Adapting Component Analysis

by

Fatemeh Dorri

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2012

© Fatemeh Dorri 2012

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.
Fatemeh Dorri

I understand that my thesis may be made electronically available to the public.
Fatemeh Dorri

Abstract

A main problem in machine learning is to predict the response variables of a test set given the training data and its corresponding response variables. A predictive model can perform satisfactorily only if the training data is an appropriate representative of the test data. This intuition is reflected in the assumption that the training data and the test data are drawn from the same underlying distribution. However, the assumption may not be correct in many applications for various reasons. For example, gathering training data from the test population might not be easily possible, due to its expense or rareness. Or, factors like time, place, weather, etc can cause the difference in the distributions.

I propose a method based on kernel distribution embedding and Hilbert Schmidt Independence Criteria (HSIC) to address this problem. The proposed method explores a new representation of the data in a new feature space with two properties: (i) the distributions of the training and the test data sets are as close as possible in the new feature space, (ii) the important structural information of the data is preserved. The algorithm can reduce the dimensionality of the data while it preserves the aforementioned properties and therefore it can be seen as a dimensionality reduction method as well. Our method has a closed-form solution and the experimental results on various data sets show that it works well in practice.

Acknowledgements

I would like to express my appreciation to those who made this thesis possible:

I would like to express my gratitude to my supervisor, Professor Ali Ghodsi, for providing me with the opportunity to work in his group, for his support, guidance and the lessons he taught me.

I would like to thank Professor Pascal Poupart and Professor Daniel Lizotte for reviewing my thesis.

I would like to thank all my friends in Waterloo, specially my reliable and true friend Razieh for her encouragements, help and good time we had together.

Last but not least, I would like to thank all my family member with love: my father, Behrouz and my mother, Zahra, for their end-less love and support; my dearest twin sister, Faezeh, for being with me every minutes of my life; my brother-in-law, Hamid, for his kindness and guidance; and my beloved husband, Mohsen, for his love, patience, help and support.

To my beloved husband,

Mohsen

Table of Contents

List of Tables **xiii**

List of Figures **xv**

1 Introduction **1**

1.1 Notation 4

1.2 Overview 5

1.2.1 Re-weighting source instances 7

1.2.2 Changing the representation space 9

1.2.3 Others 10

1.3 Contribution 11

1.4 Outline 12

2 Adapting Component Analysis **13**

2.1 Introduction 13

2.2 Preliminaries 15

2.2.1 Hilbert-Schmidt Independence Criterion 15
2.2.2 Maximum Mean Discrepancy 17
2.3 Adapting Component Analysis 18
2.3.1 Minimizing The Distance Between $P(\Phi_{tr})$ and $P(\Phi_{ts})$ 20
2.3.2 Preserving the Properties of X 20
2.3.3 Unsupervised-ACA 22
2.3.4 Semi-Supervised ACA 24

3 Experimental Result **29**
3.1 The kernel on the response variables 30
3.2 Toy Classification Example 31
3.3 Real world data sets 34

4 Conclusion **45**

References **47**

List of Tables

2.1 Kernel Functions $\mathbf{K}(\mathbf{x}, \mathbf{y})$ 23

3.1 The error rate of the SVM on the original data and on the modified data of US-ACA and SS-ACA 33

3.2 Different data set generated from MNIST database 35

3.3 Test result on various data sets by different methods. 1-nearest neighbour has been used for classification. 41

List of Figures

1.1 Probability distributions of training and test data sets in a binary classification problem. 2

2.1 The training and test data set samples in the original space. 21

2.2 The representation of the training and test data set samples in the new feature space. 22

3.1 (a) 2-dimensional data in the original space. Circles, $\circ$, and crosses, $\times$ are two classes of the training data set and diamonds,$\diamond$, and stars, $*$, are two classes of the test data set. (b) The new representation of the data in new feature space based on US-ACA and (c) The new representation of the data in new feature space based on SS-ACA 32

3.2 The error rate comparison for different algorithms and the baseline on a toy example 34

3.3 The error rate comparison for different algorithms and baseline on the data sets generated from MNIST data base. (1) to (7) on the X-axis stands for Digits(1) to Digits(7) respectively 36

3.4 (a) The 2-dimensional representation of Digits(1) data set based on US-ACA. (b) The 2-dimensional representation of the same data based on SS-ACA. Circles, $\circ$, and crosses, $\times$ are two classes of the training data set. Diamonds,$\diamond$, and stars, $*$, are two classes of the test data set after applying 1-NN. 37

3.5 The error rate comparison for different algorithms in three data sets. 1, 2 and 3 on the X-axis stands for Newsgroup1, Newsgroup2 and Newsgroup3 data sets respectively. 38

3.6 (a), (c) and (e) are the error rate comparison in different algorithms on Breast Cancer data set with *Biasing Ratio* of 70%, 80% and 90% respectively. The X-axis is showing the features which the biasing process is based on. (b), (d) and (e) are the Normalized Improvement of SS-ACA with respect to the baselines of (a), (c) and (e) respectively. The bars from left to right correspond to Basline, US-ACA, SS-ACA, MMDE and CODA. 42

3.7 (a) The error rate changes of SS-ACA vs. different dimensions on Digits(1), Digits(2) and Digits(3) data sets. (b) The error rate changes of SS-ACA vs. different dimensions on Wine data set. (c) The error rate changes of SS-ACA vs. different dimensions on Newsgroup2 data set. 43

Chapter 1

Introduction

In the realm of machine learning, a model is trained to predict the response variables of a test data set. The training procedure is usually based on minimizing a loss function over all samples of a training data set and their corresponding response variables. However, learning achieves its purpose only when the training data set is a suitable representative of the test data set; otherwise the method learns unrelated information, and therefore the efficiency of the prediction is not satisfactory.

In conventional predictive models, the intuition that the training data is a suitable representative of the test data is reflected in the assumption that the underlying distributions of the training and test data sets are identical. But this assumption is not always valid. As an example, Figure 1.1 shows artificial samples that belong to a binary classification problem with non-similar probability distributions for the training and test data sets. It shows the training samples are not good representatives of the test data set.

Different reasons may cause the underlying probability distributions of the training and test data sets to be different. The reason might be in the difficulty or uncontrollability in gathering data. For example, having samples in the training data set from the test data

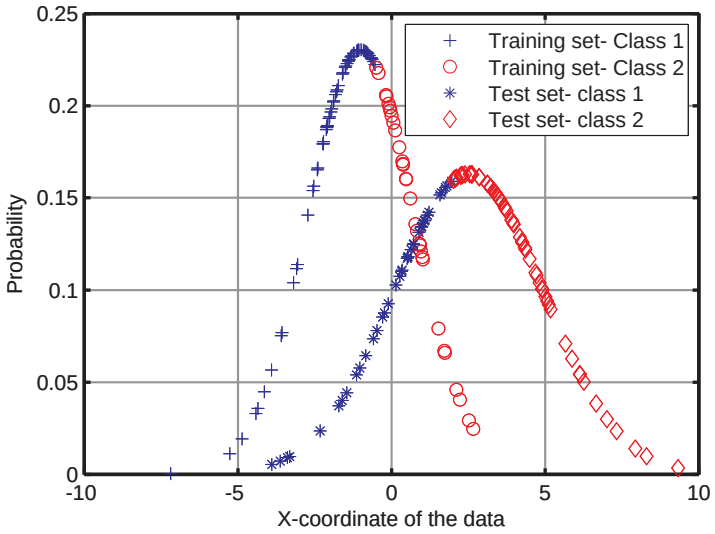

Figure 1.1: Probability distributions of training and test data sets in a binary classification problem.

set distribution might not be possible as it costs a lot or because of its unavailability at the moment. In other cases, the process of gathering training data might involve a selection process, i.e., the training data samples are selected based on a parameter which might cause the difference in the probability distributions of the training and test data sets[1]. Therefore, for various reasons, the training and test data set domains need to be adapted before any further predictive analysis.

In last decades, attention has been focused on domain adaptation problem in machine

[1]A typical example is that of credit scoring: the training data set consists of only customers who have requested for a loan and their request has been accepted (the customers whose requests have been rejected are not included in the training data set).

learning [1, 2]. It is now widely used in diverse fields [2, 3]. Some examples in which a domain adaptation algorithm helps improving the efficiency of a predictive model are:

- *The problem of generating a predictive model for a certain cancer diagnosis.* Available data sets are usually from older populations who are more likely to have the cancer and are willing to be monitored; however the test data set population is not necessarily from the same age group.

- *Analysis of web data volumes that are increasing daily.* Obviously, the distribution of customers' feedback for a new shoe in Amazon is different from reviewers' feedback for a book (because people use different terminology to express similar concepts for different objects.). However, it might be desirable to classify data into the positive and the negative feedbacks for the new shoe product with a model trained based on the feedbacks for books. Achieving this goal is accomplished by transferring the learned information from book reviews in an intelligent way.

- *Studying the effect of new developed drugs on human body.* The test data set response variables might not be accessible due to ethical concerns. Or, although the response variables to the previously tested drugs or similar drugs on other species can be available, the training data set might not include the effect of a new or a risky drug on humans. So in spite of having numerous samples in the training data set, its distribution might differ from that of a new untested drug data for various reasons like difference in the probability distributions of the people who are prescribed for different drugs.

- *The biased procedure of collecting data in a web survey.* The training data set includes people who have access to the internet while the test data set is for example, all citizens of a country. So gathering the data from on-line questionnaires makes the training distribution inclined to a specific group of people.

- *Data might only be valid for a period of time.* For example, the attitude about a book, author, music, etc. can change over time. So, if the test data set is gathered after the time when the training data is valid, the training data set will not be a good representative of the test data set.

In this thesis, I propose a new domain adaptation method and test its performance on toy and real data sets. The remainder of this chapter describes the notation and the overview of the existing methods.

1.1 Notation

Let $\mathcal{X}$ denote a random variable (i.e. input variables in the original feature space) and $\mathcal{Y}$ denote its corresponding response variables (i.e. output variables like class labels). $P(\mathcal{X}, \mathcal{Y})$ is the underlying joint probability distributions of $\mathcal{X}$ and $\mathcal{Y}$. In domain adaptation problems, the probability distributions of the training and test data sets (source and target domains respectively) are different. $P_{tr}(\mathcal{X}, \mathcal{Y})$ and $P_{ts}(\mathcal{X}, \mathcal{Y})$ denote the underlying true joint probability distributions of the training and test data sets respectively. $P_{tr}(\mathcal{X})$, $P_{tr}(\mathcal{Y})$, $P_{ts}(\mathcal{X})$ and $P_{ts}(\mathcal{Y})$ denote the true marginal probability distributions of $\mathcal{X}$ and $\mathcal{Y}$ in the training and test data sets. Similarly, $P_{tr}(\mathcal{X}|\mathcal{Y})$ and $P_{ts}(\mathcal{X}|\mathcal{Y})$ are used to show the true conditional probability distributions in the two domains.

The bold lower case alphabet is used to refer to a vector like a specific observation of a variable. $P(\mathbf{x}, \mathbf{y})$ where $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$, shows the joint probability of a sample $\mathbf{x}$ and its corresponding label (response variable) $\mathbf{y}$. It is equal to $P(\mathcal{X} = \mathbf{x}, \mathcal{Y} = \mathbf{y})$.

$\mathbf{x}$ is a d-dimensional sample. $\mathbf{X}_{tr}$ and $\mathbf{X}_{ts}$ denote the matrices of the training and test data set samples of size $d \times n_{tr}$ and $d \times n_{ts}$ respectively. n_{tr} and n_{ts} are the numbers of the samples in the training and test data sets respectively. $\mathbf{X}_{d \times n} = [\mathbf{X}_{tr} \mathbf{X}_{ts}]$ is a matrix of n, $d-$dimensional samples where $n = n_{tr} + n_{ts}$.

