prove that our approach in Algorithm 3 converges to a local minimum.

Theorem 1. *The objective function value in Eq. (4.2) is non-increasing under the optimization procedure in Algorithm Eq. (3).*

Proof. To prove Theorem (1), we only need to prove that the objective function value of Eq. (4.2) is non-increasing after each step in line 3 and in line 5. With fixed M, the objective function value of Eq. (4.2) with respect to U_t equals to the objective function value of Eq. (4.3). With fixed U_t, the objective function value of Eq. (4.2) with respect to M equals to the objective function value of Eq. (4.4). Since the objective function values in Eq. (4.3) and Eq. (4.4) are guaranteed to converge to some local mimima and the Frobenius norm has 0 as a lower bound, so the convergence of our iterative algorithm to a local optimum is guaranteed. □

4.3.3 Utilizing Labels of Source Data

We first review the supervised PCA method introduced in [3], which incorporates the label information into PCA based on HSIC dependency criteria [40], so that the PCs are more relevant to the responsible labels. If we use X to denote data, U to denote the subspace

Algorithm 2: Subspace Learning with Data Alignment.

Input : 1. Source data (X_s, Y_s), target data $(X_t, ?)$.
2. Number of subspace basis vectors K.

Output: Label of target data Y_t

Initialize M as an identity matrix.
repeat
  Update U_t by solving Eq. (4.3).
  Update $U_s = MU_t$ by solving Eq. (4.5).
  Update M using Eq. (4.8).
until *Eq. (4.1) converges*;
$Y_t \leftarrow Classifer(U_t M X_s, Y_s, U_t X_t)$

Table 4.1: Classification Accuracy for USPS Digit Recognition with Rotation or Gaussian Blur Domain Shift.

%	45°Rotation	90°Rotation	Blur(width=2)	45°Rotation+Blur(width=2)
No Adaptation	78.22(8.63)	59.20(7.11)	80.57(8.28)	73.61(10.25)
TCA [66]	89.65(3.94)	82.55(11.34)	**86.00(0.57)**	87.04(4.90)
GFK [10]	71.10(13.45)	62.10(12.41)	72.92(10.35)	64.89(13.34)
SA[32]	90.44(3.31)	77.15(7.60)	74.54(3.49)	86.31(2.79)
Ours(Unsupervised)	91.13(1.27)	86.60(8.23)	82.95(2.22)	88.71(1.27)
Ours(Supervised)	**91.50(1.42)**	**88.67(4.95)**	84.14(2.20)	**88.98(1.30)**

basis, the supervised PCA solves:

$$\arg\max_{U} \mathbf{tr}(U^T X L X^T U) \; s.t. \quad U^T U = I \tag{4.9}$$

where L is the kernel matrix computed using label Y. For classification problem, we use a simple kernel function, if two samples have the same label, the corresponding value in L is set as 1, otherwise 0. The solution to this problem is the eigenvectors of the top K eigenvalues of XLX^T.

Since L is the kernel matrix, which is semi-definite, it can be decomposed as $L = \Delta^T \Delta$. This indicates that in order to integrate the label information, we can simply replace X with

$X\Delta^T$, and rewrite Eq. (4.2) as follows:

$$\begin{aligned} &\arg\min_{U_t,M} \|X_s\Delta^T - (MU_t)(MU_t)^T X_s\Delta^T\|_F^2 + \|X_t - U_t U_t^T X_t\|_F^2 \\ &s.t. \quad U_t^T U_t = I \quad M^T M = I \end{aligned} \tag{4.10}$$

which can be also solved using Algorithm 3. Note that although we apply the same strategy as in supervised PCA to incorporate label information, our contribution is that we reformulate it in the subspace based domain adaptation framework.

4.4 Empirical Study

In this section, we present the empirical study of the proposed algorithm. The only parameter in our method is dimension K of the subspace U_t, which we set through cross-validation. In particular, for each K, we did leave-one-out training of a classier, and pick the one which gives the lowest prediction error on source data. In our experiments, K is usually equal or slightly large than the category numbers. We ran two versions of our algorithm: unsupervised version which does not use the label information of source data and supervised version. We compare to baseline methods as well as other state-of-the-art methods, including: Subspace alignment (SA) [32], Geodesic flow kernel (GFK) [10], and Transfer component analysis (TCA) [66]. We use linear SVM as classifiers in all experiments.

4.4.1 Handwritten Digit Recognition

In this section, we evaluate our method using a synthetic dataset. We use digit images of 3 and 8 from the USPS handwritten digit dataset to form a binary classification problem. We randomly sample a subset of images as source domain, and add rotation and blurring to the other images to generate target data. In particular, we select 100 images per class for training and 1000 images per class from the rest of images for test. We add rotation (45

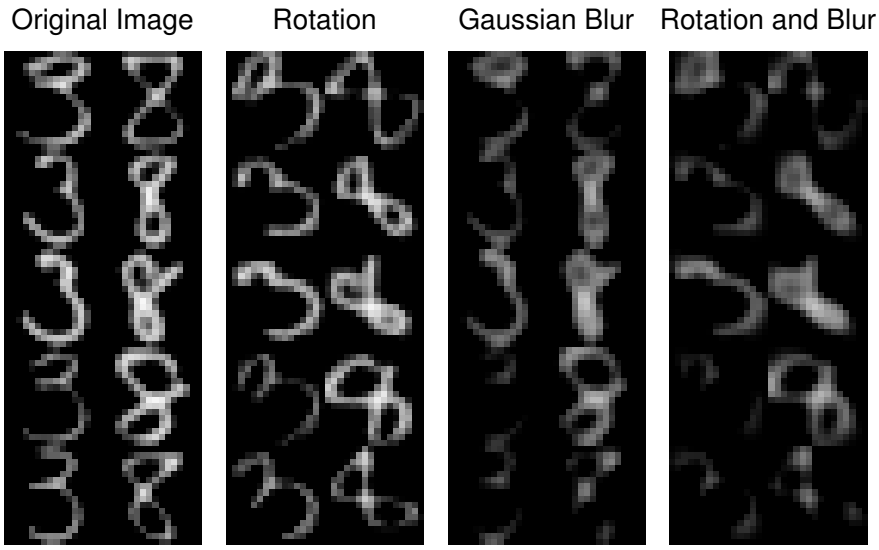

Figure 4.2: Examples of source and target data for handwritten digit recognition. We use original images as the source data and the rotated and blurred images as target data.

and 90 degree) and Gaussian blur (kernel width is 2) to the test images. Some examples are shown in Figure (5.2). We repeat for 20 times and report the mean and standard deviation of the recognition accuracies in Table (5.1). We set $K = 2$ in all the experiments on USPS handwritten digit datasets.

As shown by the results, our method is very robust to rotation and blur, even when the rotation degree is very large. It proves that by explicitly modeling the transformation between source and target data, our algorithm is able to find a better subspace jointly for both domains, which leads to a better predictive model for domain adaptation task.

