Singapore Management University
Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems | School of Information Systems

11-2014

Online Transfer Learning

Peilin ZHAO
Institute for Infocomm Research, Singapore

Steven C. H. HOI
Singapore Management University, CHHOI@smu.edu.sg

Jialei WANG
University of Chicago

Bin LI
Wuhan University

DOI: https://doi.org/10.1016/j.artint.2014.06.003

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

Part of the Databases and Information Systems Commons

Citation

ZHAO, Peilin; HOI, Steven C. H.; WANG, Jialei; and LI, Bin. Online Transfer Learning. (2014). *Artificial Intelligence*. 216, 76-102. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/2269

This Journal Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email libIR@smu.edu.sg.

Online Transfer Learning [☆]

Peilin Zhao [a], Steven C.H. Hoi [b,*], Jialei Wang [c], Bin Li [d]

[a] *Data Analytics Department, Institute for Infocomm Research, A*STAR, Singapore*
[b] *School of Information Systems, Singapore Management University, Singapore*
[c] *Department of Computer Science, The University of Chicago, USA*
[d] *Department of Finance, Economics and Management School, Wuhan University, 430072, PR China*

A R T I C L E   I N F O

Article history:
Received 19 April 2012
Received in revised form 3 June 2014
Accepted 16 June 2014
Available online 17 July 2014

Keywords:
Transfer learning
Online learning
Knowledge transfer

A B S T R A C T

In this paper, we propose a novel machine learning framework called "Online Transfer Learning" (OTL), which aims to attack an online learning task on a target domain by transferring knowledge from some source domain. We do not assume data in the target domain follows the same distribution as that in the source domain, and the motivation of our work is to enhance a supervised online learning task on a target domain by exploiting the existing knowledge that had been learnt from training data in source domains. OTL is in general very challenging since data in both source and target domains not only can be different in their class distributions, but also can be diverse in their feature representations. As a first attempt to this new research problem, we investigate two different settings of OTL: (i) OTL on homogeneous domains of common feature space, and (ii) OTL across heterogeneous domains of different feature spaces. For each setting, we propose effective OTL algorithms to solve online classification tasks, and show some theoretical bounds of the algorithms. In addition, we also apply the OTL technique to attack the challenging online learning tasks with concept-drifting data streams. Finally, we conduct extensive empirical studies on a comprehensive testbed, in which encouraging results validate the efficacy of our techniques.

1. Introduction

Transfer learning (TL) is an emerging family of machine learning techniques and has been actively studied in machine learning and AI communities in recent years [27]. In a regular transfer learning task, we assume two datasets, one from a source domain and the other from a target domain, are available where their data distributions or representations of the two domains can be very different. TL aims to build models from the target-domain dataset by exploring information from the source-domain dataset through some knowledge transferring process. Transfer learning is important for many applications where training data in a new domain may be limited or too expensive to collect. Despite being explored actively in literature [27,26,2,12,20], most existing approaches on transfer learning often have been studied in an offline/batch learning fashion, which assumes training data in the new domain is given a priori. Such an assumption may not always hold for some real applications where training examples may arrive in an online/sequential manner.

[☆] Code and datasets are available at http://www.stevenhoi.org/OTL/.
[*] Corresponding author.
E-mail addresses: zhaop@i2r.a-star.edu.sg (P. Zhao), chhoi@smu.edu.sg (S.C.H. Hoi), jialei@cs.uchicago.edu (J. Wang), binli.whu@whu.edu.cn (B. Li).

Unlike the existing transfer learning studies, this paper investigates a new framework of *Online Transfer Learning* (OTL) [33], which addresses the transfer learning problem in an online learning framework. Specifically, OTL makes two assumptions: (i) training data in the new domain arrives sequentially; and (ii) some classifiers/models had been learnt from source domains. Online transfer learning is beneficial to many real applications. Below we give two examples to illustrate some potential applications.

The first example application is for online spam detection, such as spam email filtering. Typically, a universal classifier is trained to detect the spam as accurately as possible by a batch learning approach [25]. However, a universal classifier might not be always optimal for every individual as different persons may have different opinions on the definition of spam. This raises an open question, i.e., how to transfer useful knowledge from the universal classifier to personalize the spam detector for every individual in an online learning manner. Such a problem can be naturally attacked by applying the proposed OTL technique, in which the key challenge is that the "spam" concept in the target domain for each individual can be very different from that in the source domain. For such problems, as we assume the feature spaces of both source and target domains are the same, we thus refer to this scenario as OTL on *homogeneous domains* of common feature space.

The second example application is for climate forecast in environment and climate science [24], such as weather forecast, earthquake and tsunami prediction. For example, consider a situation where new types of instruments or sensors are introduced to improve an existing weather forecast system. In this scenario, training data with new features will be added to the forecast system while old features are still retained. Such a problem also can be formulated as an online transfer learning task, which aims to build an improved forecasting system on the new domain with the augmented features by transferring the knowledge of the old classifier in the source domain. This task can be potentially more challenging than the previous example as the feature spaces of both source and target domains are different, making it difficult to train the classifier on the new data by a simple transfer from the old classifier. We thus refer to this scenario as OTL across *heterogeneous domains* of diverse feature spaces.

As a summary, this paper addresses two challenging scenarios: (i) OTL on homogeneous domains, and (ii) OTL across heterogeneous domains. One straightforward approach to OTL is based on a continuous learning strategy, which initializes a regular online learning algorithm on the target domain with the existing classifier learnt from source domains. However, such a simple solution suffers from some critical drawbacks: (i) when studying OTL on homogeneous domains, it could suffer from negative transfer (transferred knowledge is harmful to learning target task) whenever there exists much significant difference between two conditional probabilities; and (ii) when studying OTL across heterogeneous domains, the old classifiers cannot be trained continuously with the new features because of the inconsistence of the two feature spaces.

In addition to these two challenges, we note that online transfer learning is in general more challenging than a classical batch transfer learning task. This is because in an OTL task it is very hard to directly measure the distribution difference of the two domains as only a predictive model of the source domain is provided, and the data instances on the target domain arrive on-the-fly sequentially and typically must be predicted immediately. This work aims to investigate effective and efficient OTL techniques to tackle these challenges.

In particular, to tackle the first challenge, we propose two ensemble learning based strategies for transferring knowledge from source domain by combining two sets of classifiers built on different domains. The key idea is to dynamically update the combination weights for the base classifiers according to their online performance. We propose two effective algorithms and give theoretical bounds to justify their efficacy. To tackle the second challenge, we propose a co-regularization learning strategy for knowledge transfer, which can effectively handle the learning task on diverse feature spaces. The key idea of the proposed co-regularization strategy was partially inspired by the co-training principle for batch learning tasks (semi-supervised learning or multi-view learning) [5,29], which combines classifiers co-trained from different "views" of the same training instances to boost the learning efficacy.

Last but not least, we extend the idea of the proposed OTL technique to attack a real-world open challenge in data mining and machine learning, i.e., the concept-drifting data stream mining task [21] where the underlying distributions and concepts often change over time. Despite being studied extensively in literature, it remains a critical open challenge for the existing approaches based on either batch learning or online learning techniques. In this paper, we propose an effective algorithm to attack this challenge based on a natural extension of the proposed OTL technique.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the proposed OTL framework and addresses the homogeneous and heterogeneous OTL tasks for classification. Section 4 presents the extension of OTL to address concept-drift online learning tasks. Section 5 gives our experimental results and discussions, and Section 6 concludes this work. Finally, we note that a short version of this work has appeared in the conference proceedings of ICML-2010 [33]. In contrast to the conference paper, a substantial amount of new contents and extensions have been included in this journal article.

2. Related work

Our work is mainly related to two machine learning topics: *online learning* and *transfer learning*. Below reviews some important related work.