1.2 Overview

Predictive models learn a function, $f(x)$, which will be used for predicting the response variables. The function is estimated to have a predefined loss function minimized. The most frequently used loss functions include:

1. *Logistic loss function*

$$l_{log}(f(\mathbf{x}), \mathbf{y}) = log(1 + exp(-\mathbf{y}f(\mathbf{x})) \tag{1.2.1}$$

2. *Hinge loss function*

$$l_{hinge}(f(\mathbf{x}), \mathbf{y}) = max(0, 1 - \mathbf{y}f(\mathbf{x})) \tag{1.2.2}$$

3. *Exponential loss function*

$$l_{exp}(f(\mathbf{x}), \mathbf{y}) = exp(-\mathbf{y}f(\mathbf{x})) \tag{1.2.3}$$

4. *Squared loss function*, it is one of the favourites that is frequently used:

$$l_{squared}(f(\mathbf{x}), \mathbf{y}) = (\mathbf{y} - f(\mathbf{x}))^2 \tag{1.2.4}$$

The training classification error of a predictive model is the minimized loss function over all the training samples in the training step:

$$\varepsilon_{tr} = \int\int l(f(\mathbf{x}), \mathbf{y}) p_{tr}(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y}, \tag{1.2.5}$$

It is desired to have the minimum classification error, ε, which is the loss over the test data set. It is defined as:

$$\varepsilon = \int\int l(f(\mathbf{x}), \mathbf{y}) p_{ts}(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y}. \tag{1.2.6}$$

If the training and the test data sets do not share the same probability distribution, minimizing the objective function in Equation 1.2.5 will not, in general, yield the minimum loss over the test data set in Equation 1.2.6.

Depending on the initial assumptions, a domain adaptation problem gets different names [4]. "Covariate shift" [5, 6] and "class imbalance" [7] are two examples with different initial assumptions:

- *Covariate shift*- Decomposing the joint probability distributions of the training and test data sets as

$$p_{tr}(\mathcal{X}, \mathcal{Y}) = p_{tr}(\mathcal{X})p_{tr}(\mathcal{Y}|\mathcal{X})$$
$$p_{ts}(\mathcal{X}, \mathcal{Y}) = p_{ts}(\mathcal{X})p_{ts}(\mathcal{Y}|\mathcal{X}),$$

  the problem fits into the so called "covariate shift" [5], if $p_{tr}(\mathcal{Y}|\mathcal{X}) = p_{ts}(\mathcal{Y}|\mathcal{X})$ and $p_{tr}(\mathcal{X}) \neq p_{ts}(\mathcal{X})$. It assumes all the difference between the joint probability distributions of the training and test data sets is due to the difference between their marginal probability distributions.

- *Class imbalance*- Decomposing the joint probability distributions of the training and test data sets as

$$p_{tr}(\mathcal{X}, \mathcal{Y}) = p_{tr}(\mathcal{Y})p_{tr}(\mathcal{X}|\mathcal{Y})$$
$$p_{ts}(\mathcal{X}, \mathcal{Y}) = p_{ts}(\mathcal{Y})p_{ts}(\mathcal{X}|\mathcal{Y}),$$

  the problem fits into the so called "class imbalance" [8, 9], if $p_{tr}(\mathcal{X}|\mathcal{Y}) = p_{ts}(\mathcal{X}|\mathcal{Y})$ and $p_{tr}(\mathcal{Y}) \neq p_{ts}(\mathcal{Y})$. It assumes all the difference between the joint probability distributions of the training and test data sets is due to the difference between the marginal probability distributions:

$$p_{tr}(\mathcal{Y}) \neq p_{ts}(\mathcal{Y}). \tag{1.2.7}$$

Comparing different problems with different settings, there are also closely relevant but not equivalent problems that have been studied extensively including semi-supervised learning [10, 11] and multi-task learning [12, 13].

In semi-supervised learning, regardless of traditional learning methods, both the unlabeled data and the labeled data are utilized to improve the performance of the classifier. Domain adaptation is similar to a semi-supervised problem in the sense that ignoring the difference between the training and the test data set domains, a semi-supervised problem remains if the labeled samples of the training data set are treated as the labeled data and the unlabeled samples of the test data set are treated as the unlabeled data of the semi-supervised learning [14].

In multi-task learning, instead of having different joint probability distributions for the training and test data sets, there are M various response variables. So, there are M different joint probability distributions $\{P(\mathcal{X}, \mathcal{Y}_k)\}_{k=1}^{M}$ for learning M tasks [12, 13, 15, 16]. Multi-task learning and domain adaptation problems are very similar to each other in the sense that both deal with different probability distributions. Domain adaptation can be transformed to a special case of multi-task learning where the training domain and the test domain are treated as two tasks and they do have the same class labels. Indeed, some of the proposed algorithms for domain adaptation are basically multi-task learning algorithms [17].

Although domain adaptation has been studied under different names [4] e.g. covariate shift [5], class imbalance [7], semi-supervised learning [11, 18, 19], multi-task learning [13, 15, 16] and sample selection bias [20, 21], all these methods usually tackle the problem by two approaches: re-weighting source instances or changing the representation space [4] which are explained in Sections 1.2.1 and 1.2.2. The other approaches are mentioned briefly in Section 1.2.3.

1.2.1 Re-weighting source instances

Considering a domain adaptation problem, the goal is to have an optimal model, $f^*(\mathbf{x})$ which minimizes the expected loss function over the samples of the test data set defined in

Equation 1.2.6. However, the optimal model is estimated based on Equation 1.2.5 over the samples of the training data set. In order to solve the issue of the difference between the probability distributions of the training and test data sets; weights, $w_{tr}(\mathbf{x}, \mathbf{y})$, are assigned to the pre-defined loss function in Equation 1.2.5. A modified loss function over the training data set is then minimized [2, 22]. The objective function is rewritten as

$$f^*(\mathbf{x}) = \arg\min_{f \in \mathcal{F}} \int \int w_{tr}(\mathbf{x}, \mathbf{y}) P_{tr}(\mathbf{x}, \mathbf{y}) l(f(\mathbf{x}), \mathbf{y}) d\mathbf{x} d\mathbf{y}, \tag{1.2.8}$$

where each instance weight, $w_{tr}(\mathbf{x}, \mathbf{y})$, is defined to be

$$w_{tr}(\mathbf{x}, \mathbf{y}) = \frac{P_{ts}(\mathbf{x}, \mathbf{y})}{P_{tr}(\mathbf{x}, \mathbf{y})}, \tag{1.2.9}$$

for each sample of $(\mathbf{x}, \mathbf{y}) \in (\mathcal{X}_{tr}, \mathcal{Y}_{tr})$. Therefore, the objective function is equal to

$$\begin{aligned} f^*(\mathbf{x}) &= \arg\min_{f \in \mathcal{F}} \int \int \frac{P_{ts}(\mathbf{x}, \mathbf{y})}{P_{tr}(\mathbf{x}, \mathbf{y})} P_{tr}(\mathbf{x}, \mathbf{y}) l(f(\mathbf{x}), \mathbf{y}) d\mathbf{x} d\mathbf{y} \\ &= \arg\min_{f \in \mathcal{F}} \int \int P_{ts}(\mathbf{x}, \mathbf{y}) l(f(\mathbf{x}), \mathbf{y}) d\mathbf{x} d\mathbf{y} \end{aligned} \tag{1.2.10}$$

It shows minimizing the weighted loss function over the training data samples is the same as minimizing the loss function over the test data samples as Equation 1.2.6. To estimate the weights, $w_{tr}(\mathbf{x}, \mathbf{y})$, it is not required to calculate the joint probability distributions of the training and test data sets separately. It only needs to have an approximation of $\frac{P_{ts}(\mathbf{x},\mathbf{y})}{P_{tr}(\mathbf{x},\mathbf{y})}$.

For example, in class imbalance setting, this ratio is simplified to

$$\frac{P_{ts}(\mathbf{x}, \mathbf{y})}{P_{tr}(\mathbf{x}, \mathbf{y})} = \frac{P_{ts}(\mathbf{y})}{P_{tr}(\mathbf{y})} \frac{P_{ts}(\mathbf{x}|\mathbf{y})}{P_{tr}(\mathbf{x}|\mathbf{y})} \tag{1.2.11}$$

$$= \frac{P_{ts}(\mathbf{y})}{P_{tr}(\mathbf{y})}, \tag{1.2.12}$$

since the conditional probability distributions of the training and test data sets are assumed to be identical [8, 9]. Therefore, the problem is broken down to estimating $\frac{P_{ts}(\mathbf{y})}{P_{tr}(\mathbf{y})}$ instead

of $\frac{P_{ts}(\mathbf{x},\mathbf{y})}{P_{tr}(\mathbf{x},\mathbf{y})}$. The solution of this problem depends on whether the marginal test probability distributions are known a priori [8] or there is no prior knowledge about that [9, 23, 24].

Covariate shift setting is another example, this ratio is simplified differently based on the assumption that $P_{ts}(\mathbf{y}|\mathbf{x}) = P_{tr}(\mathbf{y}|\mathbf{x})$ and $P_{tr}(\mathbf{x}) \neq P_{ts}(\mathbf{x})$ [5, 23, 24]:

$$\frac{P_{ts}(\mathbf{x},\mathbf{y})}{P_{tr}(\mathbf{x},\mathbf{y})} = \frac{P_{ts}(\mathbf{x})}{P_{tr}(\mathbf{x})}\frac{P_{ts}(\mathbf{y}|\mathbf{x})}{P_{tr}(\mathbf{y}|\mathbf{x})} \tag{1.2.13}$$

$$= \frac{P_{ts}(\mathbf{x})}{P_{tr}(\mathbf{x})} \tag{1.2.14}$$

To estimate $\frac{P_{ts}(\mathbf{x})}{P_{tr}(\mathbf{x})}$, different methods have been suggested: Shimodaira proposed a method based on a non-parametric kernel density estimation [5], Zadrozny transformed the problem into a new one in which for each instance it is needed to specify if it is from the training data set or the test data set[21], and Huang et. al transformed the problem into a kernel mean matching problem in a reproducing Hilbert space [2].

In the cases where there are no assumption on the similarity of $P_{tr}(\mathbf{x}|\mathbf{y})$ and $P_{ts}(\mathbf{x}|\mathbf{y})$ or $P_{tr}(\mathbf{y}|\mathbf{x})$ and $P_{tr}(\mathbf{y}|\mathbf{x})$, the problem is handled differently. For example, Jiang et. al have suggested a supervised method in which the "misleading" samples are removed to make the joint distribution of the training and test data sets similar to each other [25].

1.2.2 Changing the representation space

In this approach, the data is mapped into a new feature space. Regardless of the dimensionality of the new representation of data which might be in a higher [26] or lower dimensional space [27, 28], the probability distributions of the training data and that of the test data in the new feature space are more similar.

Putting it into mathematics, let $\Psi : \mathcal{X} \to \mathcal{Z}$, map the data into a new space called $\mathcal{Z}$.

The probability distribution of $\mathbf{z} \in \mathcal{Z}$ is

$$P(\mathbf{z}) = \sum_{\mathbf{x} \in \mathcal{X}, g(\mathbf{x}) = \mathbf{z}} P(\mathbf{x}). \tag{1.2.15}$$

therefore the joint probability distribution of $\mathbf{z}$ and $\mathbf{y}$ is:

$$P(\mathbf{z}, \mathbf{y}) = \sum_{\mathbf{x} \in \mathcal{X}, g(\mathbf{x}) = \mathbf{z}} P(\mathbf{x}, \mathbf{y}). \tag{1.2.16}$$

It is desired to find a new representation of data in which the probability distributions of the training and test data sets are more similar. The closeness of two probability distributions is estimated based on the distance between corresponding distributions. However the new representation of data must be capable of keeping the valuable information hidden in the data (needed for any further analysis).

This approach has been studied extensively. For example, Satpal et. al proposed a feature selection algorithm where the features are chosen based on minimizing the distance function between $P_{tr}(\mathcal{Z}, \mathcal{Y})$ and $P_{ts}(\mathcal{Z}, \mathcal{Y})$ [29]. Blitzer et. al proposed a structural correspondence learning (SCL) algorithm that takes advantage of the unlabeled test data for finding a low rank representation of the data with $P_{tr}(\mathcal{Z}, \mathcal{Y}) = P_{ts}(\mathcal{Z}, \mathcal{Y})$ [28].

1.2.3 Others

The other approaches for tackling domain adaptation problems are combined models for which a mixture of models is learned to improve the performance and efficiency of the algorithm. There exist numerous algorithms that work based on this approach. Daume III et al. proposed a method which basically learns three components utilizing the response variables of both the training and test data sets. One of the components is shared between the training and test data sets and the other two are specified to the training and test data sets separately [30]. Storkey et al. also suggested a method which assumes only the

response variables of the training data set is accessible. It considers a more general mixture of models based on expectation maximization (EM) [31]. Dai et. al proposed a method that uses an algorithm called AdaBoost to address the domain adaptation problem [32]. There are also some work based on Bayesian Priors. For example, Li and Bilmes [33] suggested a general Bayesian divergence prior framework that can be instantiate for a variety of classifiers.

