Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 302815, 9 pages
http://dx.doi.org/10.1155/2015/302815

Hindawi

Research Article

Training Classifiers under Covariate Shift by Constructing the Maximum Consistent Distribution Subset

Xu Yu,[1] Miao Yu,[2] Li-xun Xu,[3] Jing Yang,[4] and Zhi-qiang Xie[5]

[1]*School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao 266061, China*
[2]*The College of Textiles and Fashion, Qingdao University, Qingdao 266071, China*
[3]*Sino-German Faculty, Qingdao University of Science and Technology, Qingdao 266061, China*
[4]*College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China*
[5]*College of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China*

Correspondence should be addressed to Xu Yu; yuxu0532@163.com

Received 23 July 2015; Revised 23 October 2015; Accepted 1 November 2015

Academic Editor: Eckhard Hitzer

The assumption that the training and testing samples are drawn from the same distribution is violated under covariate shift setting, and most algorithms for the covariate shift setting try to first estimate distributions and then reweight samples based on the distributions estimated. Due to the difficulty of estimating a correct distribution, previous methods can not get good classification performance. In this paper, we firstly present two types of covariate shift problems. Rather than estimating the distributions, we then desire an effective method to select a maximum subset following the target testing distribution based on feature space split from the auxiliary set or the target training set. Finally, we prove that our subset selection method can consistently deal with both scenarios of covariate shift. Experimental results demonstrate that training a classifier with the selected maximum subset exhibits good generalization ability and running efficiency over those of traditional methods under covariate shift setting.

1. Introduction

Traditional classification methods, such as Support Vector Machines (SVMs) [1, 2], decision tree [3, 4], and neural networks [5, 6], are always based on the assumption that the training and testing samples are drawn from the same distribution. In many classification scenarios, such as the class imbalance problem [7, 8], concept drift [9, 10], and covariate shift [11], however, this assumption is violated. An informal description on covariate shift is as follows. Covariate shift refers to the learning settings in which source data sets and target data sets have the same feature attributes, label attribute, and the conditional probabilities of $y \mid x$ but have different feature distributions. In this paper, we mainly study classifying under covariate shift. Below, we give two types of classification scenarios belonging to covariate shift. They are what we focus on in this paper.

Scenario 1. Classification problems contain training samples, testing samples, and auxiliary samples, where training samples and testing samples are drawn from the same distribution, while the auxiliary samples are drawn from another distribution. In addition, the training set size is very small.

In real world, lots of classification problems belong to Scenario 1. For example, suppose we want to construct a Web page classification model. The Web data used in training a Web page classification model can be easily outdated when applied to the Web sometime later, because the topics on the web change frequently. Often, new data are expensive to label and thus their quantities are limited due to cost issues. How to accurately classify the new test data by making the maximum use of the old data becomes a critical problem.

Scenario 2. Classification problems contain training samples and testing samples, where training samples and testing samples are drawn from different distributions. There are no auxiliary samples.

For example, suppose we are using a learning method to induce a model that predicts the side effects of a treatment for

a given patient. Because the treatment is not given randomly to individuals in the general population, the available training samples are not a random sample from the population. Therefore, the training samples and testing samples are drawn from different distributions. How to accurately classify the testing data by employing the training data becomes a critical problem.

In this paper, we address the two types of covariate shift problems by training on a newly constructed set following approximately the target distribution. The rest of this paper is organized as follows. In Section 2, we formally define covariate shift in machine learning terms and describe the related works on this problem. In Section 3, we propose the MIDS construction method by matching sample numbers between target set and auxiliary set in each feature subspace. In Section 4, we present the corresponding data correction methods with respect to the two scenarios, propose the corresponding classification algorithms, and analyze the time complexity of the algorithms. Our experimental results and discussion are shown in Section 5. Section 6 summaries the main contribution of this paper and gives some future works.

2. Related Concepts and Works

2.1. Related Concepts. In this section, we introduce some notations and definitions that are used in this paper. First of all, we give the definitions of a "domain" and a "task," respectively.

Definition 1 (domain). In this paper, a domain $\mathscr{D}$ consists of two components: a feature space $\mathscr{X}$ and a marginal probability distribution $P(X)$, where $X = \{x_1, \ldots, x_n\} \in \mathscr{X}$.

Definition 2 (task). Given a specific domain—$\mathscr{D} = \{\mathscr{X}, P(X)\}$—a task consists of two components: a label space $\mathscr{Y}$ and an objective predictive function $f(\cdot)$ (denoted by $\mathscr{T} = \{\mathscr{Y}, f(\cdot)\}$), which is not observed but can be learned from the training data, which consist of pairs $\{x_i, y_i\}$, where $x_i \in \mathscr{X}$ and $y_i \in \mathscr{Y}$. The function $f(\cdot)$ can be used to predict the corresponding label, $f(x)$, of a new instance x. From a probabilistic viewpoint, $f(x)$ can be written as $P(y \mid x)$.

In this section, we denote the source domain data by $D_S = \{(x_{S_1}, y_{S_1}), \ldots, (x_{S_{n_S}}, y_{S_{n_S}})\}$, where $x_{S_i} \in \mathscr{X}_S$ is the data instance and $y_{S_i} \in \mathscr{Y}_S$ is the corresponding class label. Similarly, we denote the target domain data by $D_T = \{(x_{T_1}, y_{T_1}), \ldots, (x_{T_{n_T}}, y_{T_{n_T}})\}$, where the input x_{T_i} is in $\mathscr{X}_T$ and $y_{T_i} \in \mathscr{Y}_T$ is the corresponding output. In most cases, $0 \leq n_T \ll n_S$.

We now give a formal definition of covariate shift.