Online learning (OL) has been extensively studied for years [28,7,9,32,34–36,30,19]. Unlike typical machine learning methods that assume training examples are available before the learning task, online learning is more appropriate for some real-world problems where training data arrives sequentially. Due to the merits of attractive efficiency and scalability, various online learning methods have been proposed. One well-known approach is the Perceptron algorithm [28,14], which performs a simple update on the classification model when an incoming example is misclassified. Recently many online learning algorithms have been proposed based on the criterion of maximum margin [23,9]. One example is the Passive–Aggressive (PA) method [9], which updates the model when a new example is misclassified or its classification confidence is smaller than some predefined threshold. Although the general online learning algorithms (e.g., Perceptron and PA) have solid theoretical guarantees and performs well on many applications, they usually keep the weights of the existing support vectors fixed during the whole learning process, which is clearly insufficient. To solve this issue, double updating online learning (DUOL) [34] is proposed to not only update the weight of the current support vector but also the weight of one existing support vector, which conflicts the most with the current one. Besides, traditional online learning tries to maximize the accuracy of the model while accuracy is unsuitable to many applications and scenarios. To tackle this problem, online AUC maximization (OAM) [35] is proposed to online maximize the AUC performance of the model. In addition, typical online learning usually stores all the misclassified examples as Support Vectors (SVs), which may result in high computational and memory costs. To deal with this problem, the researchers [36] have proposed a bounded online gradient descent algorithm (BOGD) to keep the number of stored SVs less than a pre-defined threshold. Moreover, most of online learning algorithms only exploit the first order information and assign all features the same learning rate, which may suffer from slow convergence rates. This can be solved by second order online learning [30], which not only use the first order but also the second order information of the examples. More extensive surveys for online learning can be found in [8,19].

Transfer learning (TL) has been actively studied recently [27]. The goal of TL is to extract knowledge from one or more source domains and then apply them to solve a learning task on a target domain. A variety of TL methods have been proposed in different learning settings. These methods can be roughly classified into three categories: *inductive*, *transductive*, and *unsupervised* learning approaches. Inductive TL [26] aims to induce the model in the target domain with the aid of knowledge transferred from the source domains; transductive TL [2,12] aims to extract the knowledge from the source domains to improve the prediction tasks in the target domain without labeled data in the target domain; while unsupervised TL aims to resolve unsupervised learning tasks in the target domain [31,11]. Moreover, according to different feature representation, TL can be classified as *homogeneous* TL or *heterogeneous* TL [1] where the feature spaces of source and target domains can be different. A comprehensive survey on batch transfer learning can be found in [27].

Although both online learning and transfer learning have been actively studied in literature, to the best of our knowledge, we are the first to formulate transfer learning in an online learning framework [33]. In addition, it is also important to note that OTL is different from *online multi-task learning* [13], which aims to learn multiple tasks in parallel in an online learning setting. Finally, our work is also related to some existing studies on concept-drifting learning and mining in machine learning and data mining literature [17,22,6]. In data mining, most existing work usually adapt some batch learning algorithms to attack concept-drift learning/mining tasks using various instance selection/weighting strategies and heuristics. As our work is focused on online learning methodology, we exclude the detailed discussions on a large body of related work on batch learning studies in data mining. We refer readers to some comprehensive surveys in data mining [37,15]. Below we review some representative work on online learning methods to handle concept drift in machine learning.

In machine learning literature, various online learning methods have been proposed to handle concept drift learning [17,3,18,16]. The well-known techniques include several variants of Perceptron-style algorithms [4,6,10]. For example, the Shifting Perceptron [6] attempts to tackle the concept drift challenge by diminishing the important of early updates by introducing some time-changing decaying factor. Most of the existing techniques usually assume some fixed or slowly changing input distribution, and typically cannot effectively handle sudden concept drift in a challenging real-world scenario. Unlike the existing approaches, we extend the idea of online transfer learning to tackle the problem of online learning with concept drift which can tackle sudden concept drift more effectively than the state-of-the-art approaches.

3. Online Transfer Learning

In this section, we first present a framework of Online Transfer Learning (OTL) for classification, and then propose algorithms to solve the OTL tasks under two different settings. We note that although the following discussion is focused on classification tasks, the similar techniques and principles could be generalized to other data mining and machine learning tasks, such as regression or ranking.

3.1. Problem formulation

Let us denote by $\mathcal{X}_s \times \mathcal{Y}_s$ the source/old data space, where $\mathcal{X}_s = \mathbb{R}^m$ and $\mathcal{Y}_s = \{-1, +1\}$. Assume that a source classifier is a linear vector $\mathbf{v} \in \mathbb{R}^m$. Typically the source classifier $\mathbf{v}$ can be obtained by applying existing learning techniques, such as online learning by the Perceptron algorithm [28,14] or regular batch learning by support vector machines (SVM).

The goal of an online transfer learning (OTL) task is to learn some prediction function $f(\mathbf{x}_t)$ on a target domain in an online fashion from a sequence of examples $\{(\mathbf{x}_t, y_t) \mid t = 1, \ldots, T\}$ in some data space $\mathcal{X} \times \mathcal{Y}$. Without loss of generality, we assume a linear prediction model is used for the prediction function, i.e., $f(\mathbf{x}_t) = sign(\mathbf{w}_t^\top \mathbf{x}_t)$.

Specifically, during the OTL task, at the t-th trial of online learning task, the learner receives an instance $\mathbf{x}_t$, and the goal of online learning is to find a good prediction function such that the predicted class label $sign(\mathbf{w}_t^\top \mathbf{x}_t)$ can match its true class label y_t. The key challenge of OTL is how to effectively transfer the knowledge from the old/source domain to the new/target domain for improving the online learning performance. Next, we study OTL in two different settings: homogeneous OTL vs. heterogeneous OTL.

3.2. Online Transfer Learning on homogeneous domains

We start by studying the homogeneous OTL, in which we assume the source and target domains share the same feature space, i.e., $\mathcal{X} = \mathcal{X}_s$ and $\mathcal{Y} = \mathcal{Y}_s$. One key challenge of this task is to address the covariate shift problem. This raises the challenge of transferring knowledge from source domain to target domain.

The basic idea of our OTL solution is based on the ensemble learning strategy. In particular, we first construct an entirely new prediction function $\mathbf{w}$ only from the data in the target domain in an *online* fashion, and then learn an ensemble prediction function that is the mixture of both the old and the new prediction functions, i.e., $\mathbf{v}$ and $\mathbf{w}$, which thus can transfer the knowledge from the source domain. The remaining issue is then how to effectively combine the two prediction functions for handling the covariate shift issue.

In order to effectively combine the two prediction functions $\mathbf{v}$ and $\mathbf{w}_t$ at the t-trial of the online learning task, we introduce two combination weighting parameters, $\alpha_{1,t}$ and $\alpha_{2,t}$, for the two prediction functions respectively. At the t-th step, given an instance $\mathbf{x}_t$, we predict its class label by the following prediction function:

$$\hat{y}_t = \text{sign}\left(\alpha_{1,t}\Pi\left(\mathbf{v}^\top \mathbf{x}_t\right) + \alpha_{2,t}\Pi\left(\mathbf{w}_t^\top \mathbf{x}_t\right) - \frac{1}{2}\right) \tag{1}$$

where $\Pi(z)\ \forall z \in \mathbb{R}$ is a projection function, i.e., $\Pi(z) = \max(0, \min(1, \frac{z+1}{2}))$. At the beginning of the OTL task, we simply initialize $\alpha_{1,1} = \alpha_{2,1} = \frac{1}{2}$. In order to perform effective transfer for the subsequent trials of the OTL task, in addition to updating the function $\mathbf{w}_{t+1}$ by some online learning method, e.g. the PA algorithm [9], we expect the two weights of both prediction functions, i.e., $\alpha_{1,t}$ and $\alpha_{2,t}$, should be adjusted dynamically. We thus suggest the following scheme for updating the weights:

$$\alpha_{1,t+1} = \frac{\alpha_{1,t}s_t(\mathbf{v})}{\alpha_{1,t}s_t(\mathbf{v}) + \alpha_{2,t}s_t(\mathbf{w}_t)}, \qquad \alpha_{2,t+1} = \frac{\alpha_{2,t}s_t(\mathbf{w}_t)}{\alpha_{1,t}s_t(\mathbf{v}) + \alpha_{2,t}s_t(\mathbf{w}_t)} \tag{2}$$

where $s_t(\mathbf{u}) = \exp\{-\eta\ell^*(\Pi(\mathbf{u}^\top \mathbf{x}_t), \Pi(y_t))\}$, $\forall \mathbf{u} \in \mathbb{R}^m$ and $\ell^*(z, y)$ is a loss function which is set to $\ell^*(z, y) = (z - y)^2$ in our approach. Finally, Algorithm 1 summarizes the proposed HomOTL-I algorithm. Before we analyze the mistake bound of the proposed algorithm, we first introduce a proposition as follows.

