F. Dorri, A. Ghodsi. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY : 1– Oct. 2013

Minimizing the discrepancy between source and target domains by learning adapting components

Fatemeh Dorri, Ali Ghodsi

Department of Computer Science, University of British Columbia
Department of Statistics and Actuarial Science, University of Waterloo

E-mail: fdorri@cs.ubc.ca, aghodsib@uwaterloo.ca

Revised 29 Oct 2013

Abstract Predicting the response variables of the target data set is one of the main problems in machine learning. Predictive models are desired to perform satisfactorily in a broad range of target domains. However, that may not be plausible if there is a mismatch between the source and target domain distributions. The goal of domain adaptation algorithms is to solve this issue and deploy a model across different target domains. We propose a method based on kernel distribution embedding and Hilbert-Schmidt Independence Criterion (HSIC) to address this problem. The proposed method embeds both source and target data into a new feature space with two properties: (i) the distributions of the source and the target data sets are as close as possible in the new feature space, and (ii) the important structural information of the data is preserved. The embedded data can be in lower dimensional space while preserving the aforementioned properties and therefore the method can be considered as a dimensionality reduction method as well. Our proposed method has a closed-form solution and the experimental results show that it works well in practice.

Keywords domain adaptation; kernel embedding; Hilbert-Schmidt independence criterion

Corresponding Author: fdorri@cs.ubc.ca

1 Introduction

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

In conventional predictive models, the statement that the source data is a suitable representative of the target data is reflected in the assumption that the underlying distributions of the source and target data sets are identical. But this assumption is not always valid. Different reasons may cause the underlying probability distributions of the source and target data sets to be different. The reason might be in the difficulty or uncontrollability in gathering data. For example, having samples in the source data set from the target data set distribution might not be possible as it costs a lot or because of its unavailability at the moment.

In last decades, attention has been focused on domain adaptation problem in machine learning [1, 2]. The developed methods are now widely used in diverse fields [2, 3]. For instance, in 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 target data set population is not necessarily from the same age group.

Domain adaptation has been studied under different names [4] e.g. covariate shift [5], class imbalance [6], semi-supervised learning [7, 8, 9], transfer learning [10, 11] multi-task learning [12, 13, 14] and sample selection bias [15, 16], but all these methods mostly tackle the problem by two approaches: re-weighting source instances or changing the representation space [4, 17].

In re-weighting source instances, to solve the issue of the difference between the probability distributions of the source and target data sets; weights, $w_s(\boldsymbol{x}, \boldsymbol{y})$, are assigned to the pre-defined loss function. The modified loss function over the source data set is then minimized to learn the predictive model [2, 18]. In changing the representation approach, the data is embedded into a new feature space. Regardless of the dimensionality of the new representation of data which might be in a higher [19] or lower dimensional space [20, 21], the probability distributions of the source data and that of the target data in the new feature space are

more similar.

There are common drawbacks among proposed methods in literature: (i) approximation of the underlying distributions makes solving the problem hard in high dimensional data sets, (ii) exploring a new representation of the data which is not necessarily linear, usually makes solving the problem computationally expensive, and (iii) some domain adaptation techniques are applicable only to restricted predictive models.

We propose a method that overcomes the above drawbacks based on kernel distribution embedding and Hilbert-Schmidt Independence Criterion. The proposed algorithm finds a new representation of the data in a new feature space such that the underlying probability distributions of the embedded source and target 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 as a dimensionality reduction technique as well.

1.1 Notation

Let $\mathcal{X}$ and Φ denote random variables (i.e. input variables in the original feature space and the new feature space respectively). $\mathcal{Y}$ denotes 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 source and target data sets are different. $P_s(\mathcal{X}, \mathcal{Y})$ and $P_t(\mathcal{X}, \mathcal{Y})$ denote the underlying true joint probability distributions of the source and target data sets respectively. $P_s(\mathcal{X})$ and $P_t(\mathcal{X})$ denote the true marginal probability distributions of $\mathcal{X}$ and $\mathcal{Y}$ in the source and target data sets. Similarly, $P_s(\mathcal{Y}|\mathcal{X})$ and $P_t(\mathcal{Y}|\mathcal{X})$ are used to show the true conditional probability distributions in the two domains.

The bold lower case alphabet, $\boldsymbol{x}$, is a d-dimensional sample. $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$ denote the matrices of the source and target data set samples of size $d \times n_s$ and $d \times n_t$ respectively. n_s and n_t are the numbers of the samples in the source and target data sets respectively. $\boldsymbol{X}_{d\times n} = [\boldsymbol{X}_s \boldsymbol{X}_t]$ is a matrix of n, $d-$dimensional samples where $n = n_s + n_t$. $\boldsymbol{\Phi}$ is the new representation of the data in the new feature space, where $\boldsymbol{\Phi}$ is defined to be

$$\Psi : \boldsymbol{X} \rightarrow \boldsymbol{\Phi}, \tag{1}$$

such that

$$\mathbf{\Phi} := [\mathbf{\Phi}_s \mathbf{\Phi}_t]. \quad (2)$$

$\mathbf{\Phi}_s$ and $\mathbf{\Phi}_t$ denote the embedded source and target data sets in the new representation space.