1.3 Contribution

There are common drawbacks among all the mentioned methods: (i) approximation of the underlying distributions makes solving the problem hard in high dimensional data sets, and (ii) some domain adaptation techniques are applicable only to restricted predictive models.

In this thesis, I propose a method that overcomes the above drawbacks based on kernel distribution embedding and Hilbert Schmidt Independence Criterion (HSIC). The proposed algorithm finds a new representation of the data in a new feature space such that the underlying probability distributions of the embedded training and test data sets are as close as possible and the important structural information of the data is also preserved for any further predictive analysis. These two constraints make a single optimization problem which has a closed-form solution. The algorithm has a good performance when the data is mapped to a lower dimensional space. So it can be used a dimensionality reduction technique as well.

1.4 Outline

This thesis is organized as follows: In Chapter 2, I briefly explain Hilbert-Schmidt Independence Criterion (HSIC), Maximum Mean Discrepancy(MMD) and then I will continue to explain my algorithm. The experimental results and conclusions are presented in Chapters 3 and 4 respectively.

Chapter 2

Adapting Component Analysis

2.1 Introduction

A predictive model problem includes the training samples, their corresponding response variables, if accessible, and the test samples. The goal is to find the response variables of the test samples. The probability distributions of the training and test data sets are generally assumed to be the same in most of the predictive models. However, the training and the test data sets may not be governed by the same probability distribution and the predictor models assumption may be violated. This problem has been called domain adaptation and it should be solved before any further predictive analysis.

We reviewed two main approaches in literature for solving a domain adaptation problem: (i) re-weighting the source instances and (ii) changing the representation space. However, these approaches have some drawbacks. In the first approach, re-weighting the source instances, it is typically required to approximate at least one probability distribution which itself can be problematic in high-dimensional data. In the second approach, changing the representation space, the algorithm is either restricted to linear transformations, or even

if it covers the non-linear cases, it is not practically efficient in a high dimensional data. That is because of their iterative nature which makes solving corresponding optimization problems computationally intensive and time consuming.

In this chapter, a domain adaptation algorithm is proposed, based on the second approach, that is changing the representation space. This algorithm overcomes the aforementioned drawbacks which exist in the mentioned algorithms: First, in the proposed method, there is no need for approximating any probability distribution which has been a problem in re-weighting the source instances approach. Second, there is no linearity constraint or assumption for solving the problem and the algorithm finds a new representation of the data without solving any iterative optimization problem. Indeed, our proposed algorithm has a closed-form solution. Moreover, the algorithm finds an appropriate representation of the data in a new feature space of a lower dimension. Therefore, having the dimensionality of the new representation smaller than its original space, the algorithm can be considered as a dimensionality reduction method.

Our proposed domain adaptation algorithm explores a new representation of the data in a new feature space satisfying the following properties: (i) it makes the probability distributions of the training and test data sets in the new feature space as close as possible. (ii) it does not disturb the meaningful and important features of the original data set.

In order to achieve these two properties, we first need to have a definition and an estimate of the distance between two probability distributions. The distance is formulated based on Hilbert-Schmidt independence criteria and Maximum Mean Discrepancy (MMD) which are explained in more details in Section 2.2. Our algorithm called, UnSupervised Adopting Component Analysis (US-ACA), is introduced in Section 2.3. It forms the foundation of its semi-supervised version introduced in the same section. We will show while US-ACA does not use any information about the labels or response variables of the data, Semi-Supervised ACA (SS-ACA) utilizes the available response variables data and

therefore its performance is improved significantly.

2.2 Preliminaries

I first review Hilbert-Schmidt Independence Criteria (HSIC) and the Maximum Mean Discrepancy (MMD). These are needed for establishing and justifying our algorithm.

2.2.1 Hilbert-Schmidt Independence Criterion

Hilbert-Schmidt Independence Criterion is a measure of dependency between two random variables. HSIC is based on the fact that if $\mathcal{X}$ and $\mathcal{Y}$ are two random variables, then these two random variables are independent if and only if any continuous function of them are uncorrelated. The correlation of the two random variables is formulated as their covariance. Assume $\mathcal{F}$ and $\mathcal{G}$ be the separable Reproducing Kernel Hilbert Space (RKHS) of real-valued functions from $\mathcal{X}$ and $\mathcal{Y}$ to $\mathbb{R}$ with universal kernels $\mathbf{K}(.,.)$ and $\mathbf{L}(.,.)$ respectively [34]. The covariance between $f(\mathbf{x})$ and $g(\mathbf{x})$ is

$$cov(f(\mathbf{x}), g(\mathbf{x})) = \mathbf{E}_{\mathbf{x},\mathbf{y}}[f(\mathbf{x})g(\mathbf{y})] - \mathbf{E}_{\mathbf{x}}[f(\mathbf{x})]\mathbf{E}_{\mathbf{y}}[g(\mathbf{y})] \tag{2.2.1}$$

where $\mathbf{E}$ denotes the expected value and f and g, are two functions in $\mathcal{F}$ and $\mathcal{G}$ respectively. It is shown that there exist a unique cross-covariance operator $C_{\mathcal{X}\mathcal{Y}} : \mathcal{G} \to \mathcal{F}$ that maps the elements of $\mathcal{G}$ to the elements of $\mathcal{F}$ such that

$$\langle f, C_{\mathcal{X}\mathcal{Y}} g\rangle = cov(f(\mathbf{x}), g(\mathbf{y})), \forall f \in \mathcal{F}, \forall g \in \mathcal{G}. \tag{2.2.2}$$

HSIC is defined to be the squared of the Hilbert-Schmidt norm of the cross-covariance operator

$$HSIC(P_{\mathcal{X}\mathcal{Y}}, \mathcal{F}, \mathcal{G}) := \|C_{\mathcal{X}\mathcal{Y}}\|^2_{HS} \tag{2.2.3}$$

where $\|.\|^2_{HS}$ denotes the squared of Hilbert-Schmidt norm and $P_{\mathcal{XY}}$ is the joint probability distribution of random variables $\mathcal{X}$ and $\mathcal{Y}$ [34]. It can be shown that the iff $\|C_{\mathcal{XY}}\|^2_{HS} = 0$, the two random variables $\mathcal{X}$ and $\mathcal{Y}$ are independent as long as their RKHSs are universal. The cross-covariance operator is

$$C_{\mathcal{XY}} := \mathbf{E}_{\mathbf{x},\mathbf{y}}[(\Phi(\mathbf{x}) - \mu_{\mathbf{x}}) \otimes (\Psi(\mathbf{y}) - \mu_{\mathbf{y}})] \tag{2.2.4}$$

where μ is the mean, Φ and Ψ, are the feature maps of $\mathcal{F}$ and $\mathcal{G}$ respectively and $\otimes$ is the tensor product. HSIC is expressed in terms of kernel functions:

$$\begin{aligned} \mathit{HSIC}(P_{\mathcal{X},\mathcal{Y}}, \mathcal{F}, \mathcal{G}) &= \mathbf{E}_{\mathbf{x},\mathbf{x}',\mathbf{y},\mathbf{y}'}[K(\mathbf{x},\mathbf{x}')L(\mathbf{y},\mathbf{y}')] \\ &+ \mathbf{E}_{\mathbf{x},\mathbf{x}'}[K(\mathbf{x},\mathbf{x}')]\mathbf{E}_{\mathbf{y},\mathbf{y}'}[L(\mathbf{y},\mathbf{y}')] \\ &- 2\mathbf{E}_{\mathbf{x},\mathbf{y}}[\mathbf{E}_{\mathbf{x}'}[(\mathbf{x},\mathbf{x}')]\mathbf{E}_{\mathbf{y}'}[L(\mathbf{y},\mathbf{y}')]] \end{aligned} \tag{2.2.5}$$

where $\mathbf{E}_{\mathbf{x},\mathbf{x}',\mathbf{y},\mathbf{y}'}$ is expectation over $(\mathbf{x},\mathbf{y})$ and $(\mathbf{x}',\mathbf{y}')$ such that $(\mathbf{x},\mathbf{y})$ and $(\mathbf{x}',\mathbf{y}')$ are random variables taken independently from $P_{\mathcal{XY}}$ from $\mathcal{Z} := \{(\mathbf{x}_1,\mathbf{y}_1),\ldots,(\mathbf{x}_n,\mathbf{y}_n)\} \subseteq \mathcal{XY}$ (i.e. the set of n observations). Estimating HSIC should be practical with a finite number of samples to make it possible to measure the dependency between a finite number of observations. An empirical estimation of HSIC is:

$$\mathit{HSIC}(\mathcal{Z},\mathcal{F},\mathcal{G}) := (n-1)^{-2}\mathbf{tr}(\mathbf{KHLH}) \tag{2.2.6}$$

where $\mathbf{H}$, $\mathbf{K}$ and $\mathbf{L} \in \mathbb{R}^{n\times n}$, $\mathbf{K}_{ij} := \mathbf{K}(\mathbf{x}_i,\mathbf{x}_j)$, $\mathbf{L}_{ij} := \mathbf{L}(\mathbf{y}_i,\mathbf{y}_j)$, $\mathbf{H}$ is the centering matrix that is defined to be $\mathbf{H} := \mathbf{I} - n^{-1}\mathbf{e}\mathbf{e}^T$, and $\mathbf{e}$ is a vector of ones. If one of the kernels, let's say $\mathbf{L}$ is already centralized, then we have $\mathbf{HLH} = \mathbf{L}$ and the formula will be simplified to $\mathbf{tr}(\mathbf{KL})$ [34]. *HSIC* is a measure of dependency so, the larger the trace of the multiplication of their corresponding kernels, $\mathbf{tr}(\mathbf{KL})$, the more the dependency between two random variables, $\mathcal{X}$ and $\mathcal{Y}$.

2.2.2 Maximum Mean Discrepancy

Let's assume $\mathbf{X}$ is the collection of two clusters $\{\mathbf{X}_i^{(1)}\}_{i=1}^{n_1}$ and $\{\mathbf{X}_i^{(2)}\}_{i=1}^{n_2}$ such that their empirical distributions are defined as

$$\hat{\pi}_k \hat{P}_k = \frac{1}{n} \sum_{i=1}^{n} \alpha_i^{(k)} \delta_{x_i} \tag{2.2.7}$$

where δ is the Dirac function, $k \in \{1, 2\}$ (on the case of two clusters) denotes the cluster label, $\alpha_i^{(k)}$ is a binary indicator variable taking 1 for $\mathbf{x}_i \in k$, and

$$\hat{\pi}_k = \frac{1}{n} \sum_{i=1}^{n} \alpha_i^{(k)}. \tag{2.2.8}$$

Therefore

$$\hat{P} = \hat{\pi}_1 \hat{P}_1 + \hat{\pi}_2 \hat{P}_2,$$

where $\forall i \in \{1, 2\}$, $\hat{P}_i$ is the probability distribution of cluster i and $\hat{\pi}_i$ is the probability of having a sample from ith cluster.

The Maximum Mean Discrepancy (MMD) [35] between the two sets of data with probability distributions $\hat{P}_1$ and $\hat{P}_2$ is a metric representative of the distance between the means of those distributions and it has been defined as

$$\begin{aligned} MMD(\hat{P}_1, \hat{P}_2) &= \|\mu_{\mathbf{x}}[\hat{P}_1] - \mu_{\mathbf{y}}[\hat{P}_2]\|_{\mathcal{H}} \\ &= \sup_{g \in \mathcal{F}, \|g\|_{\mathcal{H}} \leq 1} (\mathbf{E}_{\mathbf{x} \sim \hat{P}_1} g(\mathbf{x}) - \mathbf{E}_{\mathbf{y} \sim \hat{P}_2} g(\mathbf{y})), \end{aligned} \tag{2.2.9}$$

where $\mathbf{E}_{\mathbf{x} \sim P}[\Phi(\mathbf{x})]$ is the expectation value of the function $\Phi(\mathbf{x})$ (where the samples are drawn from probability distribution P) . It means the mean of a distribution is defined as

$$\mu_{\mathbf{x}}[P] := \mathbf{E}_{\mathbf{x} \sim P}[\Phi(\mathbf{x})], \tag{2.2.10}$$

where Φ is the embedding that $\Phi : \mathcal{X} \to \mathcal{F}, \mathbf{x} \mapsto K(\mathbf{x}, .)$. For all possible g, it has been shown that the further apart the $\hat{P}_1$ and $\hat{P}_2$, the larger the *MMD*.