4.4.2 Tumor Gene Expression Signatures for Cancer Diagnosis

In this section, we evaluate our algorithm on the 14 tumor data sets which were published by [80], and we downloaded them in the preprocessed version from [80]. The datasets contain 14 different human tumor types and 12 normal types. Each type of tumors have only an order of 10s of subjects and 15009 genes. We extract 3 types of tumors: Breast, Bladder and Lung and form 3 datasets by coupling normal and tumor samples from the same tissue type together. The sample size of each dataset is (20 vs 7) for Lung, (11 vs 7)

Table 4.2: Classification Accuracy on Cross-domain Tumor Gene Expression Signatures, where Lu is short for Lung, Bl is short for Bladderand, Br is short for Breast. The proposed method performs the best on all 6 pairs.

%	Lu-Bl	Lu-Br	Bl-Lu	Bl-Br	Br-Lu	Br-Bl
No Adaptation	74.89(4.38)	73.63(3.44)	70.44(0.52)	81.36(4.33)	78.96(3.92)	80.00(3.18)
TCA [66]	61.11(3.14)	77.27(5.54)	74.07(2.12)	77.27(2.98)	74.07(4.12)	61.11(3.76)
GFK [10]	77.22(5.29)	73.64(7.57)	74.07(2.12)	83.64(3.79)	79.92(2.50)	78.11(3.44)
SA[32]	82.00(3.09)	84.18(5.45)	70.14(2.89)	77.63(3.55)	76.15(5.46)	82.00(2.40)
Ours(Unsupervised)	82.11(2.82)	**90.09(2.37)**	76.37(1.82)	**87.00(3.18)**	80.59(3.22)	82.56(2.97)
Ours(Supervised)	**83.21(2.96)**	89.92(2.54)	**77.34(1.76)**	86.56(3.12)	**81.34(2.98)**	**83.24(3.12)**

for Bladder, (17 vs 5) for Breast. We perform experiments on 6 cross-domain tasks: Lung $\rightarrow$ Bladder, Lung $\rightarrow$ Breast, Bladder $\rightarrow$ Lung,Bladder $\rightarrow$ Breast, Breast $\rightarrow$ Lung, Breast $\rightarrow$ Bladder. To avoid over-fitting, we randomly select 500 features from the top 2000 highly correlated features in each task and repeat this procedure for 20 times. We then run baseline method, state-of-the-art method methods and our method on the reduced feature space. We report the mean and standard deviation of the classification accuracy for each method. We set $K = 2$ in all the settings on Tumor datasets.

As shown in Table 4.2, our method achieves consistently better classification accuracy on all 6 pairs of data. Especially, we achieve both higher average accuracy and lower standard deviation. A key advantage of our method is that it is robust to find common subspace even when there is only a few data samples in source and target domain.

We also show the convergence rate in Figure 4.3 for two tasks: Breast $\rightarrow$ Lung and Lung $\rightarrow$ Breast. The curves confirms that our iterative algorithm converges to a local minimum and we only need a few iterations.

4.4.3 Cross-Domain Text Classification

In this section, we evaluate the proposed approach on Reuters-21578 [1], a benchmark dataset for domain adaptation. There are 3 different domains in Reuters dataset: Orgs, People and Places. We perform experiments on 6 cross-domain tasks, i.e., Orgs $\rightarrow$ Places, Orgs

[1] www.cse.ust.hk/TL/index.html

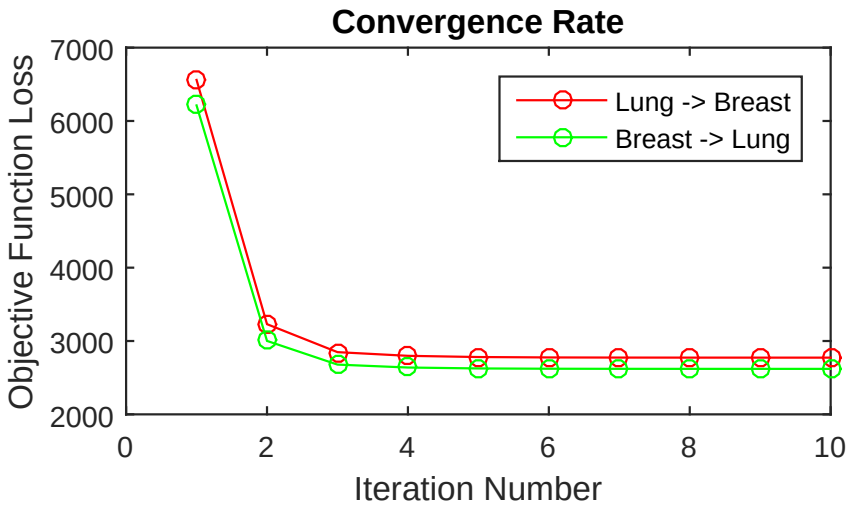

Figure 4.3: Convergence rate for subspace learning with data alignment on Lung → Breast and Breast → Lung tumor datasets. The curves shows that our iterative algorithm converges to a local minimum.

Table 4.3: Classification Accuracy on Cross-Domain Text Datasets, where Pe is short for People and Pl is short for Places. The proposed method performs the best on all 6 pairs.

%	Orgs-Pe	Orgs-Pl	Pl-Pe	Pe-Orgs	Pl-Orgs	Pe-Pl
No Adaptation	66.81(2.25)	61.52(2.71)	53.45(2.21)	71.40(2.75)	65.67(1.81)	55.63(1.74)
TCA [66]	65.23(4.95)	63.48(3.86)	55.34(2.65)	69.89(2.13)	66.54(2.67)	56.98(1.95)
GFK [10]	72.51(1.35)	66.32(3.02)	56.47(3.16)	77.82(1.80)	65.24(2.65)	57.36(1.93)
SA[32]	62.52(6.79)	65.06(6.61)	55.49(7.73)	64.16(3.46)	66.74(11.9)	57.19(4.23)
Ours(Unsupervised)	72.69(1.57)	70.20(1.92)	57.86(1.92)	77.34(1.32)	**71.31(2.82)**	61.75(1.60)
Ours(Supervised)	**74.78(1.41)**	**70.51(2.61)**	**58.78(1.37)**	**79.29(2.16)**	67.55(2.50)	**62.50(2.13)**

→ People, People → Orgs, People → Places, Places → People, Places → People. We randomly select 30% samples from the source domain and test on all the samples from the target domain. We repeat 20 times for each task and report the mean and standard deviation of the classification accuracy in Table 4.3. Our method achieves highest accuracy compared to the state-of-the-art methods. We also observe that the supervised version of our algorithm outperforms the unsupervised version in most cases, which indicates that by incorporating the label information of the source data, the subspace is better for prediction.

4.5 Conclusions

In this paper, we propose a novel linear subspace learning approach for domain adaptation. Our method explicitly aligns the data in two domains using a linear transformation while simultaneously finding a subspace which preserves the most data variance. With explicit data alignment, the subspace learning is formulated as minimizing a PCA-like objective, which consists of two variables: the basis vectors of the common subspace and the linear transformation between two domains. We show that the optimization can be solved efficiently using an iterative algorithm based on alternating minimization, and prove its convergence to a local optimum. Our method can also integrate the label information of source data, which further improves the robustness of the subspace learning and often yields better prediction. We apply our method to benchmark datasets and obtain very competitive results that outperform state-of-the-art methods.