2 Method

The main challenge of domain adaptation problem is that of the non-similarity of the probability distributions of the source and target data sets, i.e. $P_s(\mathcal{X}, \mathcal{Y})$ and $P_t(\mathcal{X}, \mathcal{Y})$. Decomposing the joint probability distributions of the source and target data sets as

$$p_s(\mathcal{X}, \mathcal{Y}) = p_s(\mathcal{X}) p_s(\mathcal{Y}|\mathcal{X})$$
$$p_t(\mathcal{X}, \mathcal{Y}) = p_t(\mathcal{X}) p_t(\mathcal{Y}|\mathcal{X}),$$

It is assumed that all the difference between the joint probability distributions of the source and target data sets is due to the difference between their marginal probability distributions and there exists a new representation of the data, $\mathbf{\Phi}$, such that the marginal probability distributions of embedded source and target data sets are similar:

$$P_s(\Phi) \approx P_t(\Phi). \quad (3)$$

2.1 Minimizing the Distance Between Two Probability Distributions

The crucial criterion for solving domain adaptation problem is to make the distance between probability distributions of the source and the target data sets as small as possible. Maximum Mean Discrepancy (MMD) is a non parametric measure of the distance between distributions of data sets. It is a metric representative of the distance between the means of those probability distributions as follows [22]

$$MMD(\hat{P}_s, \hat{P}_t) = \|\mu_{\boldsymbol{X}_s}[\hat{P}_s] - \mu_{\boldsymbol{X}_t}[\hat{P}_t]\|_{\mathcal{H}} = \quad (4)$$
$$\sup_{g \in \mathcal{F}, \|g\|_{\mathcal{H}} \leq 1} (\mathbf{E}_{\boldsymbol{X}_s \sim \hat{P}_s} g(\boldsymbol{x}_s) - \mathbf{E}_{\boldsymbol{X}_t \sim \hat{P}_t} g(\boldsymbol{x}_t)),$$

where $\mathbf{E}_{\boldsymbol{x} \sim P}[g(\boldsymbol{x})]$ is the expectation value of the function $g(\boldsymbol{x})$ (where the samples are drawn from probability distribution P) . It has been also shown by Jegelka et al. [22] that MMD can be estimated empirically as

$$\|\mu_{\boldsymbol{X}_s}[\hat{P}_s] - \mu_{\boldsymbol{X}_t}[\hat{P}_t]\|^2_{\mathcal{H}} \approx tr(\boldsymbol{H} \boldsymbol{L}_{\boldsymbol{M}} \boldsymbol{H} \boldsymbol{L}_{\boldsymbol{\Phi}}),$$

where $\boldsymbol{H}$ is the centering matrix that is defined to be $\boldsymbol{H} := \boldsymbol{I} - n^{-1} \boldsymbol{e} \boldsymbol{e}^T$, and $\boldsymbol{e}$ is a vector of ones. $\boldsymbol{L}_{\boldsymbol{\Phi}}$ is a kernel over $\mathbf{\Phi}$, let's say $\mathbf{\Phi}^T \mathbf{\Phi}$, and $\boldsymbol{L}_{\boldsymbol{M}}$ is a kernel in which the first cluster includes the source samples and the second cluster consists of the samples of the target data set:

$$\boldsymbol{L}_{\boldsymbol{M}} = \left[\begin{array}{c|c} \alpha \mathbf{1}^{n_s \times n_s} & \mathbf{0}^{n_s \times n_t} \\ \hline \mathbf{0}^{n_t \times n_s} & \beta \mathbf{1}^{n_t \times n_t} \end{array}\right] \quad (5)$$

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

$$\alpha = \frac{1}{n}\sqrt{\frac{n_s}{n}}, \quad \beta = \frac{1}{n}\sqrt{\frac{n_t}{n}}. \quad (6)$$

So the objective function is to

$$minimize \quad tr(\boldsymbol{H}\boldsymbol{L_M}\boldsymbol{H}\boldsymbol{L_\Phi}) = tr(\boldsymbol{\Phi}\boldsymbol{H}\boldsymbol{L_M}\boldsymbol{H}\boldsymbol{\Phi}^T).$$

A trivial solution of this is to collapse all the samples of each probability distribution to one point and then make those two points close to each other. But this new representation loses crucial information in data for the future predictive analysis. Therefore, this objective function by itself is not adequate and besides 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.

2.2 Preserving the Important Features of the Data

The dependency of the original data and its new representation can be used as a measure that shows how well the structure and important features for predicting the response variables are preserved. HSIC is considered as a measure for quantifying the dependency of two random variables.

Two random variables are independent iff the joint probability distribution of them is equal to the multiplication of their individual probability distributions:

$$P_{\mathcal{X},\Phi} = P_{\mathcal{X}} P_{\Phi} \tag{7}$$

So the dependency between two random variables can be measured based on the distance between the above probability distributions [23] and this metric is estimated empirically as

$$HSIC(\boldsymbol{X}, \boldsymbol{\Phi}) = (n-1)^{-2} tr(\boldsymbol{H}\boldsymbol{K_X}\boldsymbol{H}\boldsymbol{L_\Phi}), \tag{8}$$

where $\boldsymbol{L_\Phi}$ is a kernel over $\boldsymbol{\Phi}$, let's say $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$, and $\boldsymbol{K_X}$ is a valid kernel on the original data. The choice of the kernel implies the structure and important information that is desired to be preserved.

So a supplementary objective function is

$$maximize \quad tr(\boldsymbol{H}\boldsymbol{K_X}\boldsymbol{H}\boldsymbol{L_\Phi}) = tr(\boldsymbol{\Phi}\boldsymbol{H}\boldsymbol{L_M}\boldsymbol{H}\boldsymbol{\Phi}^T).$$

2.3 Adapting Component Analysis

Here, we first propose, an unsupervised domain adaptation algorithm. We will call it UnSupervised Adapting Component Analysis (US-ACA). The algorithm exploits only the source and target data sets to find a new representation of the data, $\boldsymbol{\Phi}$, which makes the probability distributions of the source and target 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 source data set are also exploited to strengthen the US-ACA and this will be called Semi-Supervised Adapting Component Analysis (SS-ACA).