Algorithm 1 Homogeneous Online Transfer Learning (**HomOTL-I**).

```
**Input:** the old classifier $\mathbf{v} \in \mathbb{R}^m$ and initial trade off $C$
Initialize $\mathbf{w}_1 = 0$ and weights $\alpha_{1,1} = \alpha_{2,1} = \frac{1}{2}$
**for** $t = 1, 2, \ldots, T$ **do**
    receive instance: $\mathbf{x}_t \in \mathcal{X}$
    predict $\hat{y}_t = sign(\alpha_{1,t}\Pi(\mathbf{v}^\top \mathbf{x}_t) + \alpha_{2,t}\Pi(\mathbf{w}_t^\top \mathbf{x}_t) - \frac{1}{2})$
    receive correct label: $y_t \in \{-1, +1\}$
    compute $\alpha_{1,t+1} = \frac{\alpha_{1,t}s_t(\mathbf{v})}{\alpha_{1,t}s_t(\mathbf{v})+\alpha_{2,t}s_t(\mathbf{w}_t)}$,   $\alpha_{2,t+1} = \frac{\alpha_{2,t}s_t(\mathbf{w}_t)}{\alpha_{1,t}s_t(\mathbf{v})+\alpha_{2,t}s_t(\mathbf{w}_t)}$
    suffer loss: $\ell_t = [1 - y_t\mathbf{w}_t^\top \mathbf{x}_t]_+$
    **if** $\ell_t > 0$ **then**
        $\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t y_t \mathbf{x}_t$, where $\tau_t = \min\{C, \ell_t/\|\mathbf{x}_t\|^2\}$
    **end if**
**end for**
```

Proposition 1. *When using the square loss $\ell^*(z, y) = (z - y)^2$ for $z \in [0, 1]$ and $y \in [0, 1]$, the above exponentially weighting update method and setting $\eta = 1/2$, we have the bound of the ensemble algorithm as:*

$$\begin{aligned}&\sum_{t=1}^{T}\ell^*\left(\alpha_{1,t}\Pi\left(\mathbf{v}^\top \mathbf{x}_t\right) + \alpha_{2,t}\Pi\left(\mathbf{w}_t^\top \mathbf{x}_t\right), \Pi(y_t)\right)\\ &\quad\le 2\ln 2 + \min\left\{\sum_{t=1}^{T}\ell^*\left(\Pi\left(\mathbf{v}^\top \mathbf{x}_t\right), \Pi(y_t)\right), \sum_{t=1}^{T}\ell^*\left(\Pi\left(\mathbf{w}_t^\top \mathbf{x}_t\right), \Pi(y_t)\right)\right\}\end{aligned} \tag{3}$$

The proof of the above proposition is in Appendix A. By Proposition 1, we can derive the mistake bound of the HomOTL-I algorithm as follows.

Theorem 1. *Let us denote by M the number of mistakes made by the HomOTL-I algorithm, we then have M bounded from above by:*

$$M \le 4\min\{\varSigma_{\mathbf{v}}, \varSigma_{\mathbf{w}}\} + 8\ln 2 \tag{4}$$

where $\varSigma_{\mathbf{v}} = \sum_{t=1}^{T} \ell^*(\varPi(\mathbf{v}^\top \mathbf{x}_t), \varPi(y_t))$ *and* $\varSigma_{\mathbf{w}} = \sum_{t=1}^{T} \ell^*(\varPi(\mathbf{w}_t^\top \mathbf{x}_t), \varPi(y_t))$.

Proof. First notice that whenever there is a mistake at some t-th step, we should have $|\alpha_{1,t}\varPi(\mathbf{v}^\top \mathbf{x}_t) + \alpha_{2,t}\varPi(\mathbf{w}_t^\top \mathbf{x}_t) - \varPi(y_t)| \ge \frac{1}{2}$. Thus, we haves

$$\sum_{t=1}^{T} \ell^*\big(\alpha_{1,t}\varPi\big(\mathbf{v}^\top \mathbf{x}_t\big) + \alpha_{2,t}\varPi\big(\mathbf{w}_t^\top \mathbf{x}_t\big), \varPi(y_t)\big)$$
$$= \sum_{t=1}^{T} \big(\alpha_{1,t}\varPi\big(\mathbf{v}^\top \mathbf{x}_t\big) + \alpha_{2,t}\varPi\big(\mathbf{w}_t^\top \mathbf{x}_t\big) - \varPi(y_t)\big)^2 \ge \frac{1}{4}M$$

Combining the above fact with Proposition 1, we have

$$\frac{1}{4}M \le \min\{\varSigma_{\mathbf{v}}, \varSigma_{\mathbf{w}}\} + 2\ln 2$$

where $\varSigma_{\mathbf{v}} = \sum_{t=1}^{T} \ell^*(\varPi(\mathbf{v}^\top \mathbf{x}_t), \varPi(y_t))$ and $\varSigma_{\mathbf{w}} = \sum_{t=1}^{T} \ell^*(\varPi(\mathbf{w}_t^\top \mathbf{x}_t), \varPi(y_t))$. The theorem follows directly by multiplying 4 at both sides of the above. □

Remark. To better understand the mistake bound, we denote by $M_{\mathbf{v}}$ and $M_{\mathbf{w}}$ the mistake bounds of model $\mathbf{v}$ and $\mathbf{w}_t$, respectively. We first note that $\ell^*(\varPi(\mathbf{v}^\top \mathbf{x}_t), \varPi(y_t))$ is the upper bound of $\frac{1}{4}M_{\mathbf{v}}$ instead of $M_{\mathbf{v}}$ (because ℓ^* is a square loss and both $\varPi(\mathbf{v}^\top \mathbf{x}_t)$ and $\varPi(y_t)$ are normalized to $[0, 1]$); similarly, $\ell^*(\varPi(\mathbf{w}_t^\top \mathbf{x}_t), \varPi(y_t))$ is the upper bound of $\frac{1}{4}M_{\mathbf{w}}$. Further, if we assume $\ell^*(\varPi(\mathbf{v}^\top \mathbf{x}_t), \varPi(y_t)) \approx \frac{1}{4}M_{\mathbf{v}}$ and $\ell^*(\varPi(\mathbf{w}_t^\top \mathbf{x}_t), \varPi(y_t)) \approx \frac{1}{4}M_{\mathbf{w}}$, we have $M \le \min\{M_{\mathbf{v}}, M_{\mathbf{w}}\} + 8\ln 2$. This gives a strong theoretical support for the HomOTL-I algorithm. However, please note that: while HomOTL-I will only make a constant number of mistakes more than the best base learner, the best base learner may still suffer some regret. Despite the nice result in theory, we note this bound may be further improved so that it can tell us exactly how much we can leverage the classifier from the source domain to improve over the target domain. However, this can be a very hard open challenge since an online transfer learning task is in general more challenging than a classical batch transfer learning task because in an OTL task only a linear classifier is stored for a source domain and the new instances for the target domain are received online sequentially, it is hard/almost impossible to directly compare the distributions of the source domain and the target domain.