Definition 3 (covariate shift). Covariate shift refers to the learning settings that have the following features: (1) source domain and target domain have the same feature and label spaces; that is, $\mathscr{X}_S = \mathscr{X}_T$ and $\mathscr{Y}_S = \mathscr{Y}_T$. (2) Source domain and target domain have different feature distribution; that is, $P_S(X) \neq P_T(X)$. (3) Source domain and target domain have the same concept; that is, $f_S(\cdot) = f_T(\cdot)$ or $P_S(Y \mid X) = P_T(Y \mid X)$.

It is worthwhile to note that there can be multiple auxiliary data sets in classification problems under covariate shift and their feature distributions can be different. In addition, from the definition of covariate shift, we can see that the two scenarios described in Section 1 do belong to covariate shift, because, for Scenario 2, testing samples can be considered as target samples and training samples can be considered as auxiliary samples.

2.2. Related Works. As described before, covariate shift includes the two scenarios described above. With respect to Scenario 1, auxiliary samples are utilized to improve the performance of classifiers. In previous works, Wu and Dietterich [12] proposed an image classification algorithm using both inadequate training data and plenty of low quality auxiliary data. They demonstrated some improvement by using the auxiliary data. However, they did not give a quantitative study using different auxiliary examples. Liao et al. [13] improved learning with auxiliary data using active learning. Rosenstein et al. [14] proposed a hierarchical naive Bayes approach for transfer learning using auxiliary data and discussed when transfer learning would improve or decrease the performance. Dai et al. [15] proposed a covariate shift-related algorithm, TrAdaBoost, which is an extension of the AdaBoost algorithm, to address the inductive transfer learning problems. TrAdaBoost assumes that the source and target domain data use exactly the same set of features and labels, but the distributions of the data in the two domains are different. In addition, TrAdaBoost assumes that, due to the difference in distributions between the source and the target domains, some of the source domain data may be useful in learning for the target domain but some of them may not and could even be harmful. It attempts to iteratively reweight the source domain data to reduce the effect of the "bad" source data while encouraging the "good" source data to contribute more to the target domain. For each round of iteration, TrAdaBoost trains the base classifier on the weighted source and target data. The error is only calculated on the target data. Furthermore, TrAdaBoost uses the same strategy as AdaBoost [16] to update the incorrectly classified examples in the target domain while using a different strategy from AdaBoost to update the incorrectly classified source examples in the source domain. However, TrAdaBoost can not deal with the case where there are multiple auxiliary data sets coming from different distributions.

With respect to Scenario 2, unlabeled testing samples are utilized to improve the performance of classifiers. Unlike semisupervising learning problem [17], for Scenario 2, the unlabeled testing samples are under a different distribution from the training samples and are used to correct the sample selection bias. In previous works, most approaches intend to estimate the importance $P(x_{S_i})/P(x_{T_i})$. If we can estimate the importance for each instance, we can solve the learning problems under covariate shift. There exist various ways to estimate $P(x_{S_i})/P(x_{T_i})$.

Zadrozny [18] proposed to estimate the terms $P(x_{S_i})$ and $P(x_{T_i})$ independently by constructing simple classification problems and then estimate the importance by taking the ratio of the estimated densities. However, estimating densities is known to be a hard problem particularly in high-dimensional cases. Therefore, this approach may not be effective.

Huang et al. [19] proposed a kernel-mean matching (KMM) algorithm to learn $P(x_{S_i})/P(x_{T_i})$ directly by matching the means between the source domain data and the target domain data in a reproducing kernel Hilbert space (RKHS). KMM is shown to work well if tuning parameters such as the kernel width are chosen appropriately. Thus, the importance estimation problem is now relocated to the model selection problem. Standard model selection methods such as cross-validation, however, are heavily biased under covariate shift. Therefore, KMM can not be directly applied in the cross-validation [20] framework.

Unlike KMM, Sugiyama et al. [21] proposed an algorithm known as Kullback-Leibler importance estimation procedure (KLIEP), which is equipped with a natural model selection procedure. KLIEP can be integrated with cross-validation to perform model selection automatically in two steps: (1) estimating the weights of the source domain data; (2) training models on the reweighted data.

In this paper, we propose a novel method by constructing a MIDS to deal with classification problems under covariate shift. The formal definition of MIDS and its construction method will be given in the next section. Unlike previous transfer learning methods, our method can consistently deal with both scenarios and the cases where there are multiple auxiliary data sets coming from different distributions. Furthermore, unlike the above sample reweighting techniques, we do not estimate distributions but match sample numbers between target set and auxiliary set in each feature subspace; we do not reweight samples but construct a new training set following approximately the target distribution.

3. The MIDS Construction Method

In this section, we use two data sets, target set and auxiliary set. Our objective is to design a method that can construct MIDS from auxiliary samples according to target distribution. First of all, we give the formal definitions of identical distribution subset (IDS) and MIDS.

Definition 4 (identical distribution subset, IDS). Let $T = \{t_1, t_2, \ldots, t_n\}$ be a target sample set and $S = \{s_1, s_2, \ldots, s_m\}$ a source sample set. Assuming that they follow different distributions and have the same feature and label space—that is, $P_S(X, Y) \neq P_T(X, Y)$—$\mathcal{X}_S = \mathcal{X}_T$ and $\mathcal{Y}_S = \mathcal{Y}_T$. Identical distribution subset is a subset of S and follows the same distribution with T.

Definition 5 (maximum identical distribution subset, MIDS). A proper identical distribution subset I is called a maximal identical distribution subset if there exists no other proper identical distribution subset J with a bigger size than that of I.

3.1. Basic Idea of MIDS Construction Method. Our basic idea of MIDS construction is first to partition the feature space into several subspaces and then construct MIDS by matching sample numbers between target set and auxiliary set in each feature subspace; that is, select a maximum amount of auxiliary samples from each subspace to compose the MIDS according to the proportion of target samples in each subspace. The detailed process of the MIDS construction method will be presented in Section 3.2.