2.3.1 Unsupervised Adapting Component Analysis

Minimizing the distance between $P(\mathbf{\Phi}_s)$ and $P(\mathbf{\Phi}_t)$ and preserving the important features of $\boldsymbol{X}$, are incorporated to establish an unsupervised single optimization problem for solving domain adaptation problem in which its solution is an embedding of the data into a new feature space. The objective function of the proposed algorithm is defined as

$$\textit{maximize} \quad \frac{tr(\boldsymbol{H}\boldsymbol{K}_{\boldsymbol{X}}\boldsymbol{H}\boldsymbol{L}_{\mathbf{\Phi}})}{tr(\boldsymbol{H}\boldsymbol{L}_{\boldsymbol{M}}\boldsymbol{H}\boldsymbol{L}_{\mathbf{\Phi}})}, \tag{9}$$

where the denominator is the measure for the distance between the probability distributions of the source data set and that of the target data set. Minimizing this measure or equivalently, maximizing the inverse of it, makes this distance as small as possible. 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 and important information of the data depending on kernel, $\boldsymbol{K}_{\boldsymbol{X}}$. 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. For example one can choose Isomap kernel:

$$\boldsymbol{K} = \frac{1}{2}\boldsymbol{H}\boldsymbol{D}^2\boldsymbol{H}, \tag{10}$$

where $\boldsymbol{D}$ is the geodesic distance matrix and $\boldsymbol{H}$ is the centring matrix. Choosing Isomap kernel will keep the geodesic distance of the data while a linear kernel preserve the Euclidean distance of the data. Rewriting the optimization problem in terms of $\mathbf{\Phi}$ we have

$$\textit{maximize} \quad \frac{tr(\boldsymbol{H}\boldsymbol{K}_{\boldsymbol{X}}\boldsymbol{H}\mathbf{\Phi}^T\mathbf{\Phi})}{tr(\boldsymbol{H}\boldsymbol{L}_{\boldsymbol{M}}\boldsymbol{H}\mathbf{\Phi}^T\mathbf{\Phi})} = \frac{tr(\mathbf{\Phi}\boldsymbol{H}\boldsymbol{K}_{\boldsymbol{X}}\boldsymbol{H}\mathbf{\Phi}^T)}{tr(\mathbf{\Phi}\boldsymbol{H}\boldsymbol{L}_{\boldsymbol{M}}\boldsymbol{H}\mathbf{\Phi}^T)}. \tag{11}$$

The objective function in (11) is invariant with respect to any scaling of $\mathbf{\Phi}$, so $\mathbf{\Phi}$ can be chosen such that the denominator $tr(\mathbf{\Phi}\boldsymbol{H}\boldsymbol{L}_{\boldsymbol{M}}\boldsymbol{H}\mathbf{\Phi}^T)$ is equal to one:

$$\begin{aligned} &\textit{maximize} \quad tr(\mathbf{\Phi}\boldsymbol{H}\boldsymbol{K}_{\boldsymbol{X}}\boldsymbol{H}\mathbf{\Phi}^T) \\ &\textit{subject to} \quad tr(\mathbf{\Phi}\boldsymbol{H}\boldsymbol{L}_{\boldsymbol{M}}\boldsymbol{H}\mathbf{\Phi}^T) = 1. \end{aligned} \tag{12}$$

This optimization problem is an instance of Rayleigh quotient and finding the optimal $\mathbf{\Phi}$ is straight forward as it has a closed-form solution. This corresponds to an eigenvector estimation problem with $\mathbf{\Phi}^T$ as a matrix of eigenvectors of $\boldsymbol{K}_{\boldsymbol{X}}^{-1}\boldsymbol{L}_{\boldsymbol{M}}$. 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, $\mathbf{\Phi}$ represents the data $\boldsymbol{X}$ 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, $[\mathbf{U}\ \mathbf{V}] := eigs(\mathbf{S}, dim, 'LR')$ estimates the first $'dim'$

Algorithm 1 Unsupervised ACA

1: $\boldsymbol{X}^{d\times n} \leftarrow [\boldsymbol{X}_s^{d\times n_s}\ \boldsymbol{X}_t^{d\times n_t}]$
2: $\boldsymbol{K_X} \leftarrow$ *A kernel on* $\boldsymbol{X}$. *e.g. Linear kernel*
3: $\boldsymbol{L_M} \leftarrow$ *The MMD kernel on two clusters of the source and target data sets*
4: $\boldsymbol{H} \leftarrow \boldsymbol{I} - n^{-1}\boldsymbol{e}\boldsymbol{e}^T$
5: $dim \leftarrow$ *The desired output dimensionality*
6: $\boldsymbol{S} := (\boldsymbol{H}\boldsymbol{K_X}\boldsymbol{H})^{-1}(\boldsymbol{H}\boldsymbol{L_M}\boldsymbol{H})$
7: $[\boldsymbol{U}\ \ \boldsymbol{V}] := eigs(\boldsymbol{S}, dim, \text{'LR'})$
8: $\boldsymbol{\Phi}_s := \boldsymbol{U}(1 : n_s, :)$.
9: $\boldsymbol{\Phi}_t := \boldsymbol{U}(n_s + 1 : end,\ :)$
10: $\boldsymbol{\Phi} = [\boldsymbol{\Phi}_s\ \boldsymbol{\Phi}_t]$
11: **return** $\boldsymbol{\Phi}$

large eigenvalues and eigenvectors of $\mathbf{S}$.

2.3.2 Semi-supervised Adapting Component Analysis

Exploiting the response variables of the source 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 available labels or response variables of the source 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. The proposed domain adaptation algorithm called Semi-Supervised Adapting Component Analysis exploits not only the source and target data sets to find a new representation of the data, $\boldsymbol{\Phi}$, but also the response variables of the source data set are exploited to strengthen performance of the algorithm. These are encapsulated in kernel $\boldsymbol{K_X}$. This algorithm finds an appropriate representation of the data without adding extra complexity to US-ACA algorithm. Once the appropriate representation is found, we can apply further predictive algorithms to the samples in the new feature space where domain adaptation problem has been solved.