In addition to the above loss-based updating algorithm, we also provided the following mistake-driven Algorithm 2.

Algorithm 2 Homogeneous Online Transfer Learning (**HomOTL-II**).

```
Input: the old classifier $\mathbf{v} \in \mathbb{R}^m$ and initial trade off $C$, discount weight $\beta \in (0, 1)$
Initialize $\mathbf{w}_1 = 0$ and weights $\theta_{i,1} = 1$, $\alpha_{i,1} = \frac{1}{2}$ where $i = 1, 2$
for $t = 1, 2, \ldots, T$ do
    receive instance: $\mathbf{x}_t \in \mathcal{X}$
    predict $\hat{y}_t = sign[\alpha_{1,t}\, sign(\mathbf{v}^\top \mathbf{x}_t) + \alpha_{2,t}\, sign(\mathbf{w}_t^\top \mathbf{x}_t)]$
    receive correct label: $y_t \in \{-1, +1\}$
    compute $z_{1,t} = \mathbb{I}_{(y_t \mathbf{v}^\top \mathbf{x}_t \le 0)}$ and $z_{2,t} = \mathbb{I}_{(y_t \mathbf{w}_t^\top \mathbf{x}_t \le 0)}$
    update $\theta_{i,t+1} = \theta_{i,t}\beta^{z_{i,t}}$, where $i = 1, 2$
    suffer loss: $\ell_t = [1 - y_t \mathbf{w}_t^\top \mathbf{x}_t]_+$
    if $\ell_t > 0$ then
        $\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t y_t \mathbf{x}_t$, where $\tau_t = \min\{C, \ell_t/\|\mathbf{x}_t\|^2\}$
    end if
    $\alpha_{i,t} = \frac{\theta_{i,t}}{\Theta_t}$, where $i = 1, 2$ and $\Theta_t = \sum_{i=1}^{2} \theta_{i,t}$
end for
```

In this framework, we use $\theta_{i,t}$ to denote the combination weight for the two classifiers at round t, which is set to 1 at the initial round. For each learning round, we update the weight $\theta_{i,t}$ by following the Hedge algorithm as follows:

$$\theta_{i,t+1} = \theta_{i,t}\beta^{z_{i,t}}$$

where $\beta \in (0, 1)$ is a discount weight parameter, which is employed to penalize the classifier that performs incorrect prediction at each learning step, and $z_{i,t}$ indicates if the corresponding classifier makes a mistake on the prediction of the example $\mathbf{x}_t$.

Next we derive a theorem to show the mistake bound for Algorithm 2.

Theorem 2. *After receiving a sequence of T training examples, denoted by $\mathcal{L} = \{(x_t, y_t), t = 1, \ldots, T\}$, the number of mistakes M made by running Algorithm 2, denoted by*

$$M = \sum_{t=1}^{T} \mathbb{I}_{(y_t \hat{y}_t \le 0)} = \sum_{t=1}^{T} \mathbb{I}_{(\sum_{i=1}^{2} \alpha_{i,t} z_{i,t} \ge 0.5)}$$

is bounded as follows

$$M \le \frac{2\ln(1/\beta)}{1-\beta} \min_{i \in \{1,2\}} M_i + \frac{2\ln 2}{1-\beta} \tag{5}$$

where $M_i = \sum_{t=1}^{T} z_{i,t}$ for $i = 1, 2$. By choosing $\beta = \frac{\sqrt{T}}{\sqrt{T}+\sqrt{\ln 2}}$, we have

$$M \le 2\left(\left(1 + \sqrt{\frac{\ln 2}{T}}\right) \min_{i \in \{1,2\}} M_i + \ln 2 + \sqrt{T \ln 2}\right)$$

Proof. We bound $\ln(\Theta_{t+1}/\Theta_t)$ from both the above and the below. Firstly, to upper bound $\ln(\Theta_{t+1}/\Theta_t)$, we have

$$\ln \frac{\Theta_{t+1}}{\Theta_t} = \ln\left(\sum_{i=1}^{2} \frac{\theta_{i,t}}{\Theta_t} \beta^{z_{i,t}}\right) \le -(1-\beta) \sum_{i=1}^{2} \alpha_{i,t} z_{i,t}$$

By adding the inequalities of all trials, we have

$$\ln\left(\frac{\Theta_{T+1}}{\Theta_1}\right) \le -(1-\beta) \sum_{t=1}^{T} \sum_{i=1}^{2} \alpha_{i,t} z_{i,t}$$

On the other hand, we have $\ln(\Theta_{T+1}/\Theta_1)$ lower bounded as follows

$$\ln\left(\frac{\Theta_{T+1}}{\Theta_1}\right) \ge \ln \frac{\theta_{i,T+1}}{\Theta_1} = -\ln(1/\beta) \sum_{t=1}^{T} z_{i,t} - \ln 2$$

Since

$$\sum_{t=1}^{T} \mathbb{I}_{(\sum_{i=1}^{2} \alpha_{i,t} z_{i,t} \ge 0.5)} \le 2 \sum_{t=1}^{T} \sum_{i=1}^{2} \alpha_{i,t} z_{i,t},$$

we have the result in the theorem. Finally, to suggest the value for parameter β, by assuming $\sum_{t=1}^{T} z_{i,t} \le T$ and $\frac{\ln(1/\beta)}{1-\beta} \le 1/\beta$, we can derive the solution for parameter β as follows: $\beta = \frac{\sqrt{T}}{\sqrt{T}+\sqrt{\ln 2}}$, which leads to the final result as stated in the theorem. □

3.3. Online Transfer Learning across heterogeneous domains

In this section, we study the OTL problem across heterogeneous domains where the source and target domains have different feature spaces.

Heterogeneous OTL is generally more challenging than homogeneous OTL. It is very hard, if not completely impossible, to perform knowledge transfer if the feature spaces of source and target domains are not overlapped at all. To simplify the difficulty a bit, we assume the feature space of the source domain is a subset of that of the target domain. As two feature spaces are not the same, we cannot directly apply the algorithm in the previous section. Below we propose a multi-view approach to solve the challenge.

Formally, we denote the data on the target domain as: $\{(\mathbf{x}_t, y_t) \mid t = 1, \ldots, T\}$, where $\mathbf{x}_t \in \mathcal{X} = \mathbb{R}^n \supset \mathbb{R}^m$ and $y_t \in \{-1, +1\}$. Without loss of generality, we assume the first m dimensions of $\mathcal{X}$ represent the old feature space $\mathcal{X}_s$. In the multi-view setting, we split each data instance $\mathbf{x}_t$ into two instances $\mathbf{x}_{1,t} \in \mathcal{X}_s$ and $\mathbf{x}_{2,t} \in \mathcal{X}/\mathcal{X}_s$.

The key idea of our heterogeneous OTL method is to adopt a co-regularization principle of online learning two classifiers $\mathbf{w}_{1,t}$ and $\mathbf{w}_{2,t}$ simultaneously from the two views, and predict an unseen example on the target domain by $\hat{y}_t = sign(\frac{1}{2}(\mathbf{w}_{1,t}^{\top}\mathbf{x}_{1,t} + \mathbf{w}_{2,t}^{\top}\mathbf{x}_{2,t}))$.