3.2. The Detailed MIDS Construction Process. Let $X_T = \{x_1, x_2, \ldots, x_k\}$ be the n-dimensional target set drawn from the distribution $P_T(x)$ and $X_S = \{x'_1, x'_2, \ldots, x'_m\}$ the n-dimensional source set drawn from the distribution $P_S(x)$.

(1) Partitioning the Feature Space into Several Subspaces. Firstly, compute the mean of the target set by the following formula:

$$x_0 = \frac{\sum_1^k x_i}{k}. \tag{1}$$

Then, partition the n-dimensional space into 2^n subspaces. In detail, let $x = (x^1, \ldots, x^i, \ldots, x^n)$ be any vector in the feature space, where x^i denotes the ith-dimensional value of the vector x. Compare the ith-dimensional value of the vector x with the ith-dimensional value of x_0, and we can obtain n inequalities. We use an n-dimensional binary vector to represent these inequalities; that is to say, if $x^i \leq x_0^i$, we label the ith-dimensional value of the binary vector with 0, otherwise 1. Thus we can divide the feature space into 2^n subspaces, corresponding to 2^n binary vectors from $(0, 0, \ldots, 0)$ to $(1, 1, \ldots, 1)$, respectively. We number the subspace with the decimal numbers corresponding to the binary vectors.

(2) Computing the Proportion of the Target Samples in Each Subspace. Compute the number of the target samples in each subspace, and so we can obtain the proportion of samples in each subspace.

(3) Extracting Samples from the Auxiliary Set. We first compute the numbers of auxiliary samples in each subspace. Then, according to the proportion and the numbers, we select a maximum amount of samples from each subspace to compose the MIDS, noting that the proportion of the auxiliary samples selected from each subspace should be consistent with the proportion of the target samples in each subspace.

Thus we obtain the MIDS, and this subset can be considered to follow approximately the target distribution.

3.3. The Description of the MIDS Construction Algorithm and Its Time Complexity Analysis. The pseudocode of the MIDS construction algorithm is described in Algorithm 1. Firstly, we define two 2-dimensional arrays, A and B. The first dimension of Array A records the subspace number of samples in the target sample set. It is worth noting that Array A only has one record for samples with the same subspace number. The second dimension of Array A records the number of samples in the corresponding subspace. Array B is the same as Array A, but for the source sample set.

Require: the source sample set R_S, the target sample set R_T, dimension n, size of the source sample set k_S, size of the target sample set k_T
Ensure: the MIDS R
$x_0 = \frac{\sum_1^{k_T} x_i}{k_T}$;
Clarifying two 2-dimensional Arrays, A and B;
/*The first dimension of Array A records the subspace number of samples in the target sample set. Note that Array A only has one record for samples with the same subspace number. The second dimension of Array A records the number of samples in the corresponding subspace. Array B is the same as A, but for the source sample set*/
A = Array_generation(R_T, x_0) /*obtain Array A from R_T and x_0*/
B = Array_generation(R_S, x_0) /*obtain Array B from R_S and x_0*/
R = construct(a, b) /*select a maximum amount of samples from each subspace to compose the MIDS R according to Array A and Array B*/

Algorithm 1: The MIDS construction algorithm.

Then we obtain Array A from R_T and x_0. We compute the subspace number of samples by real number comparison operation. To obtain the number of samples in the corresponding subspace, we need to scan the whole data set R_T. It is the same with array B.

Finally, we select a maximum amount of samples from each subspace to compose the MIDS R according to Array A and Array B.

We split the whole space into 2^n subspaces, and for large n the number of subspaces is enormous, which would cause the curse of dimensionality if we compute the number of samples in each subspace. Luckily it is not necessary to do that, as the samples are always sparse in a high-dimensional space. Thus we only need to compute the number of samples in subspaces which consist of samples.

Therefore the time complexity is mainly composed of two parts, corresponding to calculating the proportion of the target samples in certain subspaces and calculating the numbers of the source samples in certain subspaces, respectively. It is worth noting that we only need to compute the number of samples in subspaces which consist of samples.

Thus if we define one-time real number comparison operation as one-time basic operation, we need to do $n(k_T + k_S)$ times operations, where k_S denotes the size of the source sample set and k_T denotes the size of the target sample set.

4. Classification Methods under Covariate Shift by Constructing the MIDS

In Section 3, we present a general MIDS construction method by matching sample numbers between target set and auxiliary set in each feature subspace. In this section, we will propose the special MIDS construction methods corresponding to Scenarios 1 and 2, respectively. Furthermore, we will propose the classification methods for the two scenarios.

4.1. The MIDS Construction of Scenario 1. Let $T_{\text{tr}} = \{(x_i^{\text{tr}}, y_i^{\text{tr}}) \mid i = 1, 2, \ldots, m\}$ be the target training set, $T_{\text{te}} = \{(x_j^{\text{te}}, y_j^{\text{te}}) \mid j = 1, 2, \ldots, n\}$ the target testing set, and $T_{\text{au}} = \{(x_k^{\text{au}}, y_k^{\text{au}}) \mid k = 1, 2, \ldots, p\}$ the auxiliary training set. Assume that T_{tr} and T_{te} follow the same distribution $P_t(x, y)$ and T_{au} follow another distribution $P_s(x, y)$. In this section, we will present two kinds of MIDS construction methods, where one is direct and the other is indirect. Moreover, we will prove that the effect of the indirect method is equivalent to that of the direct method.

4.1.1. The Direct MIDS Construction Method of Scenario 1. With respect to the direct MIDS construction, we consider feature vector x and label y as a joint vector. We consider $T_t = T_{\text{tr}} \cup T_{\text{te}}$ and T_{au} as the target sample set and the source sample set, respectively. Thus we can use the above algorithm directly to obtain the MIDS. The MIDS construction method by considering feature vector x and label y as a joint vector is called the direct MIDS construction method.