Choice of kernel $\boldsymbol{K_X}$ for classification task Exploiting the response variables of the source data set (which could sometimes be easily accessible) is valuable information that can improve the efficiency of the algorithm. Using information of the response variables can be advantageous in finding a new representation of data that is more appropriate for the following predictive analysis.

US-ACA finds a shared feature space where the distance between source and target data set probability distributions are reduced while the structural properties of the data set are preserved. However, the new shared feature

space does not need to keep the whole structure of the data unchanged. It is only needed 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. So, it is wise to consider the type of the task for finding the important structure or hidden features of the data. Here, we focused on classification task and therefore, we would like to find a new representation which preserves structure and important features of the data relevant to the classification predictive model in a supervised manner.

Rewriting $\boldsymbol{K}_{\boldsymbol{X}}$ based on linear kernel, we have

$$\begin{aligned} \boldsymbol{K}_{\boldsymbol{X}} = \boldsymbol{X}^T\boldsymbol{X} &= \begin{bmatrix} \boldsymbol{X}_s^T \\ \\ \boldsymbol{X}_t^T \end{bmatrix} [\boldsymbol{X}_s \ \boldsymbol{X}_t] \\ &= \begin{bmatrix} \boldsymbol{X}_s^T\boldsymbol{X}_s & \boldsymbol{X}_s^T\boldsymbol{X}_t \\ \\ \boldsymbol{X}_t^T\boldsymbol{X}_s & \boldsymbol{X}_t^T\boldsymbol{X}_t \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_s} & \boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_t} \\ \\ \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s} & \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_t} \end{bmatrix} \end{aligned} \quad (13)$$

where $\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_s}$ and $\boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_t}$ involve the information of the structure of source and target data sets receptively. These two sub-matrices are important for learning or training of the predictive model (which is the main goal) and they should be preserved. But intuitively the relative structure of the source data set and the target data set need not to be necessarily fixed as domains are intended to get closer. So, this can help us modifying the structure of the data in a supervised manner with the known response variables of the source data set. So the matrix $\boldsymbol{K}_{\boldsymbol{X}}$ can be changed to $\hat{\boldsymbol{K}}_{\boldsymbol{X}}$ where $\hat{\boldsymbol{K}}_{\boldsymbol{X}}$ is constructed based on the data and the known response variables. For example, $\boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s}$ is initially representing the similarity of the source data set and the target data set samples. But in a classification task, if two samples of the source data set are similar (they are in a same class), based on their response variables, then they should not be different from a target data set sample perspective. So the sub-matrices of $\boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s}$ and $\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_t}$ can be smoothed by a process which reduces the variation of the data in unrelated dimensions while it keeps the variation of the data along the directions which contain important information relevant to predicting the response variables. Therefore, those two sub-matrices, $\boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s}$ and $\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_t}$ are substituted with the following matrices:

$$\hat{\boldsymbol{K}}_{\boldsymbol{X}_s\boldsymbol{X}_t} = \boldsymbol{K}_{\boldsymbol{Y}_s}\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_t} \quad (14)$$

$$\hat{\boldsymbol{K}}_{\boldsymbol{X}_t\boldsymbol{X}_s} = \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s}\boldsymbol{K}_{\boldsymbol{Y}_s}, \quad (15)$$

where $\boldsymbol{K}_{\boldsymbol{Y}_s}$ is a kernel on the response variables of the source data set $\boldsymbol{X}_s$ which represents the

similarity between the labels of the source data set samples and its main role is to even out the difference between similar samples. Based on this formulation, the sample of the source data set, $\boldsymbol{x}_i$ is changed to the weighted mean of its similar samples. The weight is proportional to the similarity of sample $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ (that is the (i,j)th entry of the kernel $\boldsymbol{K}_{\boldsymbol{Y}_s}$). This makes the variation of similar samples smaller. Therefore, $\boldsymbol{K}_{\boldsymbol{X}}$ is changed to $\hat{\boldsymbol{K}}_{\boldsymbol{X}}$:

$$\hat{\boldsymbol{K}}_{\boldsymbol{X}} = \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_s} & \hat{\boldsymbol{K}}_{\boldsymbol{X}_s\boldsymbol{X}_t} \\ \\ \hat{\boldsymbol{K}}_{\boldsymbol{X}_t\boldsymbol{X}_s} & \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_t} \end{bmatrix}. \qquad (16)$$

The steps of the algorithm is summarized in Algorithm 1. Matlab notations are used for simplicity.

Algorithm 2 Semi-supervised ACA algorithm

$\boldsymbol{X}^{d\times n} \leftarrow [\boldsymbol{X}_s^{d\times n_s}\ \boldsymbol{X}_t^{d\times n_t}]$
$\boldsymbol{L_M} \leftarrow$ *The MMD kernel on two clusters of the source and target data sets*
$\hat{\boldsymbol{K}}_{\boldsymbol{X}_t\boldsymbol{X}_s} \leftarrow \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s}(\boldsymbol{K}_{\boldsymbol{Y}_s})$
$\hat{\boldsymbol{K}}_{\boldsymbol{X}_s\boldsymbol{X}_t} \leftarrow (\boldsymbol{K}_{\boldsymbol{Y}_s})\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_t}$
$\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_s} \leftarrow$ *A kernel on* $\boldsymbol{X}_s$. *e.g. Linear*
$\boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_t} \leftarrow$ *A kernel on* $\boldsymbol{X}_t$. *e.g. Linear*
$\hat{\boldsymbol{K}}_{\boldsymbol{X}} \leftarrow \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_s} & \hat{\boldsymbol{K}}_{\boldsymbol{X}_s\boldsymbol{X}_t} \\ \\ \hat{\boldsymbol{K}}_{\boldsymbol{X}_t\boldsymbol{X}_s} & \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_t} \end{bmatrix}$
$\boldsymbol{S} := (\boldsymbol{H}\hat{\boldsymbol{K}}_{\boldsymbol{X}}\boldsymbol{H})^{-1}(\boldsymbol{H}\boldsymbol{L}_{\boldsymbol{M}}\boldsymbol{H})$
$[\boldsymbol{U}\ \ \boldsymbol{V}] := eigs(\boldsymbol{S}, dim, \text{'LR'})$
$\boldsymbol{\Phi}_s := \boldsymbol{U}(1:n_s,:)$.
$\boldsymbol{\Phi}_t := \boldsymbol{U}(n_s+1:end,\ :)$
$\boldsymbol{\Phi} = [\boldsymbol{\Phi}_s\ \boldsymbol{\Phi}_t]$
return $\boldsymbol{\Phi}$