For the specific algorithm, we initialize the classifier for the first view by setting $\mathbf{w}_{1,1} = \mathbf{v}$, and setting $\mathbf{w}_{2,1} = \mathbf{0}$ for the second view. For a new example in the online learning task, we update the new functions $\mathbf{w}_{1,t+1}$ and $\mathbf{w}_{2,t+1}$ by the following *co-regularization* optimization:

$$(\mathbf{w}_{1,t+1}, \mathbf{w}_{2,t+1}) = \arg\min_{\mathbf{w}_1 \in \mathbb{R}^m, \mathbf{w}_2 \in \mathbb{R}^{n-m}} \frac{\gamma_1}{2}\|\mathbf{w}_1 - \mathbf{w}_{1,t}\|^2 + \frac{\gamma_2}{2}\|\mathbf{w}_2 - \mathbf{w}_{2,t}\|^2 + C\ell(\mathbf{w}_1, \mathbf{w}_2; t) \tag{6}$$

where γ_1, γ_2 and C are positive parameters, and the loss is defined as:

$$\ell(\mathbf{w}_1, \mathbf{w}_2; t) = \left[1 - y_t \frac{1}{2}\left(\mathbf{w}_1^\top \mathbf{x}_{1,t} + \mathbf{w}_2^\top \mathbf{x}_{2,t}\right)\right]_+ \tag{7}$$

Intuitively, the above updating method aims to make the updated ensemble classifier be able to classify the new observed example $(\mathbf{x}_t, y_t)$ correctly, and to force the two-view classifiers without deviating too much from the previous classifiers $(\mathbf{w}_{1,t}, \mathbf{w}_{2,t})$ via the first two regularization terms.

The above optimization enjoys a closed-form solution as shown in Proposition 2. To simplify our discussion, we introduce notations $z_{1,t} = \|\mathbf{x}_{1,t}\|^2$ and $z_{2,t} = \|\mathbf{x}_{2,t}\|^2$.

Proposition 2. *For the optimization problem (6), its solution can be expressed as follows:*

$$\mathbf{w}_{i,t+1} = \mathbf{w}_{i,t} + \frac{\tau_t}{2\gamma_i}\mathbf{x}_{i,t} \quad i = 1, 2 \tag{8}$$

where $\tau_t = \min\{C, \frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2 + z_{2,t}\gamma_1}\}$ *and* $\ell_t = \ell(\mathbf{w}_{1,t}, \mathbf{w}_{2,t}; t)$.

The proof of the proposition is given in Appendix A. By this proposition, we summarize the proposed "Heterogeneous Online Transfer Learning" (HetOTL) algorithm in Algorithm 3.

Algorithm 3 Heterogeneous Online Transfer Learning (**HetOTL**).

```
INPUT: the old classifier $\mathbf{v} \in \mathbb{R}^m$ and parameters $\gamma_1$, $\gamma_2$ and $C$
Initialize $\mathbf{w}_{1,1} = \mathbf{v}$ and $\mathbf{w}_{2,1} = \mathbf{0}$
for $t = 1, 2, \ldots, T$ do
    receive instance: $\mathbf{x}_t \in \mathcal{X}$
    predict: $\hat{y}_t = sign(\frac{1}{2}(\mathbf{w}_{1,t}^\top \mathbf{x}_{1,t} + \mathbf{w}_{2,t}^\top \mathbf{x}_{2,t}))$
    receive correct label: $y_t \in \{-1, +1\}$
    suffer loss: $\ell_t = [1 - y_t \frac{1}{2}(\mathbf{w}_{1,t}^\top \mathbf{x}_{1,t} + \mathbf{w}_{2,t}^\top \mathbf{x}_{2,t})]_+$
    if $\ell_t > 0$ then
        $\tau_t = \min\{C, \frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2 + z_{2,t}\gamma_1}\}$
        $\mathbf{w}_{1,t+1} = \mathbf{w}_{1,t} + \frac{\tau_t}{2\gamma_1} y_t \mathbf{x}_{1,t}$, $\mathbf{w}_{2,t+1} = \mathbf{w}_{2,t} + \frac{\tau_t}{2\gamma_2} y_t \mathbf{x}_{2,t}$
    end if
end for
```

Before we prove the mistake bound for the HetOTL algorithm, we first introduce a lemma.

Lemma 1. *Let* $(\mathbf{x}_t, y_t), t = 1, \ldots, T$ *be a sequence of examples, where* $\mathbf{x}_t \in \mathbb{R}^n$ *and* $y_t \in \{-1, +1\}$ *for all t. After we split the instance* $\mathbf{x}_t$ *into two views* $(\mathbf{x}_{1,t}, \mathbf{x}_{2,t})$, *for any* $\mathbf{w}_1 \in \mathbb{R}^m$ *and* $\mathbf{w}_2 \in \mathbb{R}^{n-m}$, *we have the following bound:*

$$\sum_{t=1}^{T} \tau_t \left(\ell_t - \ell(\mathbf{w}_1, \mathbf{w}_2; t) - \left(\frac{z_{1,t}}{8\gamma_1} + \frac{z_{2,t}}{8\gamma_2}\right)\tau_t\right) \le \frac{\gamma_1}{2}\|\mathbf{v} - \mathbf{w}_1\|^2 + \frac{\gamma_2}{2}\|\mathbf{w}_2\|^2 \tag{9}$$

where $\ell(\mathbf{w}_1, \mathbf{w}_2; t)$ *is given in Eqn. (7) and* $\ell_t = \ell(\mathbf{w}_{1,t}, \mathbf{w}_{2,t}; t)$.

The proof of the lemma is given in Appendix A. Using Lemma 1, we can show the following theorem for the mistake bound of the proposed HetOTL algorithm.

Theorem 3. *Let* $(\mathbf{x}_t, y_t), t = 1, \ldots, T$ *be a sequence of examples, where* $\mathbf{x}_t \in \mathbb{R}^n$ *and* $y_t \in \{-1, +1\}$ *for all t. If we split the instance* $\mathbf{x}_t$ *into two views* $(\mathbf{x}_{1,t}, \mathbf{x}_{2,t})$, *so that* $z_{1,t} \le R_1$ *and* $z_{2,t} \le R_2$ $t = 1, \ldots, T$. *Then for any* $\mathbf{w}_1 \in \mathbb{R}^m$ *and* $\mathbf{w}_2 \in \mathbb{R}^{n-m}$, *the number of mistakes M made by the proposed HetOTL algorithm is bounded from above as:*

$$M \le \frac{1}{\tau}\left(\gamma_1\|\mathbf{v} - \mathbf{w}_1\|^2 + \gamma_2\|\mathbf{w}_2\|^2 + 2C\sum_{t=1}^{T}\ell(\mathbf{w}_1, \mathbf{w}_2; t)\right) \tag{10}$$

where $\tau = \min\{C, \frac{4\gamma_1\gamma_2}{R_1\gamma_2 + R_2\gamma_1}\}$.

Proof. Firstly, $\tau_t = \min\{C, \frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2 + z_{2,t}\gamma_1}\} \le C$ implies $\tau_t\ell(\mathbf{w}_1, \mathbf{w}_2; t) \le C\ell(\mathbf{w}_1, \mathbf{w}_2; t)$. Combining with $\tau_t = \min\{C, \frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2 + z_{2,t}\gamma_1}\}$ $\le \frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2 + z_{2,t}\gamma_1}$, we thus have

$$\begin{aligned}
&\sum_{t=1}^{T}\tau_t\left(\ell_t-\ell(\mathbf{w}_1,\mathbf{w}_2;t)-\left(\frac{z_{1,t}}{8\gamma_1}+\frac{z_{2,t}}{8\gamma_2}\right)\tau_t\right)\\
&=\sum_{t=1}^{T}\tau_t\ell_t-\sum_{t=1}^{T}\tau_t\ell(\mathbf{w}_1,\mathbf{w}_2;t)-\sum_{t=1}^{T}\left(\frac{z_{1,t}}{8\gamma_1}+\frac{z_{2,t}}{8\gamma_2}\right)\tau_t^2\\
&\geq\sum_{t=1}^{T}\tau_t\ell_t-\sum_{t=1}^{T}C\ell(\mathbf{w}_1,\mathbf{w}_2;t)-\sum_{t=1}^{T}\left(\frac{z_{1,t}}{8\gamma_1}+\frac{z_{2,t}}{8\gamma_2}\right)\tau_t\frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2+z_{2,t}\gamma_1}\\
&=\sum_{t=1}^{T}\tau_t\ell_t-C\sum_{t=1}^{T}\ell(\mathbf{w}_1,\mathbf{w}_2;t)-\frac{1}{2}\sum_{t=1}^{T}\tau_t\ell_t\\
&=\frac{1}{2}\sum_{t=1}^{T}\tau_t\ell_t-C\sum_{t=1}^{T}\ell(\mathbf{w}_1,\mathbf{w}_2;t)
\end{aligned}\tag{11}$$