Since feature vector x and label y are considered to be one joint vector, the dimension of samples will be increased. As described in Section 3.3, with the increase of the dimension, the running time of the MIDS algorithm will increase correspondingly. In the next section, we will present the indirect MIDS construction method, for which it is not necessary to consider x and y collectively and the MIDS is constructed according to feature vector x alone. Thus the indirect method can reduce effectively the running time and moreover it can be applied to the case where the target testing set contains only feature vectors.

4.1.2. The Indirect MIDS Construction Method of Scenario 1. With respect to the case where there are no class labels in the target testing set, we can construct the MIDS according to feature vector x alone. The MIDS construction method by

considering only feature vector x is called the indirect MIDS construction method. Now let $T_{\text{te}} = \{x_j^{\text{te}} \mid j = 1, 2, \ldots, n\}$ be target testing set. First of all, we present the detailed process of the indirect MIDS construction method.

Process 1. Remove all the labels of samples in T_t, and label the set composed by the remaining feature vectors as T_t^x. Similarly, remove all the labels of samples in T_{au}, and label the set composed by the remaining feature vectors as T_{au}^x.

Process 2. Use the MIDS construction algorithm to obtain a subset t_{au}^x of T_{au}^x.

Process 3. Add the class labels removed in Process 1 to each sample of t_{au}^x correspondingly, and thus we obtain a subset t_{au} of T_{au}.

Below we will prove that the effect of the indirect method is equivalent to that of the direct method.

Theorem 6. *The subset t_{au} obtained from the auxiliary set T_{au} by the indirect MIDS method follows the same distribution with the target set T_t; that is, $P_{S'}(x, y) = P_T(x, y)$, where $P_{S'}(X, Y)$ and $P_T(x, y)$ denote the distributions of t_{au} and T_t, respectively.*

Proof. Let $P_{S'}(x)$ and $P_T(x)$ denote the distributions of t_{au}^x and T_t^x, respectively. From the definition of conditional distribution, we have

$$\begin{aligned} P_T(x, y) &= P_T(y \mid x) P_T(x), \\ P_{S'}(x, y) &= P_{S'}(y \mid x) P_{S'}(x). \end{aligned} \tag{2}$$

As described in Process 3, t_{au} is obtained by adding the original class labels to t_{au}^x correspondingly. Thus the conditional probability is unchanged; that is, $P_{S'}(y \mid x) = P_S(y \mid x)$. From Definition 3, we know that $P_T(y \mid x) = P_S(y \mid x)$. Thus we can obtain that $P_{S'}(y \mid x) = P_T(y \mid x)$. Moreover since t_{au}^x is a MIDS of T_t^x, we can obtain that $P_{S'}(x) = P_T(x)$. Therefore we have $P_{S'}(x, y) = P_T(x, y)$. □

4.2. The MIDS Construction of Scenario 2. Let $T_{\text{tr}} = \{(x_i^{\text{tr}}, y_i^{\text{tr}}) \mid i = 1, 2, \ldots, m\}$ be target training set and $T_{\text{te}} = \{(x_j^{\text{te}}, y_j^{\text{te}}) \mid j = 1, 2, \ldots, n\}$ target testing set. Assume that T_{tr} and T_{te} follow distributions $P_{\text{tr}}(x, y)$ and $P_{\text{te}}(x, y)$, respectively. We consider T_{te} and T_{tr} as the target sample set and the source sample set, respectively. Thus we also can use the above algorithm directly to obtain the MIDS.

With respect to the case where there are no class labels in the target testing set, we can also use indirect method to construct the MIDS. Now let $T_{\text{te}} = \{x_j^{\text{te}} \mid j = 1, 2, \ldots, n\}$ be target testing set. The detailed process is as follows.

Process 1. Remove all the labels of samples in T_{tr}, and label the set composed by the remaining feature vectors as T_{tr}^x.

Process 2. Use the MIDS construction algorithm to obtain a subset t_{tr}^x of T_{tr}^x.

Process 3. Add the class labels removed in Process 1 to each sample of t_{tr}^x correspondingly, and thus we obtain a subset t_{tr} of T_{tr}.

4.3. Classification Algorithms. With the help of the MIDS construction method, we can make effective classification by traditional classification method. With respect to Scenario 1, we first construct the MIDS from the auxiliary set and then train a model on the set composed by the target training set and the MIDS. The pseudocodes of the two classification algorithms corresponding to the direct and indirect MIDS construction methods are shown in Algorithms 2 and 3, respectively. With respect to Scenario 2, we first construct the MIDS from the target training set and then train a model on this MIDS. The pseudocodes of the two classification algorithms corresponding to the direct and indirect MIDS construction methods are shown in Algorithms 4 and 5, respectively.

5. Experiments

In this section, we perform experiments to test the performance of the proposed classification algorithms. As proven in Section 4, the effect of the indirect method is equivalent to that of the direct method. Thus, we just test the performance of the two indirect classification algorithms. The experiment data in this section come from the UCI Machine Learning Repository [22]. All experiments are run on 2.00 GHz, Intel (R) Core (TM) i5-4200U CPU with 4 GB main memory under window 8.

5.1. The Experiment on Scenario 1

(1) Experimental Data Construction. This experiment is performed on 20 data sets, from which the target training set, the target testing set, and the auxiliary training set are constructed by the following principles.

Principle 1. The target training set and the target testing set should follow the same distribution.

Principle 2. The auxiliary training set and the target set should follow different distributions.

Principle 3. The size of the target training set is far less than that of the auxiliary training set.