Chapter 5

Latent Subspace Discovery via Subspace Clustering for Domain Adaptation

5.1 Introduction

A typical assumption for supervised methods is that the training (source) and test (target) data are from the same distribution. However, this assumption is not satisfied in many real world applications. In order to build robust prediction models, the discrepancy between training and test data, which is often referred as *domain shift*, needs to be taken into consideration. This issue is known as *domain adaptation (DA)*. In this paper, we have data sets on which the class labels are known, which are called source domain. For a new data set, or a target domain, we aim to find the ground truth labels by exploring the information from the source domain.

Domain adaptation has been actively studied in recent years [54, 48, 23, 66]. Subspace based domain adaptation gains a lot of popularity due to its promising results on many real world applications, such as computer vision [10, 32, 78, 38] and natural language processing [9, 8, 67]. The key idea is to find a common subspace in which source and target data share similar distribution, so that the features used to build the prediction model in source domain also have support in target domain. All the approaches above assume that either source or target data lie in a ***single*** low dimensional subspace. We refer to this assumption as single subspace assumption.

We would like to make an argument that single subspace assumption is too strong in many applications, especially considering the domain could be a mixture of latent domains with significant inner-domain variations that should not be neglected [47, 37, 91] or data is from multiple sources [62, 82, 27, 97]. The key observation we make in this work is: data often lie in an ***union*** of ***multiple*** low dimensional subspaces. For example, considering an object recognition task where the training images of objects are also significantly affected by many extraneous factors, such as illumination, view angle, camera resolution in addition to intra-class appearance variations, This observation motivates us to explicitly discover and exploit multiple subspaces. To the best of our knowledge, this insight has never been considered in any existing subspace based domain adaptation approaches. We

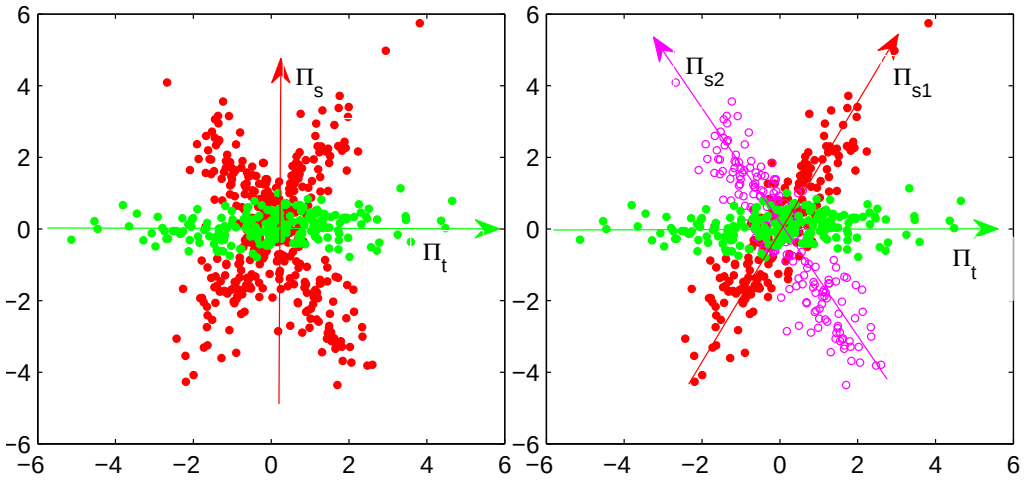

Figure 5.1: A toy example to demonstrate our motivation for latent subspace discovery with 2-dimensional data. Left: red dots are source data which lie in two 1-dimensional subspaces, and green dots are target data. If we follow single subspace assumption, and apply PCA to all source data, we get a principal component Π_s which is orthogonal to that of the target data Π_t. In this case, subspace based DA algorithms, such as SA, may not work well. Right: We identify that the source data lie in an union of two 1-dimensional subspaces, and compute the principal components of the two subspace as Π_{s1} and Π_{s2} respectively. Note that they are no longer orthogonal to Π_t, and each of them independently preserves the data variance of two source clusters. Therefore, each source subspace can be well adapted to the target subspace.

use a toy example to better illustrate our motivation in Fig. 5.1. In fact, the assumption that data in high dimensional space can be embedded into a union of low dimensional subspace has been validated in many applications. This observation motivates the research of *Subspace Clustering* algorithms. Given a set of data samples, the goal of a subspace clustering algorithm is to find the number of subspaces and the basis for each subspace. Subspace clustering algorithms have been successfully applied to solve many computer vision applications, such as face recognition and image segmentation [30, 69, 53]. As mentioned above, our key contribution is to utilize subspace clustering in domain adaptation, and demonstrate that it can significantly improve the performance of subspace based domain adaptation methods.

On a high level view, our approach consists of two steps. In the first step, we identify a union of subspaces for source and target data separately, and assign each data sample to one of the subspaces. This is in fact a subspace clustering problem. In particular, we adopt Sparse Subspace Clustering algorithm [30] in the subspace discovery step due to its robustness to data noise and outliers. In the second step, given source subspaces together with the target subspaces, we propose a novel multiple subspace alignment (Multi-SA) algorithm, in which we identify one common subspace that aligns well with both source and target subspaces, and therefore, best preserves the variance for both domains. To solve this alignment problem jointly for multiple subspaces, we formulate this problem as solving an optimization problem that minimizes the weighted sum of multiple alignment costs. A higher weight is assigned to a source subspace if its label distribution has smaller distance, measured by KL divergence, compared to the overall label distribution. By putting more weights on those subspaces, the learned common subspace is able to to preserve the distinctive information. Finally, we use all the transformed source data and their labels to train a single predictor and predict on the transformed target data.

To summarize, our contribution is twofold: 1. We point out the limitation of the single subspace assumption adopted by the existing subspace based domain adaptation algorithms, and instead assume that data lie in a union of multiple subspaces, which can be discovered by a subspace clustering algorithm. 2. We propose a novel multiple subspace alignment (Multi-SA) algorithm, which aims to find a good subspace by not only maximizing the variance of both source and target domain but also preserving the distinctive information in the source domain. We examine our approach on traditional domain adaptation tasks where there is only one domain, as well as tasks where data are drawn from multiple latent domains. We demonstrate that the proposed method achieves favorable results compared to other state-of-the-art methods.

The rest of the paper is organized as follows: we first discuss related works which also discover latent domains in order to build a more robust domain adaptive predictor. We then

give the details about our Multiple Subspace Alignment (Multi-SA) algorithm. After that, we report the experimental results.

5.2 Related Work

Multi-source domain methods [62, 27, 82, 97] assume that multiple source data are given as input, and the predictor must be adapted from them to do well in testing on target domain. Recently, [47] and [37] relax this assumption and assume that the domain label of the source data is not available for training. Both methods explicitly address the issue that source data may consist of multiple latent domains. In order to obtain the optimal domain invariant predictor, source data samples need to be first clustered into different groups, each of which corresponds to a latent domain. Hoffman et al. [47] use hierarchical clustering algorithm to group the source data samples. Based on the key insight that each feasible domain should contain multiple object categories, they use the available object category information to constrain the hierarchical clustering process. Gong et al. [37] propose two key criteria, maximum distinctiveness and maximum learn-ability, to cluster source samples into latent domains. They use a kernel-based approach [39] to measure the distribution difference. They formulate the domain discovery problem as solving an optimization problem which maximizes the distribution difference while enforcing label prior constraint (similar to [47]).