3 Experimental Results

Domain adaptation problem has been studied extensively in past decades and several methods have been developed. In this paper, the proposed method is compared with MMDE [24] and CODA [25]. MMDE is chosen since a similar measure, MMD, is considered for estimating the distance between the probability distributions of the source and target sets. CODA is chosen as a recently developed method for solving domain adaptation problem.

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 probability distribution and that of the target domain 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 Semi Definite 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 target domain [24, 26].

CODA learns a target predictor by maintaining and growing the source domain (i.e. iteratively adding target 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 [25].

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 in (11), $\boldsymbol{K_X}$ and $\boldsymbol{L_M}$ are assumed to be known. Then the kernel $\boldsymbol{K_X}$ has been changed to $\hat{\boldsymbol{K}}_{\boldsymbol{X}}$ in SS-ACA. $\hat{\boldsymbol{K}}_{\boldsymbol{X}} = \begin{bmatrix} \boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_s} & \hat{\boldsymbol{K}}_{\boldsymbol{X}_s\boldsymbol{X}_t} \\ \hat{\boldsymbol{K}}_{\boldsymbol{X}_t\boldsymbol{X}_s} & \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_t} \end{bmatrix}$ where $\hat{\boldsymbol{K}}_{\boldsymbol{X}_t\boldsymbol{X}_s} = \boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s}(\boldsymbol{K}_{\boldsymbol{Y}_s})$ and $\hat{\boldsymbol{K}}_{\boldsymbol{X}_s\boldsymbol{X}_t} = (\boldsymbol{K}_{\boldsymbol{Y}_s})\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_t}$. $\boldsymbol{K}_{\boldsymbol{Y}_s}$ represents the similarity of the response variables of the source data set and can be constructed in various forms. Without loss of generality, we use delta kernel for $\boldsymbol{Y}$:

$$\boldsymbol{L}_{ij} = \begin{cases} \frac{1}{n_{\boldsymbol{y}}} & \boldsymbol{y}_i = \boldsymbol{y}_j \\ 0 & \text{otherwise.} \end{cases}, \tag{17}$$

where $n_{\boldsymbol{y}}$ is the number of samples in the class of $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$.

Multiplication of the kernel, $\boldsymbol{K}_{\boldsymbol{Y}_s}$, with $\boldsymbol{K}_{\boldsymbol{X}_t\boldsymbol{X}_s}$ and $\boldsymbol{K}_{\boldsymbol{X}_s\boldsymbol{X}_t}$ is basically substituting each sample of the source data set by the weighted mean of its corresponding similar samples which makes the variation of the data along similar samples smaller to even out the

difference between them.

3.2 Toy Classification Example

We first test our proposed algorithms, US-ACA and SS-ACA on a toy example. The source data set consists of 100 samples of the two dimensional data drawn from a multivariate normal distribution in which the mean is $\mu_s = (-1, 3)$ and the covariance matrix is $\sigma_s = \begin{bmatrix} 2 & 0.5 \\ 0.5 & 2 \end{bmatrix}$. The source 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 target data set consists of 200 samples from another random multivariate normal distribution with different mean, $\mu_t = (2, 1)$ and similar covariance matrix. The target data set is also categorized in two classes based on their 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 source and target data set distributions along the first feature space.

Fig. 1-a demonstrates the data in the original feature space and Fig. 1-b and 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 source and target data set distributions is reduced. Consequently, the new source data samples are better representatives of the target data set. Applying the algorithms and getting the embedded data, we can classify them using SVM [1] or any other classifier. As shown in the Table 1, SVM can classify the data with 41%

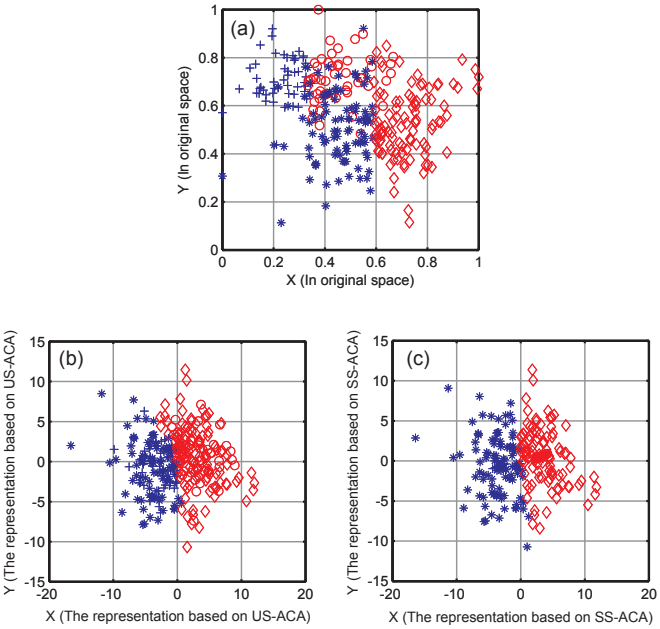

Fig. 1. (a) 2-dimensional data in the original space. Circles, $\circ$, and crosses, $\times$ are two classes of the source data set and diamonds,$\diamond$, and stars, $*$, are two classes of the target data set. (b) and (c) are the new representation of the data in the new feature space based on US-ACA and SS-ACA respectively.