We select auxiliary samples using a deliberately biased procedure (as in [19]). To describe our biased selection scheme, we need to define an additional random variable s_i for each point in the pool of possible training samples, where $s_i = 1$ means the ith sample is included and $s_i = 0$ indicates an excluded sample. In this paper, we discuss the classification problems under covariate shift, so we only consider the situation $P(s_i \mid x_i, y_i) = P(s_i \mid x_i)$. Below, we present the detailed method of experimental data construction. First of all, we select some samples randomly from the original data set to compose the target set, 1/4 of the data used for training and 3/4 for testing. Then, in the remaining samples, we consider a biased sampling scheme based on the input features to construct the auxiliary set. For convenience, in this paper, we only consider a biased sampling scheme based on one input feature. For example, with respect to breast cancer data set,

Require: the target training set $T_{\text{tr}} = \{(x_i^{\text{tr}}, y_i^{\text{tr}}) \mid i = 1, 2, \ldots, m\}$, the target testing set $T_{\text{te}} = \{(x_j^{\text{te}}, y_j^{\text{te}}) \mid j = 1, 2, \ldots, n\}$, the auxiliary sample set $T_{\text{au}} = \{(x_k^{\text{au}}, y_k^{\text{au}}) \mid k = 1, 2, \ldots, p\}$, a base learning algorithm Learner
Ensure: classification function $f(\cdot)$
$T_t = T_{\text{tr}} \cup T_{\text{te}}$; /* T_t denotes the target sample set* /
$t_{\text{au}} = \text{MIDS\_Construction}(T_t, T_{\text{au}})$; /* constructing the MIDS* /
$T_1 = t_{\text{au}} \cup T_{\text{tr}}$; /* T_1 denotes the new training set* /
$f(\cdot) = \text{Learner}(T_1)$; /* f denotes the function implemented by a base learning algorithm trained on the new training set T_1* /

Algorithm 2: The direct classification algorithm of Scenario 1.

Require: the target training set $T_{\text{tr}} = \{(x_i^{\text{tr}}, y_i^{\text{tr}}) \mid i = 1, 2, \ldots, m\}$, the target testing set $T_{\text{te}} = \{x_j^{\text{te}} \mid j = 1, 2, \ldots, n\}$, the auxiliary sample set $T_{\text{au}} = \{(x_k^{\text{au}}, y_k^{\text{au}}) \mid k = 1, 2, \ldots, p\}$, a base learning algorithm Learner
Ensure: classification function $f(\cdot)$
$T_{\text{tr}}^x = \text{select\_}x(T_{\text{tr}})$; /* T_{tr}^x denote the feature vectors set of T_{tr}* /
$T_t = T_{\text{tr}}^x \cup T_{\text{te}}$; /* T_t denote the target sample set* /
$T_{\text{au}}^x = \text{select\_}x(T_{\text{au}})$; /* T_{au}^x denote the feature vectors set of T_{au}* /
$t_{\text{au}}^x = \text{MIDS\_Construction}(T_t, T_{\text{au}}^x)$; /* constructing the MIDS* /
$t_{\text{au}} = \text{expand}(t_{\text{au}}^x, T_{\text{au}})$; /* add the corresponding labels to the feature vectors of t_{au}^x* /
$T_1 = t_{\text{au}} \cup T_{\text{tr}}$ /* T_1 denotes the new training set* /
$f(\cdot) = \text{Learner}(T_1)$; /* f denotes the function implemented by a base learning algorithm trained on the new training set T_1* /

Algorithm 3: The indirect classification algorithm of Scenario 1.

Require: the target training set $T_{\text{tr}} = \{(x_i^{\text{tr}}, y_i^{\text{tr}}) \mid i = 1, 2, \ldots, m\}$, the target testing set $T_{\text{te}} = \{(x_j^{\text{te}}, y_j^{\text{te}}) \mid j = 1, 2, \ldots, n\}$, a base learning algorithm Learner
Ensure: classification function $f(\cdot)$
$t_{\text{tr}} = \text{MIDS\_Construction}(T_{\text{tr}}, T_{\text{te}})$; /* constructing the MIDS* /
$f(\cdot) = \text{Learner}(t_{\text{tr}})$; /* f denotes the function implemented by a base learning algorithm trained on the new training set t_{tr}* /

Algorithm 4: The direct classification algorithm of Scenario 2.

Require: the target training set $T_{\text{tr}} = \{(x_i^{\text{tr}}, y_i^{\text{tr}}) \mid i = 1, 2, \ldots, m\}$, the target testing set $T_{\text{te}} = \{x_j^{\text{te}} \mid j = 1, 2, \ldots, n\}$, a base learning algorithm Learner
Ensure: classification function $f(\cdot)$
$T_{\text{tr}}^x = \text{select\_}x(T_{\text{tr}})$; /* T_{tr}^x denote the feature vectors set of T_{tr}* /
$t_{\text{tr}}^x = \text{MIDS\_Construction}(T_{\text{tr}}^x, T_{\text{te}})$; /* constructing the MIDS* /
$t_{\text{tr}} = \text{expand}(t_{\text{tr}}^x, T_{\text{tr}})$; /* add the corresponding labels to the feature vectors of t_{tr}^x* /
$f(\cdot) = \text{Learner}(t_{\text{tr}})$; /* f denotes the function implemented by a base learning algorithm trained on the new training set t_{tr}* /

Algorithm 5: The indirect classification algorithm of Scenario 2.

there are nine features, with integer values from 1 to 10. We consider a biased sampling scheme based on the first feature. Since smaller feature values predominate in the unbiased data, we consider a biased sampling scheme according to $P(s = 1 \mid x \leq 6) = 0.2$ and $P(s = 1 \mid x > 6) = 0.8$, where x is the value of the first feature.

(2) Experimental Methods. In the following, we compare our method (the indirect classification of Scenario 1, which is denoted by IDC1) against two other methods: the traditional classification algorithm (training on the set composed by target training samples and auxiliary samples) and the TrAdaBoost algorithm proposed in [15].