Our work share some common spirits with [47] and [37], as we also address similar issue that source data may be drawn from multiple latent distributions. One perspective of [47] and [37], as well as our work, is that they all solve a domain clustering problem. While [47] used k-means objective and [37] used maximum distinctiveness criteria based on kernel-mean, our approach is based on subspace clustering. Our insight ties closely to the assumption that intrinsic low dimensional subspace can be used to characterize domains. When source data are drawn from multiple latent distributions, we assume data lie

in a union of low dimensional subspaces. For data lying in the same subspace, they are *self-expressive* [30], meaning that each data sample can be reconstructed as a linear combination of other data samples in the same cluster. Therefore, the characteristic of clusters obtained by subspace clustering is fundamentally different from those in [47] and [37]. Subspace clustering also fits naturally with subspace based domain adaptation methods.

Our work is also closely related to single domain Subspace Alignment (SA) [32], which learns a linear transformation to align the source subspace coordinate system to the target one. The most important difference comparing our method to SA is that we discover and exploit multiple latent subspaces of data, while SA only identify a single subspace for source and target data respectively. In addition, we proposed a novel algorithm to align multiple source and target subspace to a common subspace, in which the variance of source and target data and distinctive information in the source domain can be well preserved.

5.3 Subspace Discovery and Domain Adaptation

We aim to solve the domain adaptation in the standard unsupervised setting. We assume that we have access to an annotated source datasets denoted by $\{(x_s, y_s)\}$, where $x_s \in R^P$ is the feature vector and $y_s \in \{1, 2, \cdots, C\}$ is the corresponding label out of C categories. The target dataset is denoted by $\{(x_t, ?)\}$ where $x_t \in R^P$, and the label is unknown. We assume that source data $\{x_s\}$ lies in an union of L_S subspaces. For the convenience of notation, we refer to the sth subspace using its projection matrix $\Pi_s \in R^{P \times K}$, where K is smaller than P. Each column in Π_s corresponds to a basis vector of the subspace. Basis vectors are orthogonal to each other, therefore the dimension of subspace Π_s is K. Each data sample in this subspace can be represented by a linear combination of the basis vectors. Similarly, we assume that the target data $\{x_t\}$ lies in L_T subspace, whose projection matrices are denoted by $\{\Pi_t\}$.

We first describe how to infer $\{\Pi_s\}$ and $\{\Pi_t\}$ from the source and target data. After that, we describe the proposed Multiple Subspaces Alignment (Multi-SA) algorithm by assuming that we know the subspaces for source and target data and their projection matrices $\{\Pi_s\}$ and $\{\Pi_t\}$.

5.3.1 Subspace Discovery via Sparse Subspace Clustering

Given a set of samples $\{x\}$, estimating the union of low-dimensional subspaces and their basis vectors $\{\Pi_l\}$ is a *Subspace Clustering* problem [2]. As solving subspace clustering problem is critical in many practical problems, there exists many subspace clustering algorithms, such as [69, 30]. In our work, we choose Sparse Subspace Clustering (SSC) algorithm [30] due to its robustness to data noise and outliers. For the completeness of this paper, we briefly describe the SSC algorithm. The main idea of SSC algorithm is to exploit the *self-expressiveness* property of the data, which states that each data point in a union of subspaces can be efficiently represented as a linear or affine combination of other points. The sparse representation of a data point ideally corresponds to a combination of a few points from its own subspace. If we let $X \in R^{P\times N}$ denote the matrix containing all data points, let $W \in R^{N\times N}$ denote the coefficient matrix, and let $E \in R^{P\times N}$ be the error matrix, such problem can be solved through sparse optimization:

$$\min_{W} ||W||_1 + \lambda||E||_1 s.t.\ X = XW + E, diag(W) = 0 \tag{5.1}$$

The coefficients are used to construct a neighborhood graph, in which the nodes represent data points, and the edge weights matrix is calculated as $|W| + |W|^T$. Each edge weight indicates how likely that pair of nodes lie in the same subspace. A spectral clustering method is then used to infer the cluster of the data. Finally, for each cluster, PCA is used to compute the basis vectors $\{\Pi_l\}$ which preserve the most variation of the data in each cluster. For source data $\{x_s\}$ and target data $\{x_t\}$, we use the above method to get the

Algorithm 3: Sparse Subspace Clustering Algorithm

Input : A set of data samples $\{x\}$, number of subspaces L, dimension of each subspace K.

Output: Subspace basis vectors $\{\Pi\}_{l=1}^{L}$ with $\Pi \in R^{P\times K}$.

1. Obtain the coefficients matrix W by solving Eq. (5.1).
2. Column-wise normalization of coefficients matrix W.
3. Form a similarity graph with N nodes representing the source data points, set the weights on the edges between the nodes by $W = |W| + |W|^T$.
4. Apply spectral clustering to the similarity graph to obtain L clusters.
5. Apply PCA to each cluster of data, compute the first K eigenvectors, each of the eigenvector is a column in the projection matrix Π.

basis vectors $\{\Pi_s\}$ for source subspaces and $\{\Pi_t\}$ for target subspaces separately. More details about SSC algorithm can be found in [30]. We summarize the SSC algorithm in Alg. (3).

5.3.2 Multiple Subspaces Alignment for Domain Adaptation

Given the data $\{x\}$ and the corresponding set of subspaces $\{\Pi_l\}$, we first describe how to assign each data sample to a unique subspace. Optimally, we consider a data sample x lies in a subspace Π_l only if x can be perfectly represented as a linear combination of the basis vectors of Π_l. However, this assumption rarely happens when the dimension of the subspace is lower than that of the original feature space. Instead, for each data sample, we aim to find its closest subspace from the union of L source subspaces. After we project data sample x into the subspace Π_l, the closeness is then characterized by the distance between original data sample x and its reconstructed representation $\Pi_l\Pi_l^T x$. The distance is often referred as reconstruction error computed as $\|x - \Pi_l\Pi_l^T x\|^2$. A lower projection error indicates that the data sample is closer to the subspace. Therefore, we obtain the closest subspace for each data sample by:

$$l^* = \underset{l=1,2,\cdots,L}{\arg\min} \|x - \Pi_l\Pi_l^T x\|^2 \tag{5.2}$$

With Eq. (5.2), we can identify the closest subspace Π_{s^*} for each source data sample x_s and the closest suspace Π_{t^*} for each target data sample x_t. We transform x_s into a lower dimensional representation $\tilde{x_s} \in R^K$ with $\Pi_{s^*}^T x_s$. According to the common assumption that data lie in an intrinsic low dimensional subspace, little predictive information is lost comparing to x_s. Similarly, a target data sample x_t can be transformed to $\tilde{x_t} = \Pi_{t^*}^T x_t$.

However, note that Π_{s^*} and Π_{t^*} contain different basis vectors, and the new representations for source data and target data are in two different coordinate systems. Therefore, the predictor trained on $\{(\tilde{x_s}, y_s)\}$ cannot be directly applied to $\tilde{x_t}$. For example, if we train a linear SVM classifier, it is obvious that the weights on the directions in Π_{s^*} can not be used as the weights on the directions in Π_{t^*}. Also, if two source data samples lie in two different subspaces, they will be transformed with different Π_{s^*}, which means that their corresponding $\tilde{x_s}$ will also be in different coordinate systems. The transformed target data may also lie in two different coordinate systems. We present how to address these three issues in the proposed Multi-SA algorithm.