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

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.

Table 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%

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 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 2, US-ACA and SS-ACA provide significant improvement in the error rate of the classification process.

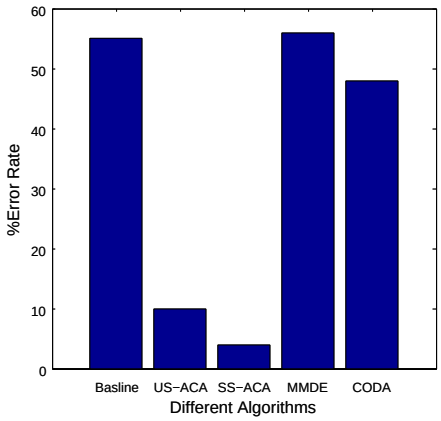

Fig. 2. The error rate comparison for different algorithms and the baseline on the toy example

3.3 Real World Data Sets

In this section, we test the proposed methods, US-ACA and SS-ACA, on different real world data sets of images, text and biological data bases.

The First data set which is a collection of images, is MNIST handwritten digits [2]. 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 target 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 algorithm using MNIST data set, we generate different data sets called Digits(1) to Digits(7) which are shown in Table 2.

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

Table 2. Different data set generated from MNIST database

NAME	SOURCE	TARGET
DIGITS(1)	$0, 1$	$3, 4$
DIGITS(2)	$5, 7$	$2, 9$
DIGITS(3)	$3, 4$	$1, 6$
DIGITS(4)	$2, 8$	$3, 9$
DIGITS(5)	$6, 9$	$5, 7$
DIGITS(6)	$1, 3$	$3, 6$
DIGITS(7)	$8, 4$	$3, 2$

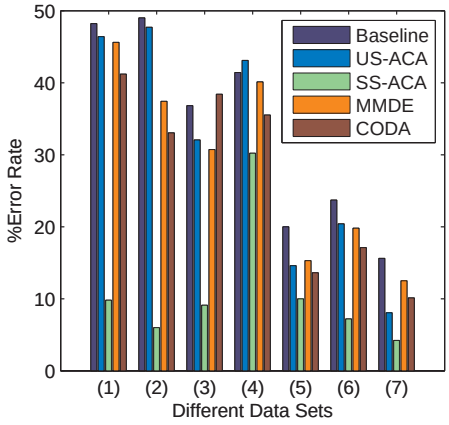

Fig. 3. The error rate comparison for different algorithms and the baseline on the data sets generated from MNIST data base. (1) to (7) on the X-axis stands for Digits(1) to Digits(7) respectively.

The source digits in each data set are showing the two digits that the algorithm is learned to classify them, while the goal is to classify the digits of the source data set. These data sets are randomly chosen among all possible cases. The number of the source and target samples are 300 and 500 respectively. The size of the source and target data sets are fixed for all of the data sets in Table 2. We have compared the error rate on the data sets defined in Table 2 for different algorithms and the baseline in Figure 3.

The dimensionality of the output data in US-ACA and SS-ACA is set to 2. The new representation of the data of Digits(1) in 2-dimensional space based on US-ACA and SS-ACA are depicted in Figure 4-a and 4-b respectively. It is clear that the new representation, particularly in SS-ACA is significantly better for classification. 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 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 of the response variables of the source data set. There-

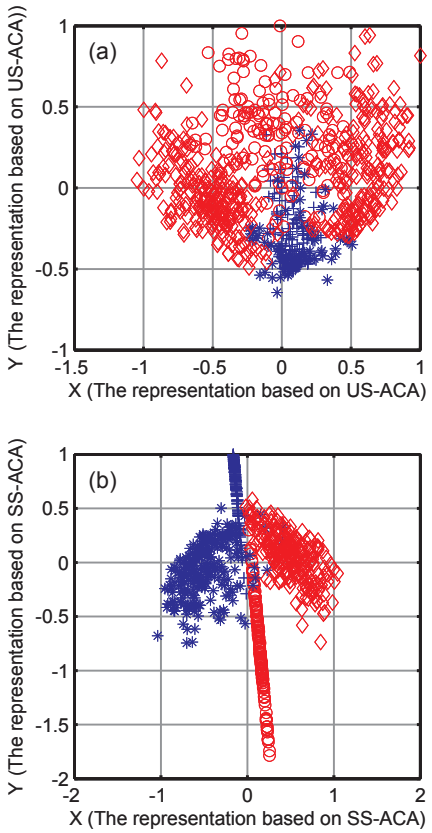

Fig. 4. The 2-dimensional representation of Digits(1) data set based on US-ACA and SS-ACA in (a) and (b) respectively. Circles, $\circ$, and crosses, $\times$ are two classes of the source data set. Diamonds,$\diamond$, and stars, $*$, are two classes of the target data set based on labels.

fore, it is not expected to have a performance as good as the algorithms which are taking advantage of the source 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 different 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 postings [3].

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 source data set, and 2000 randomly selected postings from groups 3 and 4 as the target 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 source data sets and $2, 4$ and $2, 3$ in their target data sets respectively. In each of the artificially generated data set, the task is to classify the postings of the target data set while the algorithm is learned based on the source data set.

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

We compare our proposed methods, US-ACA, SS-ACA with MMDE and CODA on the Newsgroup1, Newsgroup2 and Newsgroup3 data sets. Their corresponding error rates are compared with the baseline in Fig. 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 Fig. 5, the error rate has been decreased from almost 50% to approximately $35\% - 40\%$ for US-ACA and $25\% - 30\%$ for 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 that are characterized by missing features, and this is often the case in natural language processing while our algorithm is not developed for a specific type of data.