Table 1: Comparing ICD1 with SVM and TrAdaBoost.

Data set	SVM	IDC1	TrAdaBoost
Australian	0.756 ± 0.082	0.889 ± 0.052	0.836 ± 0.055
Balance	0.721 ± 0.072	0.846 ± 0.060	0.803 ± 0.062
Breast	0.698 ± 0.088	0.779 ± 0.086	0.722 ± 0.082
Cleveland	0.623 ± 0.072	0.697 ± 0.067	0.660 ± 0.061
Credit	0.792 ± 0.020	0.902 ± 0.025	0.826 ± 0.031
Diabetes	0.702 ± 0.056	0.816 ± 0.046	0.733 ± 0.039
Heart	0.729 ± 0.092	0.832 ± 0.070	0.830 ± 0.072
Ionosphere	0.837 ± 0.053	0.926 ± 0.052	0.903 ± 0.059
Iris	0.805 ± 0.061	0.976 ± 0.044	0.915 ± 0.049
Liver	0.672 ± 0.062	0.729 ± 0.038	0.732 ± 0.042
Page	0.892 ± 0.015	0.982 ± 0.021	0.901 ± 0.022
Sonar	0.705 ± 0.105	0.820 ± 0.102	0.767 ± 0.099
Thyroid	0.825 ± 0.081	0.961 ± 0.076	0.882 ± 0.075
Vehicle	0.708 ± 0.021	0.798 ± 0.025	0.790 ± 0.026
Voting	0.825 ± 0.046	0.963 ± 0.022	0.896 ± 0.028
Waveform21	0.721 ± 0.026	0.823 ± 0.017	0.821 ± 0.010
Waveform40	0.772 ± 0.025	0.838 ± 0.019	0.786 ± 0.029
Wine	0.793 ± 0.098	0.915 ± 0.092	0.898 ± 0.109
wdbc	0.803 ± 0.036	0.966 ± 0.023	0.909 ± 0.018
wpbc	0.682 ± 0.113	0.775 ± 0.099	0.726 ± 0.094

We select C-SVC [23] and Radial Basis Function (RBF) [1],

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{\beta}\right), \tag{3}$$

as the basic classification algorithm and kernel function, respectively, for the above three methods, where C is a penalty factor, β is a width parameter, and x and y are n-dimensional vectors in the original feature space. With respect to the multiclass data sets of the 20 selected data sets, we select one-against-all (1-v-r) approach [24], which is to transform a c-class problem into c-two-class problems, where one class is separated from the remaining ones. In this experiment, the best C and β are obtained by 10-fold cross-validation.

(3) Result Analysis. The three methods are compared on the selected 20 data sets. Five runs of 10-fold cross-validation are performed for each algorithm, and the average result is reported in Table 1, where the numbers following "±" are the standard deviations. The running time and parameter values of different algorithms are shown in Table 2, where N denote the number of iterations, C is a penalty factor, β is a width parameter, and t (ms) denotes the running time. N is set to 100 according to the parameter setting in [15], and the best C and β are obtained by 10-fold cross-validation.

As is shown in Table 1, the precision given by SVM is strictly lower than IDC1 and TrAdaBoost. Intuitively, this is true because, unlike SVM, IDC1 and TrAdaBoost are learning techniques designed for classification of Scenario 1. Furthermore Table 1 shows that IDC1 outperforms TrAdaBoost. In detail, pairwise two-tailed t-tests indicate that there are 16 data sets (*Australian*, *balance*, *breast*, *Cleveland*, *credit*, *diabetes*, *ionosphere*, *iris*, *page*, *sonar*, *thyroid*, *voting*, *waveform*40, *wine*, *wdbc*, and *wpbc*) where IDC1 is significantly more accurate than TrAdaBoost, while there is no significant difference on the remaining 4 data sets. We believe that the auxiliary set contain not only good samples, but also noisy data that caused the distribution of the auxiliary set different from that of the target set. The reason why IDC1 outperforms TrAdaBoost might be that ICD1 always employs the most important samples, which is included in the MIDS, to help the learners, while TrAdaBoost sometimes can not avoid using bad samples to help the learners.

Table 2: The running time and parameter values in Experiment 1.

Data set	SVM	IDC1	TrAdaBoost
	t, c, β	t, c, β	t, c, β, N
Australian	250.6, 10, 16	210.1, 100, 16	280.9, 50, 128, 100
Balance	322.1, 20, 8	295.2, 50, 32	353.2, 30, 256, 100
Breast	412.1, 100, 64	368.8, 10, 128	515.5, 100, 32, 100
Cleveland	243.2, 50, 32	205.1, 20, 64	301.5, 100, 64, 100
Credit	1129.2, 1000, 128	989.3, 500, 16	1250.6, 500, 64, 100
Diabetes	517.6, 1000, 32	410.1, 100, 128	620.3, 500, 32, 100
Heart	32.5, 100, 256	24.6, 30, 512	33.7, 10, 128, 100
Ionosphere	50.6, 20, 128	42.1, 1000, 32	52.3, 10, 256, 100
Iris	12.2, 10, 64	9.3, 50, 64	13.1, 100, 32, 100
Liver	19.6, 30, 32	18.2, 50, 32	22.3, 100, 128, 100
Page	22.1, 100, 128	18.8, 200, 64	22.8, 500, 16, 100
Sonar	9.6, 50, 1024	6.6, 100, 128	10.2, 80, 512, 100
Thyroid	32.3, 1000, 256	26.6, 500, 128	40.2, 60, 64, 100
Vehicle	1080.9, 100, 8	916.1, 50, 16	1120.3, 100, 128, 100
Voting	62.8, 60, 64	56.6, 100, 256	70.5, 80, 512, 100
Waveform21	632.1, 30, 32	527.1, 60, 64	636.9, 10, 16, 100
Waveform40	535.3, 50, 64	468.2, 100, 128	586.2, 200, 16, 100
Wine	50.2, 1000, 256	44.1, 500, 128	59.2, 200, 1024, 100
wdbc	289.2, 200, 128	266.1, 100, 256	299.5, 500, 32, 100
wpbc	267.1, 500, 512	256.9, 1000, 1024	268.2, 100, 512, 100

Moreover, as shown in Table 2, the running time of IDC1 is the least, while the running time of TrAdaBoost is the most. Thus IDC1 is the most effective model. The reason is below. In this experiment, traditional SVM algorithm uses the target sample set and the whole auxiliary sample set for training, while IDC1 employs the target sample set and only a subset of the auxiliary samples set for training. With respect to TrAdaBoost, the reason why its running time is the most is that it needs to do repeated iteration for a better performance.