We aims to resolve the discrepancy between source and target subspaces. The key idea is to learn multiple linear transformations to align both the source and target subspace coordinate systems to an unknown common coordinate system with basis vectors denoted as Π. We use $\{M_s\}, \{M_t\} \in R^{K \times K}$ to denote the set of linear transformations, where K is the number of basis vectors in each source and target subspace. We aim to find the optimal $\{M_s\}$ and $\{M_t\}$, which make $\{\Pi_s M_s\}$ and $\{\Pi_t M_t\}$ to be as close as possible to Π. Our objective function is to minimize the weighted sum of the alignment costs between source/target subspaces to the common latent unknown subspace Π, and is formally defined as:

$$\underset{\{M_s\},\{M_t\},\Pi}{\arg\min} \sum_{s=1}^{L_S} \lambda_s \|\Pi - \Pi_s M_s\|_F^2 + \sum_{t=1}^{L_T} \|\Pi - \Pi_t M_t\|_F^2 \tag{5.3}$$

where $\{\lambda_s\}$ are the weights to control the relative contribution of alignment costs to the overall objective.

WEIGHTS SETTING: Since we seek a union of $\{\Pi_s\}$ for the source domain, the key question is how to distinguish a **good** subspace from a **bad** subspace. Our insight is that the class labels in a **good** subspace should be distributed similar to the prior distribution (of the labels), estimated empirically from the whole source data. It only reflects the intuition that in the process of data collection, the relative percentages of different classes are approximately in accordance with a prior distribution that is independent of domains. Thus, when data samples are re-arranged into latent subspaces, the same percentages are likely to be preserved in each latent subspace. We denote the prior label distribution as $p(y)$, which is estimated empirically for all source data and label distribution for the sth subspaces as $p_s(y)$. We measure the difference between two probability distributions $p(y)$ and $p_s(y)$ with KL divergence:

$$KL(p_s(y)\|p(y)) = \sum_i p_s(y_i) log(\frac{p_s(y_i)}{p(y_i)}) \tag{5.4}$$

In order to make contributions of **good** subspaces more significant to the overall objective, we set λ_s as:

$$\lambda_s = \frac{C}{1 + KL(p_s(y)\|p(y))} \tag{5.5}$$

where C is a constant that controls the relative importance of source and target domain. We set C to make sure that the sum of weignts in target domain is $3 \sim 5$ times of the sum of weights in source domain.

Note that we assign the same weights to different subspaces in target domain in Eq. (5.3), since we are not able to evaluate the importance of subspaces without label information.

ALTERNATIVE OPTIMIZATION: To optimize Eq. (5.3), we adopt an alternating minimization approach. At each step, we alternatively optimize over Π, $\{M_s\}$ and $\{M_t\}$ with the others fixed. The details are described below.

Initialization: initialize $\{M_s\}$ and $\{M_t\}$ to identity matrices.

Algorithm 4: Multiple Subspace Alignment for Domain Adaptation

Input : Source subspaces $\{\Pi_s\}_{s=1}^{L_S}$ and target subspaces $\{\Pi_t\}_{t=1}^{L_T}$, source data $\{(x_s, y_s)\}$, target data $\{(x_t, ?)\}$.

Output: The predicted labels $\{y_t\}$ for the target data.

Obtain $\{M_s\}$, $\{M_t\}$, and Π by optimizing Eq. (5.3).
`/* Transform the coordinates of` x_s `to` Π`. */`
for *every* x_s **do**
    Find the closest subspace s^* for x_s with Eq. (5.2).
    $\tilde{x_s} = M_{s^*}^T \Pi_{s^*}^T x_s$.
end
`/* Transform the coordinates of` x_t `to` Π`. */`
for *every* x_t **do**
    Find the closest subspace t^* for x_t with Eq. (5.2)
    $\tilde{x_t} = M_{t^*}^T \Pi_{t^*}^T x_t$.
end
Train a linear SVM model using $(\tilde{x_s}, y_s)$.
Apply the trained model on $\tilde{x_t}$ to obtain the predicted label y_t.

Optimize Π with fixed $\{M_s\}$ and $\{M_t\}$: If $\{M_s\}$ and $\{M_t\}$ are constant, we compute the derivative of Eq. (5.3) with respect to Π and set it to 0 and obtain:

$$\Pi^* = \frac{\sum_{s=1}^{L_S} \lambda_s \Pi_s M_s + \sum_{t=1}^{L_T} \Pi_t M_t}{\sum_{s=1}^{L_S} \lambda_s + L_T} \tag{5.6}$$

Optimize $\{M_s\}$ and $\{M_t\}$ with fixed Π: If Π is fixed, solving Eq. (5.3) turns into solving $L_S + L_T$ separate sub-problems:

$$\arg\min_{M_l} ||\Pi - \Pi_l M_l||_F^2 \tag{5.7}$$

Eq. (5.7) is a least square problem. Given that the pseudo inverse of Π_l equals to $(\Pi_l^T \Pi_l)^{-1} \Pi_l^T$, the solution of M is $M_l^* = \Pi_l^T \Pi$.

Because each step above reduces the objective and the objective is a function with a lower bound 0, our algorithm will converge to a local minimum. We summarize our proposed Multiple Subspace Alignment algorithm in Alg. (4).

Table 5.1: Classification Accuracy on Synthetic Multiple Source Data for USPS Digit Recognition.

$\mathcal{S}$	(3,8) at 30°	(3,8) at 60°	(3,8) at 30°,60°	(1,7) at 30°	(1,7) at 60°	(1,7) at 30°, 60°
$\mathcal{T}$	(3,8) at 0°	(3,8) at 0°	(3,8) at 0°	(1,7) at 0°	(1,7) at 0°	(1,7) at 0°
NO ADAPTATION	79.33(7.07)	74.58(8.19)	76.18(5.92)	84.95(16.92)	75.32(20.62)	82.72(16.84)
SA [32]	84.55(8.35)	78.87(14.45)	81.52(19.12)	95.62(3.48)	94.60(3.67)	85.94(16.71)
OURS	**90.11(3.47)**	**90.36(2.80)**	**91.53(1.57)**	**95.96(3.83)**	**95.23(1.57)**	**96.50(2.06)**

Table 5.2: Classification Accuracy for Single Domain Adaptation with Multiple Latent Domain Discovery. A: Amazon, D: DSLR, W: Webcam, C:Caltech-256, Pe: People, Pl: Places, Or: Orgs

$\mathcal{S}$	A	A	A	C	C	C	W	W	W	Pe	Pl	Or	Pe	Pl	Or
$\mathcal{T}$	D	C	W	W	D	A	A	C	D	Or	Pe	Pe	Pl	Or	Pl
NO ADAPTATION	36.1	37.3	40.5	34.6	38.9	44.3	32.9	28.4	73.6	70.7	53.6	64.7	50.5	66.3	64.8
[66]	36.3	35.0	27.8	32.5	45.2	41.4	24.2	28.7	75.7	69.9	55.3	65.2	57.0	66.5	63.4
[10]	37.9	38.3	39.8	34.9	36.1	44.8	37.1	29.1	74.6	75.8	56.5	72.5	57.4	65.2	66.3
[32]	38.8	39.9	39.6	38.9	39.4	46.1	**39.3**	31.8	77.9	64.1	55.5	62.5	57.2	68.3	61.0
OURS	**44.6**	**41.3**	**40.6**	**41.0**	**41.5**	**49.0**	36.8	**32.5**	**79.8**	**76.4**	**59.0**	**77.6**	**60.1**	**70.5**	**70.5**

5.3.3 Parameters Settings

There are three parameters in our algorithm. One parameter is the dimension of each subspace K, Similar to [10, 32], we set K through cross validation on source data, it is usually equal or slightly larger than the number of categories. The other parameter is the number of subspaces in target domain L_T. In the applications that we know the number of domains of the testing data, we can set L_T accordingly. Otherwise, because we have no label information to set weights of the target subspaces, we always assign a relatively small value to L_T. Meanwhile, our algorithm is very robust to the number of subspaces in source domain L_S, as we can set the informative weights using label information. When L_S is increased and there are more source clusters, many small clusters with no discriminative information will be disabled in the objective function. In our experiments, we set L_S through cross-validation in which we randomly divide the training data into two halves, and use one halve as source and the other halve as target.