It has been also shown by Jegelka et al [35] that MMD can be expressed in terms of the dependency between two sets of variables:

$$\begin{aligned}\hat{\pi}_1\hat{\pi}_2\|\mu[\hat{P}_1]-\mu[\hat{P}_2]\|^2_{\mathcal{H}} &= \frac{\hat{\pi}_2}{n^2\hat{\pi}_1}(\alpha^{(1)})^T\mathbf{K}\alpha^{(1)}+\frac{\hat{\pi}_1}{n^2\hat{\pi}_2}(\alpha^{(2)})^T\mathbf{K}\alpha^{(2)}-\frac{2}{n^2}(\alpha^{(1)})^T\mathbf{K}\alpha^{(2)} \\ &= -HSIC(P,\alpha^{(1)})-const=\mathbf{tr}(\mathbf{KL})-const, \qquad (2.2.11)\end{aligned}$$

where $\mathbf{K}$ is the kernel matrix of the data and $\mathbf{L}$ is a matrix whose its entries are defined as

$$l_{ij}=\begin{cases}\frac{1}{n}\sqrt{\frac{n_1}{n}} & \alpha_i^{(1)}=\alpha_j^{(1)} \\ \frac{1}{n}\sqrt{\frac{n_2}{n}} & \alpha_i^{(2)}=\alpha_j^{(2)} \\ 0 & \text{otherwise.}\end{cases}, \qquad (2.2.12)$$

based on binary class labels. Maximizing MMD will maximize the dependency between the data distribution and the cluster assignments.

2.3 Adapting Component Analysis

As discussed in Section 1.2, the main challenge of domain adaptation problem is that of the non-similarity of the probability distributions of the training and test data sets, i.e. $P_{tr}(\mathcal{X},\mathcal{Y})$ and $P_{ts}(\mathcal{X},\mathcal{Y})$. Our proposed method is based on the assumption that having $P_{tr}(\mathcal{Y}|\mathcal{X}) = P_{ts}(\mathcal{Y}|\mathcal{X})$, there exists a new representation of the data, Φ, such that the probability distributions of the training and test data sets are similar:

$$P_{tr}(\Phi)\approx P_{ts}(\Phi), \qquad (2.3.1)$$

where Φ is defined to be

$$\Psi:\mathbf{X}\rightarrow\Phi, \qquad (2.3.2)$$

such that

$$\Phi:=[\Phi_{tr}\Phi_{ts}]. \qquad (2.3.3)$$

Φ_{tr} and Φ_{ts} denote the training and test data sets in the new representation space. Once the appropriate representation is found, we can apply further predictive algorithms to the samples in the new feature space where the joint probability distribution of the training data set and that of the test data set are no longer far apart.

The construction of domain adaptation methods are different. These methods are categorized into unsupervised, semi-supervised, and supervised ones. An unsupervised method is defined as having only the features from training and test data sets for learning. A semi-supervised method has the training and test data sets and also the response variables of the training data set for learning. In supervised algorithms, the response variables of a few test samples are also available. Therefore, the learning procedure benefit from them on top of the training and test data sets, and the response variables of the training data set [17].

Here, we first propose, an unsupervised domain adaptation algorithm. We will call it UnSupervised Adapting Component Analysis (US-ACA). The algorithm exploits only the training and test data sets to find a new representation of the data, Φ, which makes the probability distributions of the training and test data sets as close as possible. Then, we will generalize the algorithm into a semi-supervised domain adaptation algorithm, in which the response variables of the training data set are also exploited to strengthen the US-ACA and this will be called Semi-Supervised ACA (SS-ACA).

The new representation, Φ, in both unsupervised and semi-supervised versions has two properties: 1- the distance between the training and test data set distributions, Φ_{tr} and Φ_{ts}, is minimized and 2- Φ preserves the important structural properties of $\mathbf{X}$. The first property should be satisfied since it is assumed that in the new representation space, the probability distributions are no longer far apart and the second property is essential to have a reasonable performance in any further predictive analysis.

2.3.1 Minimizing The Distance Between $P(\Phi_{tr})$ and $P(\Phi_{ts})$

Assuming Φ is the new representation of data in a new feature space, the distance between the training and test data set probability distributions, $P(\Phi_{tr})$ and $P(\Phi_{ts})$, can be measured based on MMD which was introduced in Section 2.2.2 [35]:

$$Distance(P(\Phi_{tr}), P(\Phi_{ts}) = \|\mathbf{E}[\Phi_{tr}] - \mathbf{E}[\Phi_{ts}]\|^2_{\mathcal{H}}. \tag{2.3.4}$$

It can also be written as:

$$\begin{aligned} \pi_{tr}\pi_{ts}MMD(P(\Phi_{tr}), P(\Phi_{ts})) &= \pi_{tr}\pi_{ts}\|\mu[P(\Phi_{tr})] - \mu[P(\Phi_{ts})]\|_{\mathcal{H}} && (2.3.5) \\ &= \mathbf{tr}(\mathbf{H}\mathbf{L}_M\mathbf{H}\mathbf{L}_\phi) - C && (2.3.6) \end{aligned}$$

where $\mathbf{L}_\Phi = \Phi^T\Phi$. $\mathbf{L}_M$ is a kernel of the new representation of the data. $\mathbf{L}_M$ is the kernel similar to Equation 2.2.12 where the first cluster includes the training samples and the second cluster consists of the samples of the test data set:

$$\boldsymbol{L}_M = \left[\begin{array}{c|c} \alpha\mathbf{1}^{n_{tr}\times n_{tr}} & \mathbf{0}^{n_{tr}\times n_{ts}} \\ \hline \mathbf{0}^{n_{ts}\times n_{tr}} & \beta\mathbf{1}^{n_{ts}\times n_{ts}} \end{array}\right] \tag{2.3.7}$$

where $\mathbf{0}$ and $\mathbf{1}$ are the matrices of all zeros and ones with specified dimensions respectively, and α and β are defined as

$$\begin{aligned} \alpha &= \frac{1}{n}\sqrt{\frac{n_{tr}}{n}} && (2.3.8) \\ \beta &= \frac{1}{n}\sqrt{\frac{n_{ts}}{n}}. && (2.3.9) \end{aligned}$$

2.3.2 Preserving the Properties of X

Although it seems that crucial criterion for solving domain adaptation problem is to make the distance between distributions of the training and the test data sets as small as possible,

this criterion by itself may not be adequate. This is because the new representation should not lose any important and critical information in data for the future predictive analysis.

Figure 2.1 depicts the training and the test data set samples of a binary classification problem. It is obvious that their corresponding joint probability distributions are different. Figure 2.2 shows a new representation of the data that is just estimated based on minimizing the distance between the joint probability distribution of the training data set and that of the test data set. The data is completely mixed and it looses its informative geometry. Therefore, beside minimizing the distance between the aforementioned probability distributions, the new representation should also preserve the important data features that are needed for any post analysis.

A measure is needed for evaluating the second property. It seems the dependency of the original data and its new representation can be a measure that shows how well the structure and important features for predicting the response variables are preserved. As we mentioned, Hilbert-Schmidt independence criteria is a measure for estimating the dependency between two random variables:

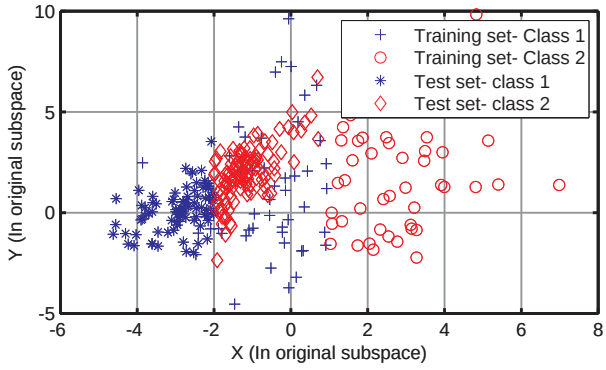

Figure 2.1: The training and test data set samples in the original space.

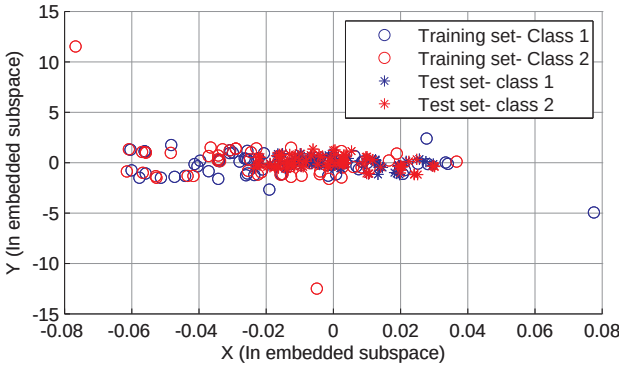

Figure 2.2: The representation of the training and test data set samples in the new feature space.

$$HSIC(\mathbf{X}, \Phi) = (n-1)^{-2}\mathbf{tr}(\mathbf{H}\mathbf{K}_\mathbf{x}\mathbf{H}\mathbf{L}_\Phi), \tag{2.3.10}$$

where $\mathbf{L}_\Phi$ is a kernel over Φ, for example $\Phi^T\Phi$, and $\mathbf{K}_\mathbf{x}$ is a valid kernel on the original data. The choice of the kernel implies the structure that is desired to be preserved. For example one can choose Isomap kernel:

$$\mathbf{K} = \frac{1}{2}\mathbf{H}\mathbf{D}^2\mathbf{H}, \tag{2.3.11}$$

where $\mathbf{D}$ is the geodesic distance matrix and $\mathbf{H}$ is the centring matrix that was defined before in Section 2.2.1. Choosing Isomap kernel will keep the geodesic distance of the data while a linear kernel preserve the Euclidean distance of the data. A list of some common kernels are summarized in Table 2.3.2.

2.3.3 Unsupervised-ACA

To incorporate the two properties, minimizing the distance between $P(\Phi_{tr})$ and $P(\Phi_{ts})$ and preserving the structural properties of $\mathbf{X}$, we propose an unsupervised domain adaptation

Table 2.1: Kernel Functions $\mathbf{K}(\mathbf{x}, \mathbf{y})$

Kernel Type	Mathematical Function	Parameters
Linear	$\mathbf{x}^T\mathbf{y} + c$	c
Polynomial	$(\alpha\mathbf{x}^T\mathbf{y} + c)^d$	α, c
Gaussian	$\exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2})$	σ
Exponential	$\exp(-\frac{\|\mathbf{x}-\mathbf{y}\|}{2\sigma^2})$	σ
Laplacian	$\exp(-\frac{\|\mathbf{x}-\mathbf{y}\|}{\sigma})$	σ
Sigmoid	$\tanh(\alpha\mathbf{x}\mathbf{y}^T + c)$	α, c
Power	$-\|\mathbf{x} - \mathbf{y}\|^d$	-
Log	$-\log(\|\mathbf{x} - \mathbf{y}\|^d + 1)$	-
ANOVA	$\Sigma_{k=1}^n \exp(-\sigma(\mathbf{x}^k - \mathbf{y}^k)^2)^d$	σ, k
Rational Quadratic	$1 - \frac{\|\mathbf{x}-\mathbf{y}\|^2}{\|\mathbf{x}-\mathbf{y}\|^2+c}$	c
Multi-quadratic	$\sqrt{\|\mathbf{x} - \mathbf{y}\|^2 + c^2}$	c
Inverse Multi-quadratic	$\frac{1}{\sqrt{\|\mathbf{x}-\mathbf{y}\|^2+c^2}}$	c
Spline	$\Pi_{i=1}^d 1 + \mathbf{x}_i\mathbf{y}_i + \mathbf{x}_i\mathbf{y}_i \min(\mathbf{x}_i, \mathbf{y}_i)$ $-\frac{\mathbf{x}_i+\mathbf{y}_i}{2}\min(\mathbf{x}_i, \mathbf{y}_i)^2 + \frac{\min(\mathbf{x}_i,\mathbf{y}_i)^3}{3}$	$-$
B-Spline	$\Pi_{i=1}^d \mathbf{B}_{2n+1}(\mathbf{x}_i - \mathbf{y}_i)$	-

algorithm based on the measures defined in the previous Sections 2.3.1 and 2.3.2. The objective function of the proposed algorithm is defined as

$$\textit{maximize} \quad \frac{\mathbf{tr}(\mathbf{H}\mathbf{K}_\mathbf{x}\mathbf{H}\mathbf{L}_\Phi)}{\mathbf{tr}(\mathbf{H}\mathbf{L}_M\mathbf{H}\mathbf{L}_\Phi)}. \tag{2.3.12}$$

The numerator is the measure for estimating the dependency between the samples in the original space and their corresponding representations. Maximizing this measure, will preserve the structure of the data as much as possible. The new representation will keep the structural information of the original data depending on the kernel, $\mathbf{K}_\mathbf{x}$. The simplest and most natural kernel is linear kernel. The linear kernel keeps the pairwise Euclidean

distance globally, but one may choose another kernel as well. The denominator is the measure for the distance between the probability distributions of the training data set and that of the test data set. Minimizing this measure or equivalently, maximizing the inverse of it, makes this distance as small as possible.