Table 3: Comparing ICD2 with SVM and KLIEP.

Data set	SVM	IDC2	KLIEP
Australian	0.787 ± 0.046	0.869 ± 0.035	0.827 ± 0.063
Balance	0.754 ± 0.049	0.823 ± 0.028	0.819 ± 0.055
Breast	0.669 ± 0.059	0.789 ± 0.062	0.783 ± 0.049
Cleveland	0.601 ± 0.068	0.677 ± 0.034	0.671 ± 0.062
Credit	0.790 ± 0.056	0.892 ± 0.068	0.832 ± 0.051
Diabetes	0.802 ± 0.044	0.826 ± 0.039	0.822 ± 0.039
Heart	0.697 ± 0.083	0.818 ± 0.088	0.831 ± 0.062
Ionosphere	0.852 ± 0.046	0.902 ± 0.044	0.899 ± 0.056
Iris	0.901 ± 0.062	0.919 ± 0.056	0.912 ± 0.072
Liver	0.653 ± 0.065	0.745 ± 0.041	0.751 ± 0.053
Page	0.836 ± 0.016	0.932 ± 0.009	0.889 ± 0.022
Sonar	0.727 ± 0.223	0.856 ± 0.102	0.846 ± 0.235
Thyroid	0.859 ± 0.088	0.933 ± 0.128	0.929 ± 0.077
Vehicle	0.665 ± 0.056	0.728 ± 0.032	0.739 ± 0.046
Voting	0.795 ± 0.052	0.929 ± 0.031	0.915 ± 0.038
Waveform21	0.771 ± 0.019	0.832 ± 0.012	0.801 ± 0.026
Waveform40	0.756 ± 0.021	0.815 ± 0.018	0.818 ± 0.025
Wine	0.876 ± 0.096	0.922 ± 0.076	0.916 ± 0.091
wdbc	0.853 ± 0.021	0.935 ± 0.032	0.939 ± 0.012
wpbc	0.681 ± 0.101	0.765 ± 0.086	0.779 ± 0.121

5.2. The Experiment on Scenario 2

(1) Experimental Data Construction. Unlike the first experiment, in this experiment, we only construct the target training set and the target testing set. We should guarantee that the target training set and the target testing set follow different distributions. Like Scenario 1, we also use a deliberately biased procedure to construct the experimental data.

(2) Experimental Method. In the following, we compare our method (the indirect classification of Scenario 2, which is denoted by IDC2) against two other methods: the traditional classification algorithm and the KLIEP algorithm proposed in [21]. We also select C-SVC and RBF as the basic classification algorithm and kernel function, respectively, and select 1-v-r approach for the multiclass data sets.

(3) Result Analysis. Five runs of 10-fold cross-validation are performed for each algorithm, and the average result is reported in Table 3, where the numbers following "±" are the standard deviations. Also the running time and parameter values of different algorithms are shown in Table 4.

As is shown in Table 3, the precision given by SVM is strictly lower than IDC2 and KLIEP. Like the analysis of Scenario 1, this is true because, unlike SVM, IDC2 and KLIEP are learning techniques designed for classification of Scenario 2. Furthermore Table 3 shows that IDC2 is comparable to KLIEP that is a state-of-the-art algorithm. In detail, pairwise two-tailed t-tests indicate that there are 4 data sets (*Australian*, *credit*, *page*, and *waveform*21) where IDC2 outperforms KLIEP, and there are 3 data sets (*heart*, *vehicle*, and *wpbc*) where IDC2 performs a little worse than KLIEP, while there is no significant difference on the remaining 13 data sets. Therefore, ICD2 exhibits a new way of classification learning of Scenario 2.

Table 4: The running time and parameter values in Experiment 2.

Data set	SVM	IDC2	KLIEP
	t, c, β	t, c, β	t, c, β
Australian	225.6, 100, 64	209.1, 60, 64	218.9, 200, 128
Balance	262.1, 20, 128	225.2, 500, 64	278.2, 80, 16
Breast	312.1, 10, 128	288.2, 60, 128	315.5, 200, 16
Cleveland	196.2, 500, 256	160.1, 200, 1024	171.5, 80, 32
Credit	1029.2, 100, 64	899.3, 30, 256	1050.6, 20, 64
Diabetes	469.2, 200, 64	420.5, 300, 1024	425.5, 50, 16
Heart	28.8, 50, 512	22.1, 10, 16	30.3, 20, 64
Ionosphere	46.5, 10, 16	40.9, 200, 64	42.5, 80, 256
Iris	9.2, 100, 128	6.3, 60, 16	9.9, 200, 32
Liver	18.2, 60, 64	13.2, 100, 32	16.2, 60, 256
Page	32.1, 80, 1024	29.6, 1000, 16	32.6, 20, 16
Sonar	12.6, 20, 64	10.1, 200, 64	29.6, 60, 256
Thyroid	29.8, 100, 64	25.5, 400, 16	30.6, 100, 64
Vehicle	1286.2, 200, 16	1120.3, 100, 128	1225.6, 200, 16
Voting	44.6, 20, 16	38.2, 80, 512	50.5, 200, 1024
Waveform21	556.3, 600, 64	513.2, 300, 16	532.8, 100, 256
Waveform40	443.3, 800, 128	399.6, 200, 64	422.1, 600, 256
Wine	49.6, 20, 256	40.3, 100, 64	44.5, 50, 512
wdbc	266.6, 600, 64	236.2, 200, 256	299.5, 10, 128
wpbc	198.7, 10, 64	155.3, 500, 128	167.8, 200, 64

As shown in Table 4, IDC2 costs less time than SVM and KLIEP. Like the case in Experiment 1, the reason is that in the training process it uses only a subset of the training samples, while SVM and KLIEP have to employ the whole training samples for training.