Table 5.3: Classification Accuracy with Domain Discovery on Multiple Domain Data for Visual Object Recognition

$\mathcal{S}$	A,C	D,W	C,D,W	W,C
$\mathcal{T}$	D,W	A,C	A	A
NO ADAPTATION	41.7	35.8	41.0	51.1
[32]	34.5	35.4	45.1	47.7
[47]	39.6	34.4	38.9	48.9
[37]	42.6	35.5	44.6	49.2
OURS	**49.7**	**38.5**	**52.8**	**53.8**

5.4 Empirical Study

We consider two types of domain adaptation tasks: a) traditional single domain adaptation with one labeled source dataset and one unlabeled target dataset, b) domain adaptation where source and target dataset are a mixture of multiple datasets. For both tasks, we validate the effectiveness of our approach through extensive experiments on benchmark data. We also visualize some intermediate results in order to provide some insights into our approach.

5.4.1 Synthetic USPS Handwritten Digits Recognition

In this section, we evaluate the proposed method on digit recognition task with synthetic data. We select two pairs of digits from USPS handwritten digit datasets [1] to form binary classification problems, which are 1 vs 7 and 3 vs 8. In each binary classification task, we randomly sample 100 images for training, and randomly sample 1000 images from the rest for testing. In order to create domain discrepancy, we apply rotation transformation to training images. In the single domain adaptation task, we rotate every training sample by the same degree (we conduct two experiments using 30 and 60 respectively). To simulate the domain adaptation tasks with multiple latent source domains, we double the number of training samples by rotating each training sample by both 30 and 60 degrees. We use raw pixel intensities as features. We run each experiment for 20 times, and report the mean

[1] http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/multiclass.html#usps

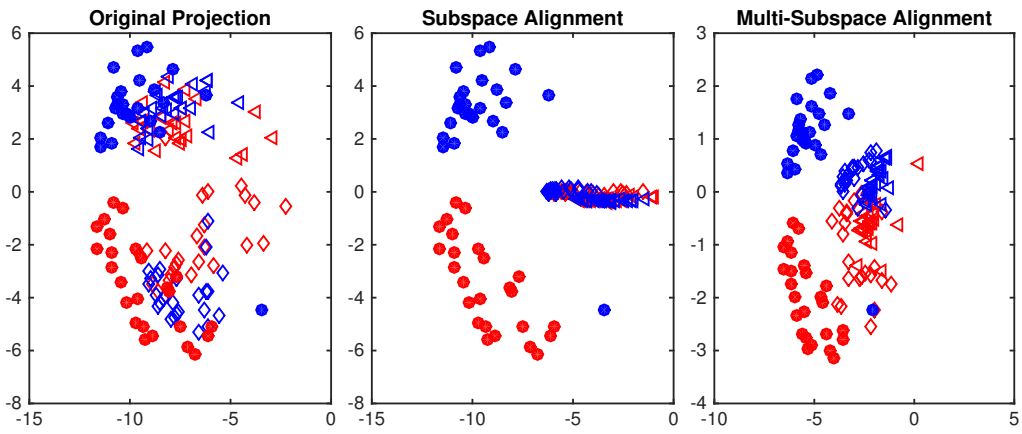

Figure 5.2: Visualization of multiple subspace alignment for multiple latent domain data . We rotate the training samples by both 30 and 60 degrees. In order to visualize the data representation with no adaptation, we use PCA to project both training and testing data to a 2D space. We set $K = 2$ in both SA and our method. Solid shapes represent the test data, shallow shapes represent the training data (different shapes represent different rotation degrees). Different color represent different classes. Our approach not only blends the source and target data, but also does well in separating the data of different classes.

classification accuracy of linear SVM and standard deviation in Table 5.1. In all the tasks, we set $L_S = 10$, $L_T = 1$ and $K = 2$.

For two single domain adaptation tasks, which satisfy the single subspace assumption, both SA and our method can improve the recognition accuracy compared to the baseline approach with no adaptation. However, SA suffers when there exits multiple latent source domains and the single subspace assumption no longer holds. Our approach significantly outperforms both SA and the baseline approach. Our approach also achieves better recognition results when using multiple domains, compared to using single domain. We visualize the adapted data representation $\tilde{x_s}$ and $\tilde{x_t}$ in Fig.5.2.

5.4.2 Single Domain Adaptation for Visual Object Recognition and Text Classification

In this section, we examine our method on single domain adaptation tasks for both visual object recognition and text classification.

For visual object recognition, we use the benchmark domain adaptation dataset introduced in [10], which consists of 4 datasets: Amazon, Webcam, DSLR and Caltech-256, each of which contains labeled images of 10 different objects. We conduct 9 experiments. In each experiment, we pick one source dataset and one target dataset and solve the domain adaptation problem. We downloaded the 800-bin bag-of-word image representation used in [10, 32].

For text classification, we conduct 6 experiments on Reuters-21578 [2], which consists of 3 domains: Orgs, People and Places.

We compare our approach to the state-of-the-art methods: Transfer Component Analysis (TCA) [66], Geodesic Flow Kernel (GFK) [10] and Subspace Alignment (SA) [32]. We report the classification accuracy in Table 5.2. For visual recognition tasks, results are quoted from [32]. Our method achieves the best results on 14 out of 15 tasks. This clearly demonstrates that the domain adaptation performance will benefit from our multiple subspaces assumption, even for single source datasets with intra domain variations.

In the next section, we examine the performance of our approach on tasks in which source dataset is indeed a mixture of multiple datasets.

5.4.3 Domain Adaptation with Multiple Domains for Visual Object Recognition

For visual object recognition, we use the benchmark domain adaptation dataset introduced in [10], which consists of 4 datasets: Amazon, Webcam, DSLR and Caltech-256, each of which contains labeled images of 10 different objects. We downloaded the 800-bin bag-of-word image representation used in [10, 32]. We conduct the same experiments as [37], in which the source dataset is a mixture of multiple datasets and the target dataset is either one dataset, or a mixture of multiple datasets.

[2] http://www.cse.ust.hk/TL/index.html

We report recognition accuracies for this domain adaptation task in Table 5.3. Our method performs significantly better than the baseline approach SA [32]. Our method also achieves the best results compared to other state-of-the-art methods.