Rewriting the optimization problem in terms of Φ we have

$$\textit{maximize} \quad \frac{\mathbf{tr}(\mathbf{H}\mathbf{K_x}\mathbf{H}\Phi^T\Phi)}{\mathbf{tr}(\mathbf{H}\mathbf{L}_M\mathbf{H}\Phi^T\Phi)} = \frac{\mathbf{tr}(\Phi\mathbf{H}\mathbf{K_x}\mathbf{H}\Phi^T)}{\mathbf{tr}(\Phi\mathbf{H}\mathbf{L}_M\mathbf{H}\Phi^T)}. \tag{2.3.13}$$

The objective function in Equation 2.3.12 is invariant with respect to any scaling of Φ, so Φ can be chosen such that the denominator $\mathbf{tr}(\Phi\mathbf{H}\mathbf{L}_M\mathbf{H}\Phi^T)$ is equal to one:

$$\begin{aligned} &\textit{maximize} && \mathbf{tr}(\Phi\mathbf{H}\mathbf{K_x}\mathbf{H}\Phi^T) \\ &\textit{subject to} && \mathbf{tr}(\Phi\mathbf{H}\mathbf{L}_M\mathbf{H}\Phi^T) = 1. \end{aligned} \tag{2.3.14}$$

It turns out that the optimization problem is an instance of Rayleigh quotient and finding the optimal Φ is straight forward as it has a closed-form solution. This corresponds to an eigenvector estimation problem with Φ^T as a matrix of eigenvectors of $\mathbf{K}_x^{-1}\mathbf{L}_M$. The number of selected eigenvectors $d' \leq d$ is the dimensionality of the data in the new feature space. If d' is chosen to be less than d, Φ represents the data $\mathbf{X}$ in a lower dimensional space, which means our method not only handles domain adaptation problems but also can be used as a dimensionality reduction method. The algorithm is described in Algorithm 1 by using Matlab notations. For example, $[\mathbf{U}\ \mathbf{V}] := eigs(\mathbf{S}, dim, \textit{'LR'})$ estimates the first $'dim'$ large eigenvalues of $\mathbf{S}$ and their corresponding eigenvectors.

2.3.4 Semi-Supervised ACA

Exploiting the response variables of the training data set (which could sometimes be easily accessible) is a valuable information that can improve the efficiency of the algorithm. The unsupervised algorithm, US-ACA, described in the previous section does not utilize the

Algorithm 1 Unsupervised ACA

$\mathbf{X}^{d\times n} \leftarrow [\mathbf{X}_{tr}^{d\times n_{tr}}\ X_{ts}^{d\times n_{ts}}]$

$\mathbf{K_x} \leftarrow$ *A kernel on* $\mathbf{X}$. *e.g. Linear kernel*

$\mathbf{L_M} \leftarrow$ *The MMD kernel on two clusters of the training and test data sets*

$\mathbf{H} \leftarrow \mathbf{I} - n^{-1}\mathbf{e}\mathbf{e}^T$

$dim \leftarrow$ *The desired output dimensionality*

$\mathbf{S} := (\mathbf{H}\mathbf{K_x}\mathbf{H})^{-1}(\mathbf{H}\mathbf{L_M}\mathbf{H})$

$[\mathbf{U}\ \mathbf{V}] := eigs(\mathbf{S}, dim, \text{'LR'})$

$\Phi_{tr} := \mathbf{U}(1 : n_{tr}, :).$

$\Phi_{ts} := \mathbf{U}(n_{tr} + 1 : end,\ :)$

$\Phi = [\Phi_{tr}\ \Phi_{ts}]$

return Φ

available labels or response variables of the training data set. Using information of the response variables could be advantageous in finding a new representation of data that is more appropriate for the following predictive analysis. The predictive analysis can be a classification problem or regression which are basically predicting the response variables. Our Semi-supervised ACA is proposed as a variation of US-ACA that takes advantage of the response variables. This algorithm finds an appropriate representation of the data without adding extra complexity to US-ACA algorithm. Since the algorithm is semi-supervised, it is called Semi-Supervised ACA (SS-ACA).

SS-ACA similar to US-ACA aims to reduce the distance between training and test data set probability distributions while preserving the structural properties of the data set. However, one does not need to keep the whole structure of the data unchanged. We only need to keep the informative features for predicting the response variables. Some of the structural properties or hidden features of the data may not be important for predicting a certain response variable while they may be very important for another one. For

example, imagine we have different photos of men with beard and glasses. The task A is to categorize them based on having or not having beard and the task B is to categorize them based on having or not having glasses. In task A the features relevant to having glasses are not relevant, but in task B, they are completely relevant. So, it is wise to consider the target which might be a discrete label in classification problem or a continues variable in regression, for finding the important structure or hidden features of the data. Therefore, instead of finding a new representation which preserves the data structure in the unsupervised ACA, we would like to find a new representation which preserves structure and important features of the data relevant to the following predictive model in a supervised manner.

Based on US-ACA, minimizing $\mathbf{tr}(\Phi\mathbf{H}\mathbf{L}_\mathbf{M}\mathbf{H}\Phi^T)$ moves the training and the test data sets probability distributions toward each other while maximizing $\mathbf{tr}(\Phi\mathbf{H}\mathbf{K}_\mathbf{X}\mathbf{H}\Phi^T)$ is the objective function that keeps the structure of the data. Rewriting $\mathbf{K}_\mathbf{X}$ based on linear kernel, we have

$$\begin{aligned}
\mathbf{K}_\mathbf{X} = \mathbf{X}^T\mathbf{X} &= \begin{bmatrix} \mathbf{X}_{tr}^T \\ \mathbf{X}_{ts}^T \end{bmatrix} [\mathbf{X}_{tr}\ \mathbf{X}_{ts}] \\
&= \begin{bmatrix} \mathbf{X}_{tr}^T\mathbf{X}_{tr} & \mathbf{X}_{tr}^T\mathbf{X}_{ts} \\ \mathbf{X}_{ts}^T\mathbf{X}_{tr} & \mathbf{X}_{ts}^T\mathbf{X}_{ts} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{tr}} & \mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} \\ \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}} & \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{ts}} \end{bmatrix},
\end{aligned} \tag{2.3.15}$$

where $\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{tr}}$ and $\mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{ts}}$ involve the information of the structure of training and test data sets receptively. These two sub-matrices are important for the learning or training of the predictive model (which is the main goal) and they should be preserved. But the relative structure of the training data set and the test data set need not to be necessarily fixed. This can help us modifying the structure of the data in a supervised manner with the known response variables of the training data set. So the matrix $\mathbf{K}_\mathbf{X}$ can be changed

to $\hat{\mathbf{K}}_{\mathbf{X}}$ where $\hat{\mathbf{K}}_{\mathbf{X}}$ is constructed based on the data and the known response variables. For example, $\mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}}$ is initially representing the correlation of the training data set and the test data set samples. But if two samples of the training data set are highly correlated based on their response variables, then they should not be different from a test data set sample perspective. So the sub-matrices of $\mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}}$ and $\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{ts}}$ can be smoothed by a process which reduces the variation of the data in unrelated dimensions while it keeps the variance of the data along the directions which contain important information relevant to predicting the response variables. Therefore, those two sub-matrices, $\mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}}$ and $\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{ts}}$ are substituted with the following matrices:

$$\hat{\mathbf{K}}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} = \mathbf{K}_{\mathbf{Y}_{tr}}\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} \tag{2.3.16}$$

$$\hat{\mathbf{K}}_{\mathbf{X}_{ts}\mathbf{X}_{tr}} = \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}}\mathbf{K}_{\mathbf{Y}_{tr}}, \tag{2.3.17}$$

where $\mathbf{K}_{\mathbf{Y}_{tr}}$ is a kernel on the response variables of the training data set $\mathbf{X}_{tr}$ which represents the similarity between the labels of the training data set samples and its main role is to even out the difference between similar samples. Based on this formulation, the sample of the training data set, $\mathbf{x}_i$ is changed to the weighted mean of its similar samples. The weight is proportional to the similarity of sample $\mathbf{x}_i$ and $\mathbf{x}_j$ (that is the (i, j)th entry of the kernel $\mathbf{K}_{\mathbf{Y}_{tr}}$). This makes the variance of similar samples smaller. Therefore, $\mathbf{K}_{\mathbf{X}}$ is changed to $\hat{\mathbf{K}}_{\mathbf{X}}$ which is rewritten as

$$\hat{\mathbf{K}}_{\mathbf{x}} = \begin{bmatrix} \mathbf{K}_{\mathbf{X}_{tr}\mathbf{x}_{tr}} & \hat{\mathbf{K}}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} \\ \hat{\mathbf{K}}_{\mathbf{X}_{ts}\mathbf{x}_{tr}} & \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{ts}} \end{bmatrix} \tag{2.3.18}$$

The steps of the semi-supervised algorithm is summarized in Algorithm 2. Matlab notations are used for simplicity.

Algorithm 2 Semi-Supervised ACA

$\mathbf{X}^{d\times n} \leftarrow [\mathbf{X}_{tr}^{d\times n_{tr}}\ \mathbf{X}_{ts}^{d\times n_{ts}}]$

$\mathbf{L_M} \leftarrow$ *The MMD kernel on two clusters of the training and test data sets*

$\hat{\mathbf{K}}_{\mathbf{X}_{ts}\mathbf{X}_{tr}} \leftarrow \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}}(\mathbf{K}_{\mathbf{Y}_{tr}})$

$\hat{\mathbf{K}}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} \leftarrow (\mathbf{K}_{\mathbf{Y}_{tr}})\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{ts}}$

$\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{tr}} \leftarrow$ *A kernel on* $\mathbf{X}_{tr}$. *e.g. Linear kernel*

$\mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{ts}} \leftarrow$ *A kernel on* $\mathbf{X}_{ts}$. *e.g. Linear kernel*

$$\hat{\mathbf{K}}_{\mathbf{X}} \leftarrow \begin{bmatrix} \mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{tr}} & \hat{\mathbf{K}}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} \\ \hat{\mathbf{K}}_{\mathbf{X}_{ts}\mathbf{X}_{tr}} & \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{ts}} \end{bmatrix}$$

$\mathbf{S} := (\mathbf{H}\hat{\mathbf{K}}_{\mathbf{X}}\mathbf{H})^{-1}(\mathbf{H}\mathbf{L}_{\mathbf{M}}\mathbf{H})$

$[\mathbf{U}\ \mathbf{V}] := eigs(\mathbf{S}, dim, \text{'LR'})$

$\Phi_{tr} := \mathbf{U}(1 : n_{tr}, :)$.

$\Phi_{ts} := \mathbf{U}(n_{tr} + 1 : end,\ :)$

$\Phi = [\Phi_{tr}\ \Phi_{ts}]$

return Φ

Chapter 3

Experimental Result

Domain adaptation problem has been studied extensively in the past decades and several methods have been developed. As described in the previous chapter, one of our proposed algorithms, US-ACA, is unsupervised, while the other one, SS-ACA, is a semi-supervised algorithm which takes advantage of the response variables of the training data set. In this chapter, US-ACA and SS-ACA are compared with semi-supervised algorithms, CODA [36] and MMDE [37].

MMDE has been developed as an unsupervised and semi-supervised algorithm but our algorithms are compared with its semi-supervised version. MMDE basically learns a kernel of the embedded data based on four constraints/objectives simultaneously: (i) the distance between the source domain (training data set) probability distribution and that of the target domain (test data set) is minimized, (ii) the pairwise distance of the data samples is preserved locally (The choice of keeping pairwise distance of some samples differ in unsupervised and semi-supervised versions.), (iii) the embedded data is centered, and (iv) the variance of the data is maximized. The optimization problem has been written as SemiDefinite Program (SDP). After obtaining the kernel, Principle Component Analysis (PCA) is applied to the kernel to get the new low-dimensional representation of the data.

The classifier is learned based on the new representation of the data to predict the labels of the source domain (test data set) [37].