Combining the above inequality with the conclusion of Lemma 1, we have

$$\frac{1}{2}\sum_{t=1}^{T}\tau_t\ell_t\leq\frac{\gamma_1}{2}\|\mathbf{v}-\mathbf{w}_1\|^2+\frac{\gamma_2}{2}\|\mathbf{w}_2\|^2+C\sum_{t=1}^{T}\ell(\mathbf{w}_1,\mathbf{w}_2;t)\tag{12}$$

Furthermore, when a mistake occurs, $\ell_t\geq 1$; thus, we have

$$\tau_t\ell_t=\min\left\{C,\frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2+z_{2,t}\gamma_1}\right\}\ell_t\geq\min\left\{C,\frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2+z_{2,t}\gamma_1}\right\}\geq\min\left\{C,\frac{4\gamma_1\gamma_2}{R_1\gamma_2+R_2\gamma_1}\right\}=\tau.$$

Combining the above observation with the inequality in Eq. (12), we have

$$\frac{1}{2}M\times\tau\leq\frac{\gamma_1}{2}\|\mathbf{v}-\mathbf{w}_1\|^2+\frac{\gamma_2}{2}\|\mathbf{w}_2\|^2+C\sum_{t=1}^{T}\ell(\mathbf{w}_1,\mathbf{w}_2;t)$$

The theorem follows directly by multiplying $2/\tau$ on both sides of the above inequality. □

Corollary 4. *Under the assumption in Theorem 3, if we further assume $R_1=R_2=1$ and $\gamma_1=\gamma_2$, we have the following bound for the HetOTL algorithm*

$$M\leq\frac{1}{\min\{C,1\}}\left[\frac{1}{2}\|\mathbf{v}-\mathbf{w}_1\|^2+\frac{1}{2}\|\mathbf{w}_2\|^2+2C\sum_{t=1}^{T}\ell(\mathbf{w}_1,\mathbf{w}_2;t)\right]$$

Proof. It is easy to verify that to minimize the left hand side of the inequality (10) is equivalent to (set $R_1=R_2=1$)

$$\min_{\gamma_1,\gamma_2,C>0}\frac{[\frac{\gamma_1}{\gamma_1+\gamma_1}\|\mathbf{v}-\mathbf{w}_1\|^2+\frac{\gamma_2}{\gamma_1+\gamma_2}\|\mathbf{w}_2\|^2+\frac{2C}{\gamma_1+\gamma_2}\sum_{t=1}^{T}\ell(\mathbf{w}_1,\mathbf{w}_2;t)]}{\min\{\frac{C}{\gamma_1+\gamma_2},\frac{4\gamma_1\gamma_2/(\gamma_+\gamma_2)^2}{\frac{\gamma_2}{\gamma_1+\gamma_2}+\frac{\gamma_1}{\gamma_1+\gamma_2}}\}}$$

which further is equivalent to

$$\min_{\gamma_1,\gamma_2,C>0,\gamma_1+\gamma_2=1}\frac{1}{\min\{C,4\gamma_1\gamma_2\}}\left[\gamma_1\|\mathbf{v}-\mathbf{w}_1\|^2+\gamma_2\|\mathbf{w}_2\|^2+2C\sum_{t=1}^{T}\ell(\mathbf{w}_1,\mathbf{w}_2;t)\right].$$

Plugging $\gamma_1=\gamma_2$ into the above inequality will result in the conclusion. □

4. Application of OTL for mining concept-drifting data streams

In this section, we apply the online transfer learning technique to attack the online learning task on concept-drifting data streams.

4.1. *Concept-Drifting Online Learning algorithm*

Consider a binary classification task in concept drift setting where a learner is presented with a sequence of data with time stamps. At time step t, the algorithm is provided with instance $\mathbf{x}_t \in \mathbb{R}^d$, and will predict its label as $\hat{y}_t = sign(\mathbf{w}_t^\top \mathbf{x}_t) \in \{-1, +1\}$, where $\mathbf{w}_t$ is the current prediction function. After prediction, the environment will disclose the real label y_t, so that the learner will suffer a loss $\ell((\mathbf{x}_t, y_t); \mathbf{w}_t)$, which is the difference between its prediction and the true label. Specifically, we will still adopt hinge loss $\ell((\mathbf{x}_t, y_t); \mathbf{w}_t) = \max(0, 1 - y_t \mathbf{w}_t^\top \mathbf{x}_t)$. After suffering from the loss, the leaner will update the prediction function using the current example with respect to some criterion. The overall objective of this learning process is to minimize the total mistake (or cumulative loss) over the entire sequence of examples. However, in the concept drifting setting, when the distribution changes frequently too much over time, traditional online algorithms will not work well.

Our main idea is that during the online learning process, we will divide the whole learning process into several periods. In each period, we will transfer the well learnt knowledge from the old classifier to a new one using the previous studied OTL technique. Specifically, the old classifier is the best one selected from the two classifiers in last period, and the new classifier is initialized as zero vector. As a result, if concept drift occurs, the newly learnt classifier may be adapted better than the older one; if no concept drift occurs, the old classifier will still perform well.

To formulate the above idea, we define a window size parameter P_i as the number of instances received in the i-th period. We maintain two classifiers: a source classifier $\mathbf{v}_t$ and a target classifier $\mathbf{w}_t$, which are weighted by $\alpha_{1,t}$ and $\alpha_{2,t}$, respectively. As a result, at the tth step, given an instance $\mathbf{x}_t$, we predict its class label by the following ensemble function:

$$\hat{y}_t = sign\left(\alpha_{1,t}\Pi\left(\mathbf{v}_t^\top \mathbf{x}_t\right) + \alpha_{2,t}\Pi\left(\mathbf{w}_t^\top \mathbf{x}_t\right) - \frac{1}{2}\right) \tag{13}$$

The key problem is how to effectively tune the weights. It is obvious that at the first period, source classifier is constantly zero function, so the source function is weighted with 0; while target function is weighted with 1 throughout it. To dynamically adjust the weights for the remaining steps, we use the following performance-driven exponential weighted updating scheme: when $\text{mod}(t, P_i) \neq 0$

$$\alpha_{1,t+1} = \frac{\alpha_{1,t} s_t(\mathbf{v}_t)}{\alpha_{1,t} s_t(\mathbf{v}_t) + \alpha_{2,t} s_t(\mathbf{w}_t)}, \qquad \alpha_{2,t+1} = \frac{\alpha_{2,t} s_t(\mathbf{w}_t)}{\alpha_{1,t} s_t(\mathbf{v}_t) + \alpha_{2,t} s_t(\mathbf{w}_t)} \tag{14}$$

where $s_t(\mathbf{u}) = \exp\{-\eta \ell^*(\Pi(\mathbf{u}^\top \mathbf{x}_t), \Pi(y_t))\}$, $\forall \mathbf{u} \in \mathbb{R}^m$ and $\ell^*(z, y) = (z - y)^2$ in our approach. Finally, Algorithm 4 summarizes the proposed Concept Drift Online Learning (CDOL) algorithm.