5.5 Conclusion

In this work, we propose a novel subspace based domain adaptation algorithm. In contrast to the common single subspace assumption made by existing methods. We assume that data lie in a union of low dimensional subspaces. In our approach, we first use a subspace clustering algorithm to identify multiple subspaces of data. We propose a multiple subspaces alignment(Multi-SA) algorithm. Our goal is to find one common subspace that preserves the variance for both source and target data and then align all source data and target data to the common subspace. We extensively evaluate our method on many domain adaptation tasks. Our method achieves favorable results compared to other state-of-the-art methods, which clearly demonstrates its effectiveness.

Chapter 6

Conclusions and Future Directions

Traditional machine learning methods usually assumes that the training and test data are draw from the same distribution. However, this assumption does not always hold in many practical problems. In this dissertation, we focus on utilizing data come from a different but closely related distribution as that of the test data, to aid learning the model. We are concerning two learning scenarios which are different from supervised learning, unsupervised learning and semi-supervised learning: 1) domain adaptation 2) learning from multiple domains or multiple latent domains.

The proposed algorithms can be divided into two groups based on the basic building blocks.

Graph Learning for Domain Adaptation: In Chapter 2, we explore the locality preserving projection for domain adaptation with multi-objective learning. We propose multi-objective formulation for domain adaptation. The search space of our objective is the joint numerical range of two graphs. We find a relaxed mutually orthogonal optimal sets by using Pareto optimizations. In Chapter 3, we demonstrate that affinity learning can be a very successful tool for domain adaptation. Our approach is able to learn the joint geometric structure of source and target domain based on the preservation of intra-domain and across domain information.

Subspace Learning for Domain Adaptation In Chapter 4, we propose a novel linear subspace learning approach for domain adaptation. Our method explicitly aligns the data in two domains using a linear transformation while simultaneously finding a subspace which preserves the most data variance. In Chapter 5, we assume that data lie in a union of low dimensional subspaces in contrast to the common single subspace assumption made by existing methods. In our approach, we first use a subspace clustering algorithm to identify multiple subspaces of data. We propose a multiple subspaces alignment(Multi-SA) algorithm to find one common subspace that preserves the variance for both source and target data and then align all source data and target data to the common subspace.

As future works, we would like to explore the work in different settings and related tasks. At first, we would like to solve heterogeneous domain adaptation aims to exploit labeled training data from a source domain for learning prediction models in a target domain under the condition that the two domains have different input feature representation spaces. Second, we would like to explore when all marginal and conditional distributions are allowed to change with smooth changes. Third, we would like to explore efficient domain adaptation algorithms for large scale applications.

Bibliography

[1] Tameem Adel and Alexander Wong. A probabilistic covariate shift assumption for domain adaptation. In *AAAI*, pages 2476–2482, 2015.

[2] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *SIGMOD*, pages 94–105, 1998.

[3] Elnaz Barshan, Ali Ghodsi, Zohreh Azimifar, and Mansoor Zolghadri Jahromi. Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, 44(7):1357–1371, 2011.

[4] Mikhail Belkin and Partha Niyogi. Semi-supervised learning on riemannian manifolds. *Machine learning*, 56(1-3):209–239, 2004.

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

[6] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *ICML*, pages 81–88, 2007.

[7] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007.

[8] John Blitzer, Sham Kakade, and Dean P. Foster. Domain adaptation with coupled subspaces. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 173–181, 2011.

[9] John Blitzer, Ryan T. McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *EMNLP*, pages 120–128, 2006.

[10] Boqing, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, pages 2066–2073, 2012.

[11] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.*, 2(2):121–167, 1998.

[12] Deng Cai, Xiaofei He, Wei Vivian Zhang, and Jiawei Han. Regularized locality preserving indexing via spectral regression. In *CIKM*, pages 741–750, 2007.

[13] Marine Carpuat, Hal Daumé III, Katharine Henry, Ann Irvine, Jagadeesh Jagarlamudi, and Rachel Rudinger. Sensespotting: Never let your parallel data tie you to an old domain. In *ACL*, pages 1435–1445, 2013.

[14] Olivier Chapelle, Bernhard Schölkopf, Alexander Zien, et al. Semi-supervised learning.

[15] Rita Chattopadhyay, Qian Sun, Wei Fan, Ian Davidson, Sethuraman Panchanathan, and Jieping Ye. Multisource domain adaptation and its application to early detection of fatigue. *TKDD*, 6(4):18, 2012.

[16] Minmin Chen, Kilian Q. Weinberger, and John Blitzer. Co-training for domain adaptation. In *NIPS*, pages 2456–2464, 2011.

[17] Minmin Chen, Zhixiang Eddie Xu, Kilian Q. Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *ICML*, 2012.

[18] Corinna Cortes and Mehryar Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theor. Comput. Sci.*, 2014.

[19] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theor.*, 13(1):21–27, September 2006.

[20] Koby Crammer, Michael Kearns, and Jennifer Wortman. Learning from multiple sources. *Journal of Machine Learning Research*, 9:1757–1774, 2008.

[21] Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. Transferring naive bayes classifiers for text classification. In *AAAI*, pages 540–545, 2007.

[22] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for transfer learning. In *ICML*, pages 193–200, 2007.

[23] Hal Daumé III. Frustratingly easy domain adaptation. *CoRR*, abs/0907.1815, 2009.

[24] Ian Davidson, Buyue Qian, Xiang Wang, and Jieping Ye. Multi-objective multi-view spectral clustering via pareto optimization. In *SDM*, pages 234–242, 2013.

[25] James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[26] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, pages 647–655, 2014.

[27] Lixin Duan, Ivor W. Tsang, Dong Xu, and Tat-Seng Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *ICML*, pages 289–296, 2009.

[28] Lixin Duan, Dong Xu, and Shih-Fu Chang. Exploiting web images for event recognition in consumer videos: a multiple source domain adaptation approach. In *ICML*, pages 1338–1345, 2012.

[29] Lixin Duan, Dong Xu, and Ivor Wai-Hung Tsang. Domain adaptation from multiple sources: A domain-dependent regularization approach. *IEEE Trans. Neural Netw. Learning Syst.*, 23(3):504–518, 2012.

[30] Ehsan Elhamifar and René Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2765–2781, 2013.

[31] Wei Fan, Ian Davidson, Bianca Zadrozny, and Philip S. Yu. An improved categorization of classifier's sensitivity on sample selection bias. In *ICDM*, pages 605–608, 2005.

[32] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, pages 2960–2967, 2013.

[33] Yanwei Fu, Timothy M. Hospedales, Tao Xiang, Zhen-Yong Fu, and Shaogang Gong. Transductive multi-view embedding for zero-shot recognition and annotation. In *ECCV*, pages 584–599, 2014.

[34] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, pages 513–520, 2011.

[35] Gene H. Golub and Charles F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[36] Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, pages 222–230, 2013.

[37] Boqing Gong, Kristen Grauman, and Fei Sha. Reshaping visual datasets for domain adaptation. In *NIPS*, pages 1286–1294, 2013.

[38] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, pages 999–1006, 2011.

[39] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. A kernel method for the two-sample-problem. In *NIPS*, pages 513–520, 2006.

[40] Arthur Gretton, Olivier Bousquet, Alexander J. Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *ALT*, pages 63–77, 2005.