6. Conclusion

In this paper, we first propose a MIDS construction method by matching sample numbers between the target set and auxiliary set in each feature subspace, and then we propose a novel approach for classification under covariate shift by training on a new data set. Our basic idea is to train a model on a newly constructed data set following approximately the target distribution. Our approach consists of two methods, including a direct method and an indirect one. The theoretical analysis shows that the indirect method is equivalent to the direct method for the MIDS construction, but with less running time. In our experiments, the two indirect algorithms, ICD1 and ICD2, demonstrate better classification abilities than traditional learning techniques. In addition, our method can consistently deal with both scenarios of covariate shift and the cases where there are multiple auxiliary data sets coming from different distributions.

We note that our method assumes that the source domain and the target domain have the same concept and can not deal with the case where they have different concepts, that is, the problem of concept drifts. In the future, we will try to extend the proposed method to address this issue.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is sponsored by the National Natural Science Foundation of China (nos. 61402246, 61402126, 61370083, 61370086, 61303193, and 61572268), a Project of Shandong Province Higher Educational Science and Technology Program (no. J15LN38), the National Research Foundation for the Doctoral Program of Higher Education of China (no. 20122304110012), the Natural Science Foundation of Heilongjiang Province of China (no. F201101), the Science and Technology Research Project Foundation of Heilongjiang Province Education Department (no. 12531105), and Heilongjiang Province Postdoctoral Research Start Foundation (no. LBH-Q13092).

References

[1] V. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, 1999.

[2] Z. Zhang, J. Cheng, J. Li, W. Bian, and D. Tao, "Segment-based features for time series classification," *The Computer Journal*, vol. 55, no. 9, pp. 1088–1102, 2012.

[3] J. Quinlan, *C4. 5: Programs for Machine Learning*, Morgan kaufmann, 1993.

[4] D. McSherry, "Mixed-initiative problem solving with decision trees," *Artificial Intelligence Review*, vol. 28, no. 1, pp. 17–33, 2007.

[5] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[6] T. Hu, X. Guo, X. Fu, and Y. Lv, "A neural network approach for solving linear bilevel programming problem," *Knowledge-Based Systems*, vol. 23, no. 3, pp. 239–242, 2010.

[7] C. Elkan, "The foundations of cost-sensitive learning," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI '01)*, vol. 2, pp. 973–978, Seattle, Wash, USA, August 1997.

[8] J. Yang, X. Yu, Z.-Q. Xie, and J.-P. Zhang, "A novel virtual sample generation method based on gaussian distribution," *Knowledge-Based Systems*, vol. 24, no. 6, pp. 740–748, 2011.

[9] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine Learning*, vol. 23, no. 1, pp. 69–101, 1996.

[10] P. Vivekanandan and R. Nedunchezhian, "Mining data streams with concept drifts using genetic algorithm," *Artificial Intelligence Review*, vol. 36, no. 3, pp. 163–178, 2011.

[11] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of Statistical Planning and Inference*, vol. 90, no. 2, pp. 227–244, 2000.

[12] P. Wu and T. Dietterich, "Improving svm accuracy by training on auxiliary data sources," in *Proceedings of the 21st International Conference on Machine Learning*, p. 110, ACM, 2004.

[13] X. Liao, Y. Xue, and L. Carin, "Logistic regression with an auxiliary data source," in *Proceedings of the 22nd International Conference on Machine Learning (ICML '05)*, pp. 505–512, ACM, August 2005.

[14] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, "To transfer or not to transfer," in *Proceedings of the Workshop Inductive Transfer Years Later (Nips '05)*, vol. 878, 2005.

[15] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, pp. 193–200, ACM, Oregon, Ore, USA, June 2007.

[16] Y. Freund and R. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational Learning Theory*, pp. 23–37, Springer, Berlin, Germany, 1995.

[17] A. Blum and T. Mitchell, "COMbining labeled and unlabeled data with co-training," in *Proceedings of the 11th annual conference on Computational learning theory*, pp. 92–100, ACM, 1998.

[18] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proceedings of the 21st International Conference on Machine Learning*, p. 114, ACM, 2004.

[19] J. Huang, A. Smola, A. Gretton, K. Borgwardt, and B. Scholkopf, "Correcting sample selection bias by unlabeled data," in *Advances in Neural Information Processing Systems*, vol. 19, pp. 601–608, MIT Press, Cambridge, Mass, USA, 2007.

[20] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI '95)*, vol. 2, pp. 1137–1145, Lawrence Erlbaum Associates, 1995.

[21] M. Sugiyama, S. Nakajima, H. Kashima, P. Von Buenau, M. Kawanabe, and M. Sugiyama, "Direct importance estimation with model selection and its application to covariate shift adaptation," in *Advances in Neural Information Processing Systems*, vol. 20, pp. 1433–1440, 2008.

[22] Z.-H. Zhou and Y. Jiang, "NeC4.5: neural ensemble based C4.5," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 6, pp. 770–773, 2004.

[23] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[24] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.