To test the performance of the proposed algorithm on different types of data sets, we run a set of classification experiments on several UCI data sets [4] in which they are biased artificially. To make an artificial biased data set, the data is randomly divided into the source and target data sets. Then an additional variable, s_i, for each sample of source data set is defined [16]. s_i is set to depend only on one of the sample features, therefore, the biasing procedure is called, simple bias [27]. This additional variable determines whether the corresponding sample is contributing in the biased source data set or not. It means if $s_i = 1$, then the ith sample is included in the biased source data set, else 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 source data set.

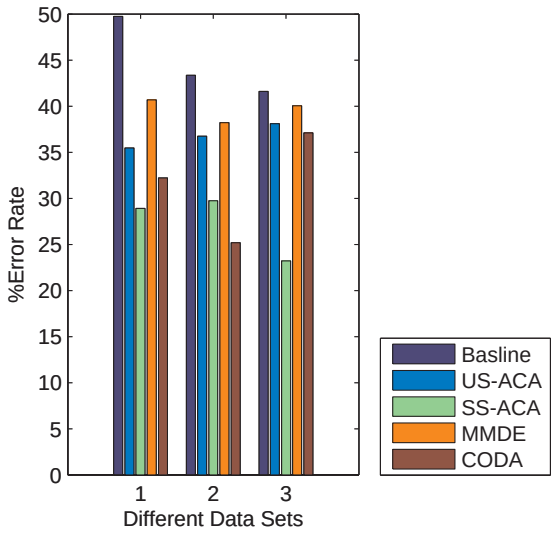

Fig. 5. The error rate comparison for different algorithms in three data sets. 1, 2 and 3 on the X-axis stands for Newsgroup1, Newsgroup2 and Newsgroup3 data sets respectively.

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

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 source data set.

The Breast Cancer dataset from the UCI archive 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 Fig. 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 Fig. 6. As can be seen, SS-ACA has better performance compared with 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

$$NI = \frac{|Error_A - Error_B|}{Error_A}. \qquad (18)$$

On the right column of Fig. 6 the Normalized Improvement of SS-ACA with respect to the baseline is shown. As can be seen after adapting the domains of source and target data sets, the performance is improved approximately up

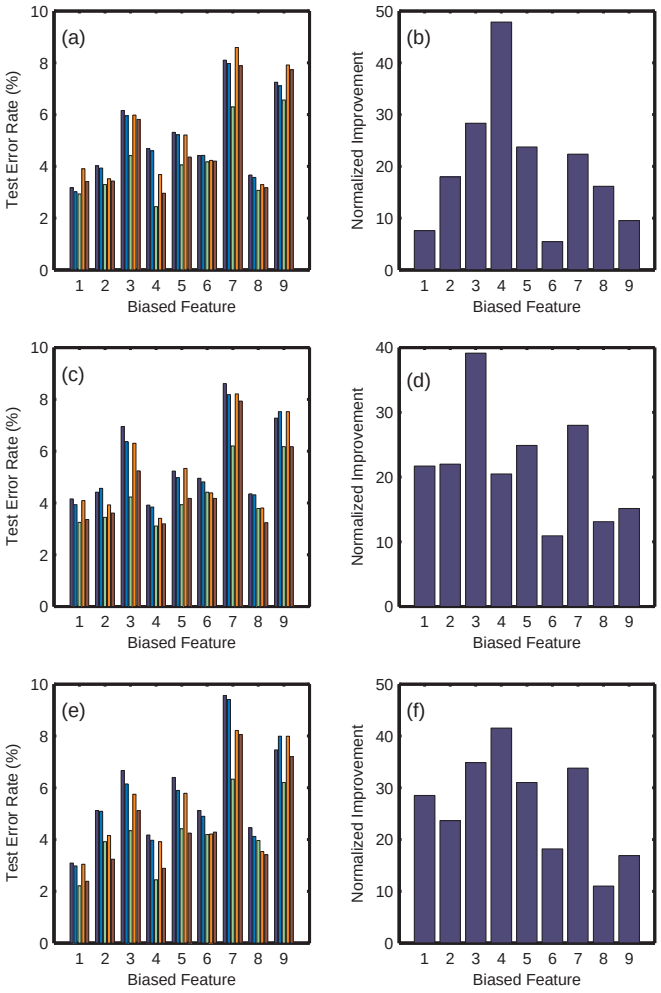

Fig. 6. (a), (c) and (e) are the error rate comparison in different algorithms on Breast Cancer data set with *Biasing Ratio* of 70%, 80% and 90% respectively. The X-axis is showing the features which the biasing process is based on. (b), (d) and (f) are the Normalized Improvement of SS-ACA with respect to the baselines of (a), (c) and (e) respectively. The bars from left to right correspond to Baseline, US-ACA, SS-ACA, MMDE and CODA.

Table 3. Test result on various data sets by different methods. 1-nearest neighbour has been used for classification.

Data set	(n_s, n_t)	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%

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 where their biasing process is as explained above, and their biasing ratio is 80%. The number of source and target data set samples, the biased feature and also the number of classes in each data set is depicted in Table 3. The error rate of different methods on these data sets are also available on Table 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 Fig. 7(a). 784 is the dimensionality of the data in original space. The error rate is minimum in low dimensional space.

The changes of the error rate along different dimensions of the Newsgroup2 is also demonstrated in Fig. 7(b). 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 SS-ACA can be considered as a dimension reduction technique as well. The appropriate dimension in each data set can be calculated by cross validation in practice.