[41] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[42] Quanquan Gu, Marina Danilevsky, Zhenhui Li, and Jiawei Han. Locality preserving feature learning. In *AISTATS*, pages 477–485, 2012.

[43] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Unsupervised learning*. Springer, 2009.

[44] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In *NIPS*, 2005.

[45] Xiaofei He and Partha Niyogi. Locality preserving projections. In *NIPS*. 2003.

[46] Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and HongJiang Zhang. Face recognition using laplacianfaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(3):328–340, 2005.

[47] Judy Hoffman, Brian Kulis, Trevor Darrell, and Kate Saenko. Discovering latent domains for multisource domain adaptation. In *CVPR*, pages 702–715, 2012.

[48] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, pages 601–608, 2006.

[49] J. J. Hull. A database for handwritten text recognition research. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(5):550–554, May 1994.

[50] Hal Daumé III. Frustratingly easy domain adaptation. In *ACL*, 2007.

[51] Hal Daumé III and Jagadeesh Jagarlamudi. Domain adaptation for machine translation by mining unseen words. In *ACL*, pages 407–412, 2011.

[52] Hal Daumé III, Abhishek Kumar, and Avishek Saha. Co-regularization based semi-supervised domain adaptation. In *NIPS*, pages 478–486, 2010.

[53] Pan Ji, Mathieu Salzmann, and Hongdong Li. Efficient dense subspace clustering. In *IEEE Winter Conference on Applications of Computer Vision, Steamboat Springs, CO, USA, March 24-26, 2014*, pages 461–468, 2014.

[54] Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in NLP. In *ACL*, 2007.

[55] Jing Jiang and ChengXiang Zhai. A two-stage approach to domain adaptation for statistical classifiers. In *CIKM*, pages 401–410, 2007.

[56] Yu-Gang Jiang, Jun Wang, Shih-Fu Chang, and Chong-Wah Ngo. Domain adaptive semantic diffusion for large scale context-based video annotation. In *ICCV*, pages 1420–1427, 2009.

[57] Ian T. Jolliffe. Principal component analysis. In *International Encyclopedia of Statistical Science*, pages 1094–1096. 2011.

[58] Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, pages 1785–1792, 2011.

[59] Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, 2011.

[60] Xuejun Liao, Ya Xue, and Lawrence Carin. Logistic regression with an auxiliary data source. In *ICML*, pages 505–512, 2005.

[61] Biao Liu, Minlie Huang, Jiashen Sun, and Xuan Zhu. Incorporating domain and sentiment supervision in representation learning for domain adaptation. In *IJCAI*, pages 1277–1283, 2015.

[62] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *NIPS*, pages 1041–1048, 2008.

[63] Geoffrey McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, 2004.

[64] John Neter, Michael H Kutner, Christopher J Nachtsheim, and William Wasserman. *Applied linear statistical models*, volume 4. Irwin Chicago, 1996.

[65] Sinno Jialin Pan, James T. Kwok, and Qiang Yang. Transfer learning via dimensionality reduction. In *AAAI*, pages 677–682, 2008.

[66] Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. In *IJCAI*, pages 1187–1192, 2009.

[67] Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

[68] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[69] Dohyung Park, Constantine Caramanis, and Sujay Sanghavi. Greedy subspace clustering. In *NIPS*, pages 2753–2761, 2014.

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

[71] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML*, 2007.

[72] Marcus Rohrbach, Sandra Ebert, and Bernt Schiele. Transfer learning in a transductive setting. In *NIPS*, pages 46–54, 2013.

[73] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

[74] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010.

[75] Avishek Saha, Piyush Rai, Hal Daumé III, Suresh Venkatasubramanian, and Scott L. DuVall. Active supervised domain adaptation. In *ECML PADD*, pages 97–112, 2011.

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

[77] Gabriele Schweikert, Christian Widmer, Bernhard Schölkopf, and Gunnar Rätsch. An empirical analysis of domain adaptation algorithms for genomic sequence analysis. In *NIPS*, pages 1433–1440, 2008.

[78] Sumit Shekhar, Vishal M. Patel, Hien Van Nguyen, and Rama Chellappa. Generalized domain-adaptive dictionaries. In *CVPR*, pages 361–368, 2013.

[79] Si Si, Dacheng Tao, and Bo Geng. Bregman divergence-based regularization for transfer subspace learning. *IEEE Trans. Knowl. Data Eng.*, 22(7):929–942, 2010.

[80] Alexander Statnikov, Constantin F. Aliferis, Ioannis Tsamardinos, Douglas Hardin, and Shawn Levy. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, 21(5):631–643, 2005.

[81] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul von Bünau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *NIPS*, pages 1433–1440, 2007.

[82] Qian Sun, Rita Chattopadhyay, Sethuraman Panchanathan, and Jieping Ye. A two-stage weighting framework for multi-source domain adaptation. In *NIPS*, pages 505–513, 2011.

[83] Shi-Liang Sun and Hong-Lei Shi. Bayesian multi-source domain adaptation. In *ICMLC*, pages 24–28, 2013.

[84] Shiliang Sun, Zhijie Xu, and Mo Yang. Transfer learning with part-based ensembles. In *MCS*, pages 271–282, 2013.

[85] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[86] Xuezhi Wang, Tzu-Kuo Huang, and Jeff Schneider. Active transfer learning under model shift. In *ICML*, pages 1305–1313, 2014.

[87] Xuezhi Wang and Jeff Schneider. Flexible transfer learning under support and model shift. In *NIPS*, pages 1898–1906, 2014.

[88] Min Xiao and Yuhong Guo. Semi-supervised kernel matching for domain adaptation. In *AAAI*, 2012.

[89] Min Xiao and Yuhong Guo. Feature space independent semi-supervised domain adaptation via kernel matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(1):54–66, 2015.

[90] Dikan Xing, Wenyuan Dai, Gui-Rong Xue, and Yong Yu. Bridged refinement for transfer learning. In *PKDD*, pages 324–335, 2007.

[91] Caiming Xiong, Scott McCloskey, Shao-Hang Hsieh, and Jason J. Corso. Latent domains modeling for visual domain adaptation. In *AAAI*, pages 2860–2866, 2014.

[92] Zhijie Xu and Shiliang Sun. Multi-source transfer learning with multi-view adaboost. In *Proceedings of the 19th International Conference on Neural Information Processing - Volume Part III*, ICONIP'12, pages 332–339, Berlin, Heidelberg, 2012. Springer-Verlag.

[93] Xingwei Yang, Lakshman Prasad, and Longin Jan Latecki. Affinity learning with diffusion on tensor product graph. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):28–38, 2013.

[94] Quanzeng You, Jiebo Luo, Hailin Jin, and Jianchao Yang. Robust image sentiment analysis using progressively trained and domain transferred deep networks. In *AAAI*, pages 381–388, 2015.

[95] Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *ICML*, 2004.

[96] Chao Zhang, Lei Zhang, Wei Fan, and Jieping Ye. Generalization bounds for representative domain adaptation. *CoRR*, abs/1401.0376, 2014.

[97] Kun Zhang, Mingming Gong, and Bernhard Schölkopf. Multi-source domain adaptation: A causal view. In *AAAI*, pages 3150–3157, 2015.

[98] Xiaojin Zhu. Semi-supervised learning literature survey. 2005.

[99] Xiaojin Zhu, Zoubin Ghahramani, and John D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.