CODA learns a target predictor by maintaining and growing the source domain (training data set) (i.e. iteratively adding target (test data set) features and samples that are confident according to the current algorithm.) In each iteration two adapted logistic regression classifiers have been trained based on two mutually exclusive views where the co-training is effective. Then the samples with exactly one confident classifier, are moved to the source domain. These samples are classified correctly and have the potential to improve the classifier in next iterations [36].

In general, the performance of an unsupervised algorithm will rarely beat the performance of a supervised algorithm. However, to have a better overview, their results are presented together. It will be shown that US-ACA, as an unsupervised algorithm, improves the performance of the algorithm, although it is not significant. Also, SS-ACA as a supervised algorithm has a better performance with respect to CODA and MMDE. The proposed algorithms can also be considered as a dimension reduction technique since they reach the highest efficiency in lower dimensions and the related results are depicted later.

3.1 The kernel on the response variables

In the objective function defined by Equation 2.3.12, $\mathbf{K_X}$ and $\mathbf{L_M}$ are assumed to be known. Then the kernel $\mathbf{K_X}$ has been changed to $\hat{\mathbf{K}}_\mathbf{X}$ in SS-ACA. $\hat{\mathbf{K}}_\mathbf{X} = \begin{bmatrix} \mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{tr}} & \hat{\mathbf{K}}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} \\ \hat{\mathbf{K}}_{\mathbf{X}_{ts}\mathbf{X}_{tr}} & \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{ts}} \end{bmatrix}$ where $\hat{\mathbf{K}}_{\mathbf{X}_{ts}\mathbf{X}_{tr}} = \mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}}(\mathbf{K}_{\mathbf{Y}_{tr}})$ and $\hat{\mathbf{K}}_{\mathbf{X}_{tr}\mathbf{X}_{ts}} = (\mathbf{K}_{\mathbf{Y}_{tr}})\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{ts}}$. $\mathbf{K}_{\mathbf{Y}_{tr}}$ represents the similarity of the response variables of the training data set and can be constructed in various forms. Without loss of generality, we assume that our data is categorized into c classes or c unique labels such that $\forall i, y_i \in \{1, .., c\}$. We choose a kernel of the form $\mathbf{K}_{\mathbf{Y}_{tr}} = \mathbf{B}\mathbf{B}^T$,

where $\mathbf{B}$ is defined as:

$$\mathbf{B} = [\mathbf{b}_1, ..., \mathbf{b}_c], \tag{3.1.1}$$

such that $\mathbf{b}_i$ is $\frac{1}{\sqrt{n_i}}$ and n_i is the number of samples in class i. For example if there are five data points such that the first two data points are in class 1 and the last three data points are in class 2, then $\mathbf{B}$ and $\mathbf{K}_{\mathbf{Y}_{tr}}$ can be formed as follows:

$$\mathbf{B} := \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{3}} \\ 0 & \frac{1}{\sqrt{3}} \\ 0 & \frac{1}{\sqrt{3}} \end{bmatrix}, \mathbf{K}_{\mathbf{Y}_{tr}} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}. \tag{3.1.2}$$

Multiplication of the kernel, $\mathbf{K}_{\mathbf{Y}_{tr}}$, with $\mathbf{K}_{\mathbf{X}_{ts}\mathbf{X}_{tr}}$ and $\mathbf{K}_{\mathbf{X}_{tr}\mathbf{X}_{ts}}$ is basically substituting each sample of the training data set by the weighted mean of its corresponding similar samples which makes the variation of the data along similar samples smaller.

3.2 Toy Classification Example

We first test our algorithm on a toy example. The training data set consists of 100 samples of the two dimensional data drawn from a multivariate normal distribution in which the mean is $\mu_{tr} = (-1, 3)$ and the covariance matrix is $\sigma_{tr} = \begin{bmatrix} 2 & 0.25 \\ 0.25 & 1 \end{bmatrix}$. The training data set is categorized in two classes. The samples whose first feature values are smaller than -1 belong to the first class, and the ones that their first feature values are larger than -1 belong to the second class. The test data set consists of 200 samples from a different multivariate normal distribution whose distributional properties are $\mu_{ts} = (2, 1)$ and $\sigma_{ts} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 3 \end{bmatrix}$. The test data set is also categorized in two classes based on their

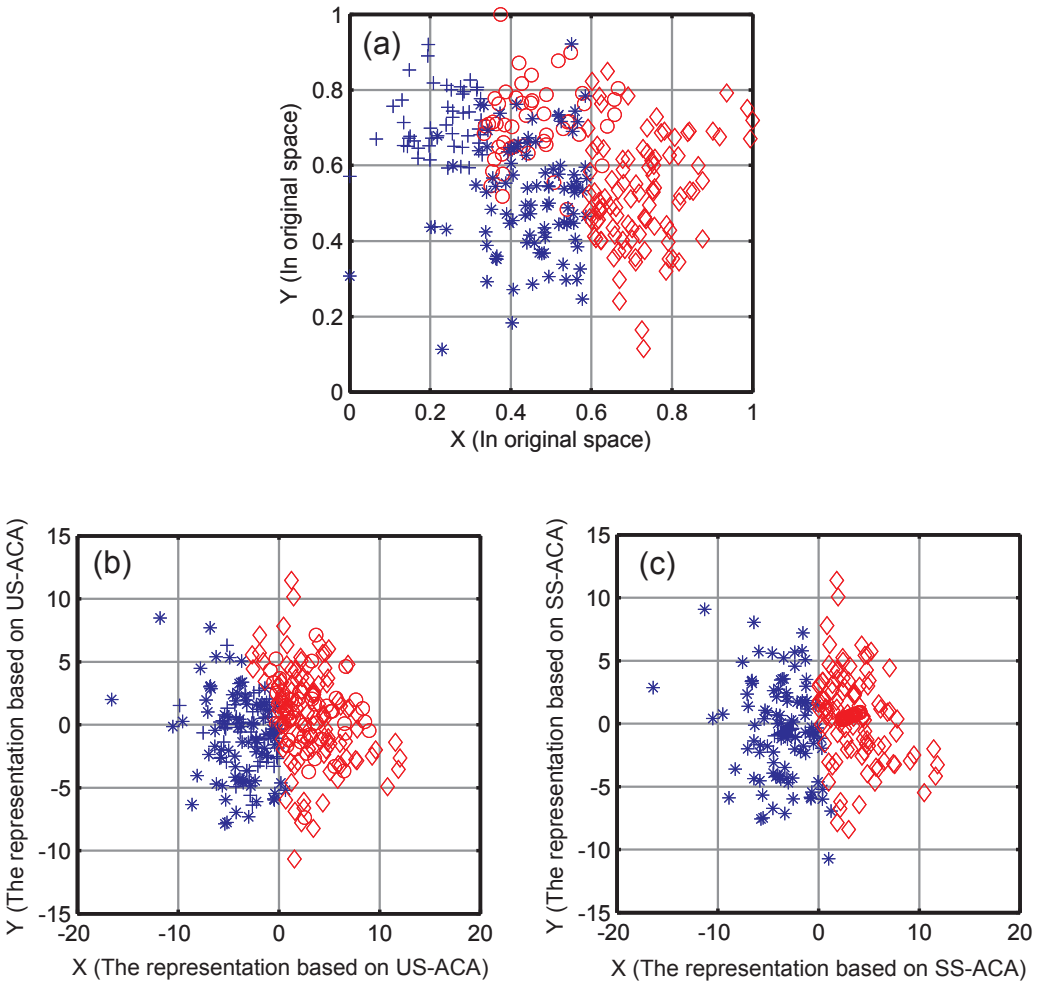

Figure 3.1: (a) 2-dimensional data in the original space. Circles, $\circ$, and crosses, $\times$ are two classes of the training data set and diamonds,$\diamond$, and stars, $*$, are two classes of the test data set. (b) The new representation of the data in new feature space based on US-ACA and (c) The new representation of the data in new feature space based on SS-ACA

Table 3.1: The error rate of the SVM on the original data and on the modified data of US-ACA and SS-ACA

Algorithms	Error rate
Original–SVM	41%
US-ACA –SVM	5.5%
SS-ACA –SVM	2.5%

first features. A sample belongs to the first class if its first feature is smaller than 2, and it belongs to the second class if it is larger than 2. The values -1 and 2 are the means of the training and test data set distributions along the first feature space.

Figure 3.1-a demonstrates the data in the original feature space without any changes and Figures 3.1-b and 3.1-c are the embedded data where US-ACA and SS-ACA algorithms are applied respectively. As it can be seen, the distance between the embedded training and test data set distributions is reduced. Consequently, the new training data samples are better representatives of the test data set. Applying the algorithms and getting the embedded data, we can classify them using SVM [38] or any other classifier. As shown in the Table 3.2, SVM can classify the data with 41% error rate for the original data. By applying US-ACA and SS-ACA to the original data, the error rate is decreased to 5.5% and 2.5% for this particular sample of data.

The proposed algorithms are also compared with MMDE and CODA on the toy example. US-ACA, SS-ACA is applied to the data and 1-NN has been used as a classifier. The classifier error rate in each cases is depicted in Figure 3.2. The error rate is the mean of the number of misclassified samples. To evaluate the efficiency of these methods, we have also classified the original data without any changes using 1-NN and estimate its corresponding error rate as the baseline. As It is shown in Figure 3.2, US-ACA and SS-ACA provide

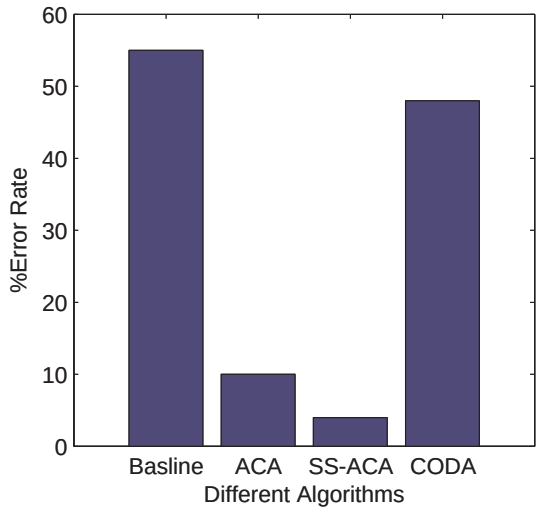

Figure 3.2: The error rate comparison for different algorithms and the baseline on a toy example

significant improvement in the error rate of the classification process.

3.3 Real world data sets

In this section, we test the proposed methods on different real world data sets varying from images to text and biological data.

The First data set which is a collection of images, is MNIST handwritten digits [39]. This data set consists of 8-bit gray scale images of the digits between "0" and "9". The domain adaptation problem is defined as distinguishing two different digits of the test data set (e.g. 3 and 4) while the algorithm is trained for distinguishing two other digits (e.g. 1 and 2). To test the algorithms using MNIST data set, we generate different data sets

Table 3.2: Different data set generated from MNIST database

NAME	TRAINING SET	TEST SET
DIGITS(1)	$1, 2$	$3, 4$
DIGITS(2)	$1, 2$	$7, 8$
DIGITS(4)	$9, 0$	$6, 8$
DIGITS(7)	$8, 4$	$2, 3$
DIGITS(3)	$3, 5$	$2, 6$
DIGITS(5)	$3, 8$	$1, 5$
DIGITS(6)	$7, 6$	$0, 9$

called Digits(1) to Digits(7) which are shown in Table 3.2. The training data set digits in each data set is showing the two digits that the algorithm is learned to classify them, while the goal is to classify the digits of the test data set. These data sets are randomly chosen among all possible cases. The number of the training and test samples are 300 and 500 respectively. The size of the training and test data sets are fixed for all of the data sets in Table 3.2. We have compared the error rate on the data sets defined in Table 3.2 for different algorithms and the baseline in Figure 3.3. The numbers (1) to (7) in Figure 3.3 stands for the *Digits*(1) to *Digits*(7) data sets respectively.