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. The proposed methods has a closed-form solution and it is

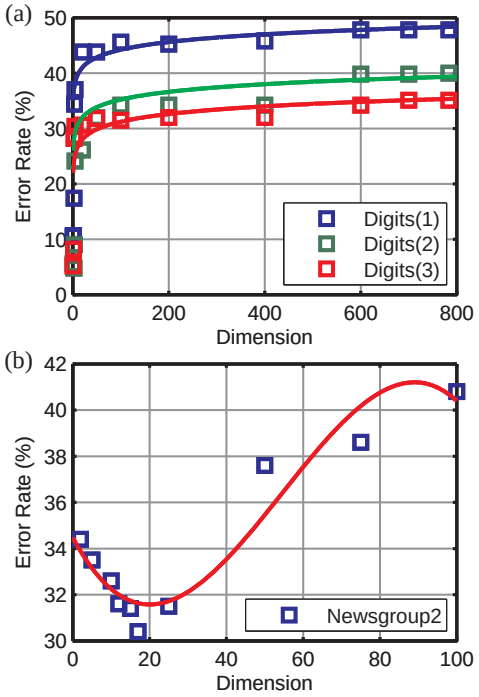

Fig. 7. (a) The error rate changes of SS-ACA vs. different dimensions on Digits(1), Digits(2) and Digits(3) data sets. (b) The error rate changes of SS-ACA vs. different dimensions on Newsgroup2 data set.

faster than MMDE and CODA.

4 Conclusion and future work

We have presented a domain adaptation algorithm in which the data samples are transferred to a new feature space. The new representation of the data is explored such that the source and the target 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 a fast 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 valuable information of the response variables of the source 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 the 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 the developed optimization problem 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, Vaughan J W. A theory of learning from different domains. *Machine Learning Journal*, 2010, 79(1-2): 151–175.

[2] Huang J, Smola A J, Gretton A, Borgwardt K M, Scholkopf B. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing System*, MIT Press, Cambridge, MA, 2007, 19:601–608.

[3] Liu Q, Mackey A, Roos D, Pereira F. Evigan: A hidden variable model for integrating gene evidence for Eukaryotic gene prediction. *Bioinformatics*, 2008, 24(5):597–605.

[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*, 2000, 90(2): 227–244.

[6] Japkowicz N, Stephen S. The classic imbalance problem: a systematic study. *Intelligent Data Analysis*, 2002, 6(5):429–450.

[7] Chapelle O, Scholkopf B, Zien A, editors. Semi-supervised learning. *MIT Press*, 2006.

[8] Dai W, Xue G, Yang Q, Yu Q. Transferring naive Bayes classifier for text classification. In *Proc. the 22nd AAAI Conference on Artificial Intelligence*, Jul. 2007, pp. 540–545.

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

[10] Wang B, Tang J, Fan W, Chen S, Tan C, Yang Z. Query-dependent cross-domain ranking in heterogeneous network. *Knowledge and Information Systems*, 2013, 34 (1):109–145.

[11] Shao H, Tong B, Suzuki E. Extended MDL principle for feature-based inductive transfer learning. *Knowledge and Information Systems*, 2013, 35(2):365–389.

[12] Ben-David S, Schuller R. Exploiting task relatedness for multiple task learning. In

Proc. of the 16th Annual Conference on Learning Theory, Aug. 2003, pp. 567–580.

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

[14] Micchelli C A, Pontil M. Kernels for multi-task learning. In *Advances in Neural Information Processing Systems 17*, MIT Press, 2005, pp. 921–928.

[15] Heckman J J. Sample selection bias as a specification error. *Econometrica*, 1979, 47 (1):153–161.

[16] Zadrozny B. Learning and evaluating classifiers under sample selection bias. In *Proc. the 21th Annual International Conference on Machine Learning*, Jul. 2004, pp. 114–121.

[17] Morvant E, Habrard A, Ayache S. Parsimonious unsupervised and semi-supervised domain adaptation with good similarity functions. *Knowledge and Information Systems*, 2012, 33(2):309–349.

[18] Chan Y S, Ng H T. Estimating class priors in domain adaptation for word sense disambiguation. In *Proc. the 21th International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Jul. 2006, pp. 89–96.

[19] Daume III H, Kumar A and Saha A. Co-regularization based semi-supervised domain adaptation. In *Proc. the Conference on Neural Information Processing Systems*, Dec. 2010, pp. 1–9.

[20] Xue G, Dai W, Yang Q, Yu Y. Topic-bridged PLSA for cross-domain text classification. In *Proc. the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, Jul. 2008, pp. 627–634.

[21] Blitzer J, McDonald R, Pereira F. Domain adaptation with structural correspondence learning. In *Proc. the 2006 Conference on Empirical Methods in Natural Language Processing*, Jul. 2006, pp. 120–128.

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

[23] Gretton A, Bousquet O, Smola A J, Scholkopf B. Measuring statistical dependence between Hilbert-Schmidt norms. In

Proc. Algorithmic Learning Theory, Oct. 2005, 227:63–77.

[24] Pan S J, Kwok J T, Yang Q. Transfer learning via dimensionality reduction. In *Proc. AAAI*, Jul. 2008, pp. 677–682.

[25] Chen C, Weinberger K Q, Blitzer J C. Co-training for domain adaptation. In *Advances in Neural Information Processing Systems 24*, 2011, pp. 2456–2464.

[26] Pan S J, Tsang I W, Kwok J T, Yang Q. Domain adaptation via transfer component analysis. In *IEEE Transactions on Neural Networks*, 2011, 22(2):199-210.

[27] Gretton A, Smola A, Huang J, Schmittfull M, Borgwardt K, Schoelkopf B. Covariate shift by kernel mean matching. In *Dataset Shift in Machine Learning, Covariate Shift and Local Learning by Distribution Matching*, MIT Press, 2008, pp.131–160.

[28] Dorri F, Ghodsi A. Adapting component analysis. In *Proc. the 12th IEEE International Conference on Data Mining*, Dec. 2012, pp. 846-851.