Algorithm 4 Concept Drift Online Learning (CDOL).

```
Initialize $\mathbf{v}_1 = \mathbf{0}$, $\mathbf{w}_1 = \mathbf{0}$, $\alpha_{1,1} = 0$ and $\alpha_{2,1} = 1$, and $i = 1$
for $t = 1, 2, \ldots, T$ do
    receive instance: $\mathbf{x}_t \in \mathcal{X}$
    predict $\hat{y}_t = sign(\alpha_{1,t}\Pi(\mathbf{v}_t^\top \mathbf{x}_t) + \alpha_{2,t}\Pi(\mathbf{w}_t^\top \mathbf{x}_t) - \frac{1}{2})$
    receive true label: $y_t \in \{-1, +1\}$
    suffer loss: $\ell_t = \max\{0, 1 - y_t \mathbf{w}_t^\top \mathbf{x}_t\}$
    if $\ell_t > 0$ then
        $\mathbf{w}_{t+1} = \mathbf{w}_t + \tau_t y_t \mathbf{x}_t$, where $\tau_t = \min\{C, \ell_t/\|\mathbf{x}_t\|^2\}$
    end if
    $\mathbf{v}_{t+1} = \mathbf{v}_t$
        $\alpha_{1,t+1} = \frac{\alpha_{1,t} s_t(\mathbf{v}_t)}{\alpha_{1,t} s_t(\mathbf{v}_t) + \alpha_{2,t} s_t(\mathbf{w}_t)}$,   $\alpha_{2,t+1} = 1 - \alpha_{1,t+1}$
    if $\text{mod}(t, P_i) = 0$ then
        $\mathbf{v}_{t+1} = \begin{cases} \mathbf{v}_{t+1} & \text{if } \alpha_{1,t+1} \geq \alpha_{2,t+1} \\ \mathbf{w}_{t+1} & \text{otherwise} \end{cases}$
        $\mathbf{w}_{t+1} = 0$ and $\alpha_{1,t+1} = \alpha_{2,t+1} = \frac{1}{2}$, and $i = i + 1$
    end if
end for
```

4.2. *Theoretical analysis*

Next we analyze the mistake bound of the algorithm. By Proposition 1, we derive the mistake bound of the CDOL algorithm as follows.

Theorem 5. *Assume the proposed CDOL algorithm is provided with a sequence of examples $\{(\mathbf{x}_t, y_t) \mid t = 1, 2, \ldots, T\}$, where $\|\mathbf{x}_t\| \leq R$, $y_t \in \{-1, +1\}$ and $T = \sum_{i=1}^{a} P_i$ (a is a positive integer). Let us denote by M the number of mistakes made by the CDOL algorithm, for any vector $\mathbf{u}$, we then have M bounded from above by:*

$$M \le \max\left\{R^2, \frac{1}{C}\right\}\left[\|\mathbf{u}\|^2 + 2C\sum_{t=1}^{P_1}\ell\big((\mathbf{x}_t, y_t); \mathbf{u}\big)\right] + 4\sum_{i=2}^{a}\min\{\Sigma_{\mathbf{v},i}, \Sigma_{\mathbf{w},i}\} + (8\ln 2)(a-1) \tag{15}$$

where $\Sigma_{\mathbf{v},i}$ and $\Sigma_{\mathbf{w},i}$ are the cumulative ℓ^* loss suffered by the source classifier and the target classifier, respectively during the i-th period.

Proof. Let us denote by M_i the number of mistakes made in period i, and M the total number of mistakes, which satisfies $M = \sum_{i=1}^{a} M_i$. For the first period, the algorithm runs the same with the PA-I algorithm. Thus, the mistake bound for this period is the same as that of PA-I, i.e.,

$$M_1 \le \max\left\{R^2, \frac{1}{C}\right\}\left[\|\mathbf{u}\|^2 + 2C\sum_{t=1}^{P_1}\ell\big((\mathbf{x}_t, y_t); \mathbf{u}\big)\right]. \tag{16}$$

After the first period, notice that whenever there is a mistake at some t-th step, we should have $(\alpha_{1,t}\Pi(\mathbf{v}_t^\top\mathbf{x}_t) + \alpha_{2,t}\Pi(\mathbf{w}_t^\top\mathbf{x}_t) - \Pi(y_t))^2 \ge \frac{1}{4}$. Thus, for $i = 2, \dots, a$, we have

$$\begin{aligned}&\sum_{t=\sum_{j=1}^{i-1}P_j}^{\sum_{j=1}^{i}P_j}\ell^*\big(\alpha_{1,t}\Pi\big(\mathbf{v}_t^\top\mathbf{x}_t\big) + \alpha_{2,t}\Pi\big(\mathbf{w}_t^\top\mathbf{x}_t\big), \Pi(y_t)\big)\\ &\quad= \sum_{t=\sum_{j=1}^{i-1}P_j}^{\sum_{j=1}^{i}P_j}\big(\alpha_{1,t}\Pi\big(\mathbf{v}_t^\top\mathbf{x}_t\big) + \alpha_{2,t}\Pi\big(\mathbf{w}_t^\top\mathbf{x}_t\big) - \Pi(y_t)\big)^2 \ge \frac{1}{4}M_i.\end{aligned}$$

Combining the above facts with Proposition 1, we have

$$\sum_{i=2}^{a} M_i \le 4\sum_{i=2}^{a}\min\{\Sigma_{\mathbf{v},i}, \Sigma_{\mathbf{w},i}\} + (a-1) * 8\ln 2 \tag{17}$$

The final mistake bound can be proved by combining Eq. (16) and Eq. (17). □

To better understand the above theorem, we will show a corollary. To do so, we will need the following proposition.

Proposition 3. *Denote $\ell_t^* = (\Pi(\mathbf{w}_t^\top\mathbf{x}_t) - \Pi(y_t))^2$ and $\ell_t = \max\{0, 1 - y_t\mathbf{w}_t^\top\mathbf{x}_t\}$, then the two losses satisfy the following inequality:*

$$\ell_t^* \le \min\left\{\frac{\ell_t}{2}, \frac{\ell_t^2}{4}\right\} \tag{18}$$

The proof to the above proposition can be found in Appendix A. Combining this proposition with Theorem 5, we have the following corollary:

Corollary 6. *Under the same assumption in Theorem 5, if further assume $R \le 1$ and $C \ge 2$, we have the following bound for the proposed CDOL algorithm in Algorithm 4*

$$M \le \sum_{i=1}^{a}\left\{\|\mathbf{u}_i\|^2 + \sum_{t=\sum_{j=1}^{i-1}P_j}^{\sum_{j=1}^{i}P_j} 2C\ell\big((\mathbf{x}_t, y_t); \mathbf{u}_i\big)\right\} + (8\ln 2)(a-1)$$

where M is the number of mistakes, and $\mathbf{u}_i, i = 1, 2, \dots, a$ are any a vectors which may or may not be the same.