The dimensionality of the output data in US-ACA and SS-ACA is set to 2. For instance, the new representation of the data of Digits(1) in 2-dimensional space is depicted in Figure 3.4(a) and (b) based on US-ACA and SS-ACA. To make a fair comparison, the dimensionality of the output features is 2 and it is constant through all experiments in this section. It is shown in Figure 3.3 that SS-ACA outperforms in comparison with the other algorithms. We observe that in the case of SS-ACA, the error rate is on average decreased to 10% approximately which is considerably less than the error rate of CODA and MMDE. Notice that US-ACA is an unsupervised algorithm and does not consider the information

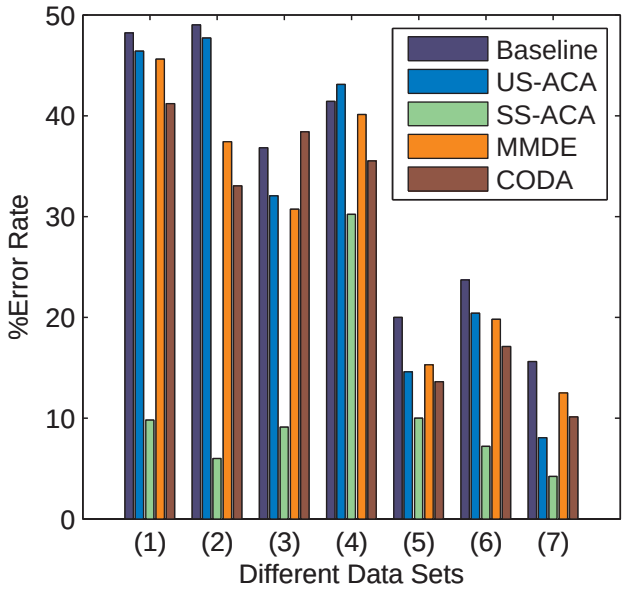

Figure 3.3: The error rate comparison for different algorithms and baseline on the data sets generated from MNIST data base. (1) to (7) on the X-axis stands for Digits(1) to Digits(7) respectively

of the response variables of the training data set. Therefore, it is not expected to have a performance as good as the algorithms which are taking advantage of the training data set response variables. However, the result of US-ACA is reasonable with respect to the other methods.

The second data, the 20 Newsgroups data set, consists of about 20,000 newsgroup text documents, categorized almost evenly across 20 different newsgroups based on their subjects. Some newsgroups can use common words and are related to each other (e.g.newsgroups of IBM hardware and Mac hardware are similar topics), while the others can use differ-

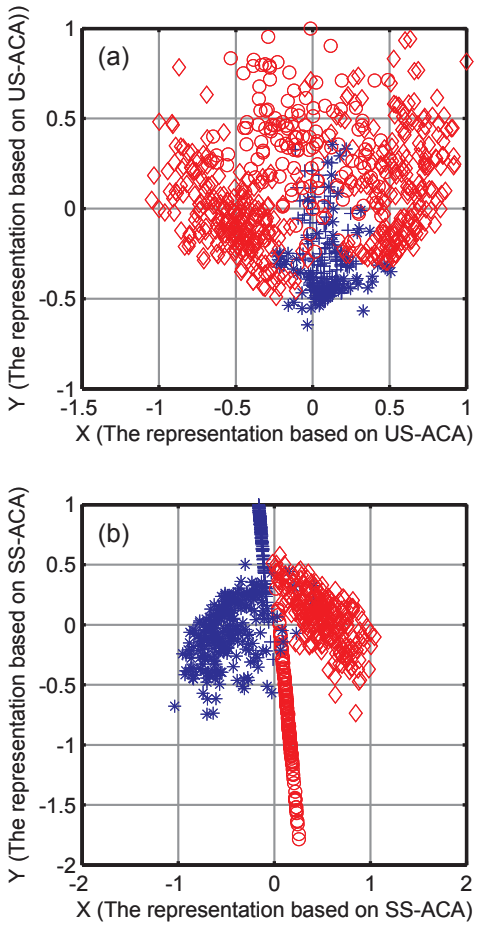

Figure 3.4: (a) The 2-dimensional representation of Digits(1) data set based on US-ACA. (b) The 2-dimensional representation of the same data based on SS-ACA. Circles, $\circ$, and crosses, $\times$ are two classes of the training data set. Diamonds,$\diamond$, and stars, $*$, are two classes of the test data set after applying 1-NN.

ent language (e.g newsgroup about Ads for sale and a religious topic newsgroup are not similar). This data set is recategorized into four groups based on similar topics. The new

version is binary occurrence of the data for 100 words across 16242 posting [40]. In order to have a domain adaptation problem, we generate three data sets from this new version of the 20newsgroups data in which the data is categorized in 4 groups. "Newsgroup1" data set consists of 1000 randomly selected postings from groups 1 and 2 as the training data set, and 2000 randomly selected postings from groups 3 and 4 as the test data set. Similarly, "Newsgroup2" and "Newsgroup3" data sets have the same number of postings randomly selected from groups $1, 3$ and $1, 4$ in their training data set and $2, 4$ and $2, 3$ in their test data set respectively. In each of the artificially generated data set, the task is to classify the postings of the test data set while the algorithm is learned based on the training data set.

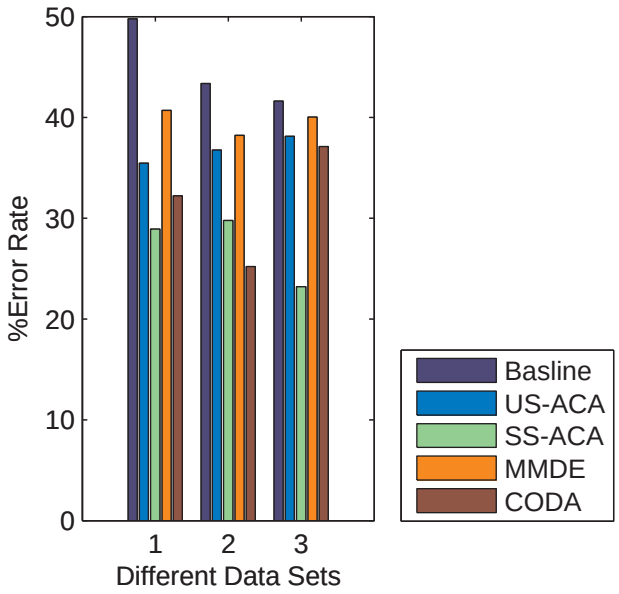

Figure 3.5: The error rate comparison for different algorithms in three data sets. 1, 2 and 3 on the X-axis stands for Newsgroup1, Newsgroup2 and Newsgroup3 data sets respectively.

We compare our proposed methods, US-ACA and SS-ACA with MMDE and CODA on the Newsgroup1, Newsgroup2 and Newsgroup3 data sets. Their corresponding error rates are compared with baseline in Figure 3.5. The error rate is the average error over 10 trials where in each trial the samples are randomly chosen from the original data set. As it is depicted in Figure 3.5, the error rate has been decreased from almost 50% to approximately $25\% - 30\%$ based on SS-ACA. SS-ACA outperforms the other methods except in the second database which is Newsgroup2. For Newsgroup2 the CODA has a slightly better error rate, and that could be partly because the 2-dimensional representation of the data is not appropriate in this case or, because CODA is initially designed to solve domain adaptation problem in text. That is because CODA focused on domain adaptation problems that are characterised by missing features, and this is often the case in natural language processing. However, our algorithm is not developed for a specific type of data.

To test the performance of the proposed algorithms on different types of data sets, we run a set of classification experiments on several UCI data sets [41] in which they are biased artificially. To make an artificial biased data set, the data is randomly divided in to the training and test data sets. Then an additional variable, s_i, for each sample of training data set is defined [21, 14]. s_i is set to depend only on one of the sample features, therefore, the biasing procedure is called, simple bias [6]. This additional variable determines whether the corresponding sample is contributing in the biased training data set or not. It means if $s_i = 1$, then the ith sample is included in the biased training data set, otherwise it is excluded. There is also a parameter called *Biasing Ratio*. It determines the percentage of the samples with $s_i = 1$ that are included in or the percentage of the samples with $s_i = 0$ that are excluded from the training data set. The *Biasing Ratio* is 100%, if all the samples with $s_i = 1$ are in and all the samples with $s_i = 0$ are out of the training data set.

The Breast Cancer dataset from the UCI Archive [41] is a biological data set. The data includes 699 examples from 2 classes: benign (positive label) and malignant (negative label). This is a binary classification problem from 9 initial features.

The performance of the US-ACA and SS-ACA are compared with the baseline, MMDE and CODA in Figure 3.6. The X-axis is the feature number that the additional variable s_i depends on it. We repeat the experiment with different *Biasing Ratio*s equal to 70%, 80% and 90%. All the results are depicted in left column of Figure 3.6. As can be seen, SS-ACA has better performance with respect to the other methods.

Another parameter for showing the efficiency of a method is Normalized Improvement (NI) which quantifies how much algorithm A outperforms with respect to the algorithm B. This parameter is estimated as

$$NI = \frac{|Error_A - Error_B|}{Error_A}. \tag{3.3.1}$$

On the right column of Figure 3.6 the Normalized Improvement of SS-ACA with respect to the baseline is shown. As can be seen after adapting the domains of training and test data sets, the performance is improved approximately up to 50% with respect to the baseline in some features.

Wine, German Credit, India diabetes and Ionosphere are the other data sets from UCI archive [41] where their biasing process is as explained above, and their biasing ratio is 80%. The number of training and test data set samples, the biased feature and also the number of classes in each data set is depicted in Table 3.3. The error rate of different methods on these data sets are also available on Table 3.3. It shows that SS-ACA outperforms the other methods in these data sets as well.

As it is mentioned earlier, SS-ACA can also be used as a dimension reduction technique. We run the SS-ACA algorithm on different data sets. For Digits(1), Digits(2) and Digits(3) data sets, the error rates versus the output dimension which varies from 1 to 784 is depicted in Figure 3.7(a). 784 is the dimensionality of the data in original space. The error rate for Wine data set is depicted in Figure 3.7(b). In this data set, although variation of dimension does not change the efficiency, the algorithm keeps its performance in low dimensions. The

Table 3.3: Test result on various data sets by different methods. 1-nearest neighbour has been used for classification.

Data set	(n_{tr}, n_{ts})	Dim.	#Feature	Classes	Baseline	SS-ACA	MMDE	CODA
Wine	(47,156)	13	8	3	39.44%	30.99%	48.26%	31.98%
German Credit	(213,600)	24	9	2	41.50%	30.06%	40.62%	32.48%
India diabetes	(92,568)	8	3	2	42.13%	38.01%	40.71%	40.35%
Ionosphere	(64,201)	32	8	2	24.61%	22.29%	26.71%	20.50%

changes of the error rate along different dimensions of the Newsgroup2 is also demonstrated in Figure 3.7(c). The error rate is minimum when the dimension of the data is about 15-25 in this case. As can be seen the algorithm has a good performance in low dimensions. So it can be considered as a dimension reduction technique as well. The appropriate dimension in each data set can be calculated by cross validation.

The running time of the proposed method is less than MMDE and CODA. The MMDE optimization problem is modeled as a semidefinite program and CODA is an iterative method that both consume a lot of time. However, the proposed methods has a closed-form solution and it is faster than MMDE and CODA.

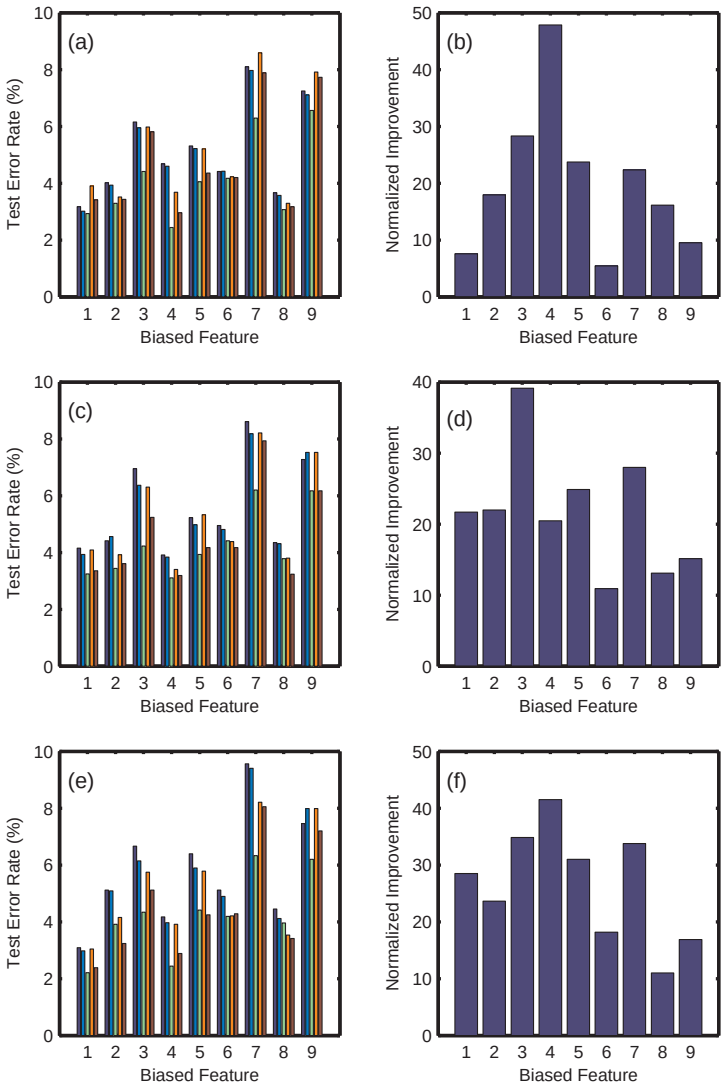

Figure 3.6: (a), (c) and (e) are the error rate comparison in different algorithms on Breast Cancer data set with *Biasing Ratio* of 70%, 80% and 90% respectively. The X-axis is showing the features which the biasing process is based on. (b), (d) and (e) are the Normalized Improvement of SS-ACA with respect to the baselines of (a), (c) and (e) respectively. The bars from left to right correspond to Basline, US-ACA, SS-ACA, MMDE and CODA.

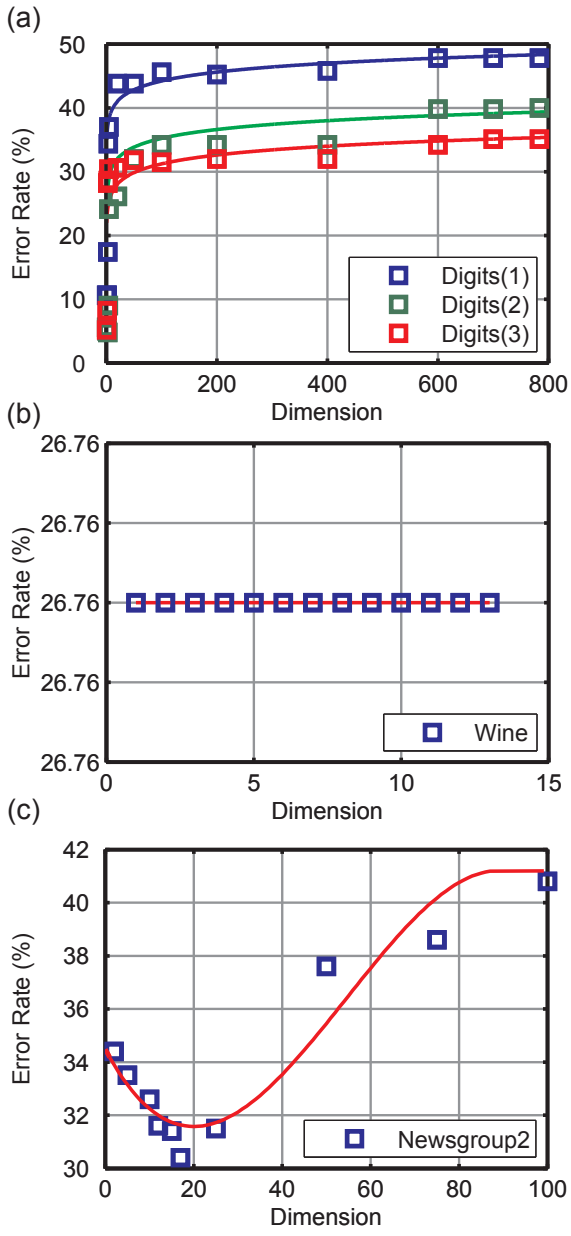

Figure 3.7: (a) The error rate changes of SS-ACA vs. different dimensions on Digits(1), Digits(2) and Digits(3) data sets. (b) The error rate changes of SS-ACA vs. different dimensions on Wine data set. (c) The error rate changes of SS-ACA vs. different dimensions on Newsgroup2 data set.

Chapter 4

Conclusion

We have presented two domain adaptation algorithms in which the data samples are transferred to a new feature space. The new representation of the data is explored such that the training and the test data sets in the new feature space are as close as possible while the important structural information of the data is preserved. In order to solve this problem and satisfy the aforementioned properties, we have defined an optimization problem such that its solution is known to be eigenvectors of a given matrix. Our experimental results show that the algorithm performs well in practice and has a good efficiency in lower dimensions, so it can be used as a dimensionality reduction technique.

To keep the important structure of the data, we have used a specified kernel over the data which is modified to involve the information of the response variables of the training data beside taking advantage of the data itself. The proposed kernel is useful for classification, but an immediate future direction can be investigating other appropriate kernels that can utilize the response variable information beside the structural information of the data (such that the algorithm is applicable for any predictive model tasks).

One of the advantages of the proposed method is that the objective function of our

optimization problem is independent of the classifier. It implies any classifier can be used for classifying the new representation of the data after applying SS-ACA. However the performance of the classifiers can be improved, if we could develop an optimization problem that simultaneously minimizes the error rate of a predictive model in addition to satisfying our current objective functions.

References

[1] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F. and Vaughan, J. W. A Theory of Learning from Different domains. *Machine Learning Journal*, 79(1-2): 151–175, 2010.

[2] Huang, J. Smola, A. J., Gretton, A., Borgwardt, K. M. and Scholkopf, B. Correcting Sample Selection bias by Unlabeled Data. In Scholkopf, J. P. and Hoffman, T. (ed.), *Advances in Neural Information Processing System*, chapter 19, pp. 601–608. MIT Press, Cambridge, MA, 2007.

[3] Liu, Q., Mackey, A., Roos, D. and Pereira, F. Evigan: A Hidden Variable Model for Integrating Gene Evidence for Eukaryotic Gene Prediction. *Bioinformatics*, 24(5): 597–605, 2008.

[4] Jiang, J. A literature Survey on Domain Adaptation of Statistical Classifier. 2008. ($http://sifaka.cs.uiuc.edu/jiang4/domain_adaptation/survey/da_survey.pdf$)

[5] Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

[6] Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., and Schoelkopf, B., Covariate Shift by Kernel Mean Matching In *Dataset Shift in Machine Learning,*

Covariate Shift and Local Learning by Distribution Matching, J. Quionero-Candela and M. Sugiyama and A. Schwaighofer and N. Lawrence (Eds.), MIT Press, Cambridge, MA pp.131–160, 2008.

[7] Japkowicz, N. and Stephen, S. The Classic Imbalance Problem: A Systematic Study. *Intelligent Data Analysis*, 6(5):429–450, 2002.

[8] Lin, Y., Lee, Y., and Wahba, G,. Support vector machines for classification in non-standard situations. *Machine Learning*, 46(1-3):191-202, January 2002.

[9] Chan, Y. S., and Ng, H. T. Word sense disambiguation with distribution estimation. In *Proceed- ings of the 19th International Joint Conference on Articial Intelligence*, pp. 1010-1015, Edingurgh, Scotland, July 2005.

[10] Zhu, X. Semi-supervised learning literature survey. *Technical Report 1530, University of Wisconsin- Madison*, 2005.

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

[12] Caruana, R. Multi-task learning. *Machine Learning*, 28(1):4175, July 1997.

[13] Ben-David, S., and Schuller, R. Exploiting task relatedness for multiple task learning. In *Proceedings of the 16th Annual Conference on Learning Theory*, Washington D.C., USA, August 2003.

[14] Rosset, S., Zhu, J., Zou, H., and Hastie, T. A method for inferring label sampling mechanisms in semi-supervised learning. In *Advances in Neural Information Processing Systems 17*, 2004.

[15] Xue, Y., Liao, X., Carin, L., and Krishnapuram, B. Multi-task learning for classification with Dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

[16] Micchelli, C. A., and Pontil, M. Kernels for multi-task learning. *In Saul, L. K., Weiss, Y., and Bottou, L., editors, Advances in Neural Information Processing Systems 17* pp. 921–928, MIT Press, Cambridge, Massachusetts, USA, 2005.

[17] Daume III, H., Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pp. 256-263, Prague, Czech Republic, June 2007.

[18] Dai, W., Xue, G., Yang, Q., and Yu, Q. Transferring naive bayes classiers for text classification. In *Proceedings of the 22nd AAAI Conference on Articial Intelligence*, pp. 540-545, Vancouver, British Columbia, Canada, July 2007.

[19] Xing, D., Dai, W., Xue, G., and Yu. Y. Bridged renement for transfer learning. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 324-335, Warsaw, Poland, September 2007.

[20] Heckman, J. J. Sample Selection Bias as a Specification Error. *Econometrica*, 47(1): 153–161, 1979.

[21] Zadrozny, B. Learning and Evaluating Classifiers Under Sample Selection Bias. In *Proceedings of the 21th Annual International Conference on Machine Learning (ICML 2004)*, pp. 114–121, Banff, CA, 2004.

[22] Chan, Y. S., Ng, H. T. Estimating Class Priors in Domain Adaptation for Word Sense Disambigution. In *Proceedings of the 21th International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 89–96, Sydney, Australia, 2006.

[23] Sugiyama, M. and Muller, K. Input-dependent estimation of generalization error under covariate shift. *Statistics and Decisions*, 23(4):249-279, 2005.

[24] Bickel, S., and Scheffer, T. Dirichlet-enhanced spam filtering based on biased samples. In *B. Scholkopf, J. Platt, and T. Hoffman, editors, Advances in Neural Information Processing Systems 19*, pp. 161-168. MIT Press, Cambridge, Massachusetts, USA, 2007.

[25] Jiang, J. and Zhai, C. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pp. 254–271, Prague, Czech Republic, June 2007.

[26] Daume III, H., Kumar, A. and Saha, A. Co-regularization Based Semi-supervised Domain Adaptation. In *Proceedings of the Conference on Neural Information Processing Systems(NIPS 2010)*, pp. 1–9, Vancouver, Canada, 2010.

[27] Xue, G., Dai, W., Yang, Q. and Yu, Y. Topic-bridged PLSA for Cross-domain Text Classification. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2008)*, pp. 627–634, Singapore, 2008.

[28] Blitzer, J., McDonald, R., and Pereira, F. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pp. 120-128, Sydney, Australia, July 2006.

[29] Satpal, S., and Sarawagi, S. Domain adaptation of conditional probability models via feature subsetting. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 224-235, Warsaw, Poland, September 2007.

[30] Daume III, H., and Marcu, D. Domain adaptation for statistical classifiers. *Journal of Articial Intelligence Research*, 26:126, May 2006.

[31] Storkey, A. J., and Sugiyama, M. Mixture regression for covariate shift. In *B. Scholkopf, J. Platt, and T. Hoffman, editors, Advances in Neural Information Processing Systems 19*, pp. 1337-1344. MIT Press, Cambridge, Massachusetts, USA, 2007.

[32] Dai, W., Yang, q., Xue, G. R., and Yu, Y. Boosting for transfer learning. In *Proceedings of the 24th Annual International Conference on Machine Learning*, pp. 193-200, Corvallis, Oregon, USA, June 2007.

[33] Li, X., Blimes, J. A Bayesian divergence prior for classifier adaptation In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rice, March 2007.

[34] Gretton, A., Bousquet, O., Smola, A. J., and Scholkopf, B. Measuring Statistical Dependence Between Hilbert-Schmidt Norms . *Proceedings of Algorithmic Learning Theory (ALT)*, 227:63–77, 2005.

[35] Jegelka, S., Gretton, A., Scholkopf, B., Sriperumbudur, B. K., and Luxburg., U. V. Generalized clustering via kernel embeddings. *Proceedings of the 32nd annual German conference on Advances in artificial intelligence (KI 09)*, pp. 144–52, 2009.

[36] Chen, C., Weinberger, K. Q., and Blitzer, J. C. Co-training for domain adaptation. In *J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, Advances in Neural Information Processing Systems 24 (NIPS-24)*, pp. 2456-2464. 2011.

[37] Pan, S. J., Kwok, J. T., and Yang, Q. Transfer Learning via Dimensionality Reduction. In *Proceedings of AAAI*, pp. 677–682, Chicago, Illinois, USA, 2008.

[38] http://www.csie.ntu.edu.tw/ cjlin/libsvm/

[39] http://yann.lecun.com/exdb/mnist/

[40] http://www.cs.nyu.edu/ roweis/data.html

[41] http://archive.ics.uci.edu/ml

[42] Chelba, C. and Acero, A. Adaptation of maximum entropy capitalizer: Little data can help a lot. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pp. 285–292, Barcelona, Spain, 2004.

[43] Jiang, J., and Zhai, C. X. A two-stage approach to domain adaptation for statistical classifiers. In *Proceedings of the ACM 16th Conference on Information and Knowledge Management*, pp. 401-410, 2007.