Proof. According to Lemma 1 in [9], we have the following inequalities:

$$\sum_{t=\sum_{j=1}^{i-1}P_j}^{\sum_{j=1}^{i}P_j}\tau_t\big(2\ell_t - \tau_t\|\mathbf{x}_t\|^2 - 2\ell\big((\mathbf{x}_t, y_t); \mathbf{u}_i\big)\big) \le \|\mathbf{u}_i\|^2, \quad \forall i \in \{2, 3, \dots, a\}, \tag{19}$$

where $\mathbf{u}_i, i = 2, 3, \dots, a$ are any vectors which may or may not be the same. Because $\tau_t\|\mathbf{x}_t\|^2 = \min\{C, \ell_t/\|\mathbf{x}_t\|^2\}\|\mathbf{x}_t\|^2 \le \ell_t$ and $\tau_t \le C$, the inequality (19) implies

$$\sum_{t=\sum_{j=1}^{i-1} P_j}^{\sum_{j=1}^{i} P_j} \tau_t \ell_t \le \|\mathbf{u}_i\|^2 + \sum_{t=\sum_{j=1}^{i-1} P_j}^{\sum_{j=1}^{i} P_j} 2C\ell\big((\mathbf{x}_t, y_t); \mathbf{u}_i\big) \quad \forall i \in \{2, 3, \ldots, a\} \tag{20}$$

Furthermore, because $\|\mathbf{x}_t\| \le 1$ and $C \ge 2$, we have $\{2\ell_t, \ell_t^2\} \le \min\{C\ell_t, \frac{\ell_t^2}{\|\mathbf{x}_t\|^2}\}$

$$4\Sigma_{\mathbf{w},i} = 4 \sum_{t=\sum_{j=1}^{i-1} P_j}^{\sum_{j=1}^{i} P_j} \ell_t^* \le \min\{2\ell_t, \ell_t^2\} \le \sum_{t=\sum_{j=1}^{i-1} P_j}^{\sum_{j=1}^{i} P_j} \min\left\{C\ell_t, \frac{\ell_t^2}{\|\mathbf{x}_t\|^2}\right\} = \sum_{t=\sum_{j=1}^{i-1} P_j}^{\sum_{j=1}^{i} P_j} \tau_t \ell_t, \tag{21}$$

where $i \in \{2, 3, \ldots, a\}$. Combining the above inequalities (20) and (21), we have

$$4\sum_{i=2}^{a} \min\{\Sigma_{\mathbf{v},i}, \Sigma_{\mathbf{w},i}\} \le \sum_{i=2}^{a} 4\Sigma_{\mathbf{w},i} \le \sum_{i=2}^{a} \left\{\|\mathbf{u}_i\|^2 + \sum_{t=\sum_{j=1}^{i-1} P_j}^{\sum_{j=1}^{i} P_j} 2C\ell\big((\mathbf{x}_t, y_t); \mathbf{u}_i\big)\right\}$$

Finally, combining the above inequality with Theorem 5 results in

$$M \le \sum_{i=1}^{a} \left\{\|\mathbf{u}_i\|^2 + \sum_{t=\sum_{j=1}^{i-1} P_j}^{\sum_{j=1}^{i} P_j} 2C\ell\big((\mathbf{x}_t, y_t); \mathbf{u}_i\big)\right\} + (8\ln 2)(a-1). \qquad \square$$

Although the above theorem offers a nice theoretical guarantee of Algorithm 4, its empirical performance could be affected by the selection of the window size parameters P_i at different periods. One simple way is to fix all P_i values to a proper constant P, which ideally should match the concept drift cycle. Such an approach is practically infeasible because (i) finding a proper parameter P is hard since the optimal window size for concept drift can only be known in hindsight; and (ii) concept drift often occurs irregularly, which would make a single windows size parameter fail in practice. To overcome the challenge of parameter selection for P_i, in this paper, we propose an automated parameter selection technique, an Online Window Adjustment (OWA) algorithm as shown in Algorithm 5, which can automatically determine a proper value for window size parameter P_i during the online learning process. We note that this algorithm was inspired by the existing Window Adjustment algorithm [22] used for solving batch concept drift tasks.

Algorithm 5 Online Window Adjustment Algorithm (OWA).

```
Input small window size $P$ and trade-off $C$
Initialize $\mathbf{u}_{j,1} = \mathbf{0}$, $M_j = 0$, where $j = 1, 2$ and $P_1 = P$, $i = 1$
for $t = 1, 2, \ldots, T$ do
  receive instance: $\mathbf{x}_t \in \mathcal{X}$
  predict $\hat{y}_{j,t} = sign(\mathbf{u}_{j,t}^\top \mathbf{x}_t)$, where $j = 1, 2$
  receive true label: $y_t \in \{-1, +1\}$
  compute $M_j = M_j + \mathbb{I}_{(\hat{y}_{j,t} \ne y_t)}$, where $\forall j = 1, 2$
  suffer loss: $\ell_{j,t} = \max\{0, 1 - y_t \mathbf{u}_{j,t}^\top \mathbf{x}_t\}$, where $j = 1, 2$
  $\mathbf{u}_{j,t+1} = \mathbf{u}_{j,t} + \tau_{j,t} y_t \mathbf{x}_t$, where $\tau_{j,t} = \min\{C, \ell_{j,t}/\|\mathbf{x}_t\|^2\}$, $j = 1, 2$
  if mod$(t, P) = 0$ then
    if $M_1 > M_2$ then
      $i = i + 1$, $P_i = P$
    else
      $P_i = P_i + P$
    end if
    $\mathbf{u}_{1,t+1} = \mathbf{u}_{2,t+1}$ $\mathbf{u}_{2,t+1} = \mathbf{0}$, $M_j = 0$, $j = 1, 2$
  end if
end for
```

5. Experimental results

In this section, we evaluate the empirical performance of the proposed OTL technique for three sets of experiments: (i) homogeneous OTL for classification tasks, (ii) heterogeneous OTL for classification tasks, and (III) application of OTL for concept-drifting online learning tasks. The whole experimental testbed including all the datasets and source code are available at http://www.stevenhoi.org/OTL/.

Table 1
Datasets used in the homogeneous OTL classification tasks.

Dataset	# examples	# features	# source instances
books-dvd	4000	473,857	2000
dvd-books	4000	473,857	2000
electronics-kitchen	4000	473,857	2000
kitchen-electronics	4000	473,857	2000
landmine1	14,820	9	8535
landmine2	14,820	9	6285

5.1. *Experiment I: homogeneous OTL for classification tasks*

5.1.1. *Experimental testbed and setup*

Our first experiment is to evaluate the performance of HomOTL from homogeneous data. We compare our HomOTL technique against other popular online learning techniques, including the Passive–Aggressive algorithms ("PA") [9] without exploiting any knowledge from the source domain, and a variant of it, which is the **PA** method **I**nitialized with the **O**ld classifier **v**, denoted as **PAIO** for short. For our HomOTL technique, in addition to Algorithms 1 and 2, we also implement another variant, which is implemented by fixing the ensemble weights of the HomOTL-I algorithm to 1/2, denoted "HomOTL(fixed)" for short. This helps examine the efficacy of the proposed weighting strategy. We evaluate the proposed algorithms on six benchmark datasets for transfer learning as listed in Table 1.

These datasets were created based on the "sentiment" and "landmine" datasets downloaded from the website,[1] which are popularly used to benchmark transfer learning algorithms. The first four datasets are named in the form of "name1–name2", which means data "name1", one domain from "sentiment", is used as training data in the source domain, and data "name2", another domain from "sentiment", is treated as test data for online learning in the target domain. The last two datasets were created from "landmine", which consists of 19 tasks, where 1–10 were collected at foliated regions and 11–19 were collected at regions that are bare earth or desert. Thus, "landmine1" uses 1–10 as the source data and the rest as target data; while "landmine2" uses "11–19" as source data and the rest as target data.

Finally, we adopt the PA algorithm to run on the source dataset and adopt the average classifier as the source classifier, which generally enjoys better generalization ability [7]. Further, we draw 20 times of random permutation of the instances in the target domain in order to obtain stable results by averaging over the 20 trials. All the algorithms adopt the kernel-based implementation with the same Gaussian kernel function. For fair comparison and simplicity, we set the kernel parameter $\sigma_1 = 4$ for the source domain and $\sigma_2 = 8$ for the target domain for all the datasets and algorithms. In addition, we set the regularization parameter $C = 5$ for all algorithms, and parameter $\beta = \sqrt{T}/(\sqrt{T} + \sqrt{\ln 2})$ for HomOTL-II. We will also conduct experiments to examine the parameter sensitivity in subsequent sections. For performance evaluation, we evaluate the predictive accuracy of online learning methods by measuring the standard mistake rate, the sparsity of the resulting classifiers by evaluating the number of support vectors, and time efficiency by calculating average time costs.

5.1.2. *Performance evaluation results*

Table 2 summarizes the performance of the compared algorithms. Several observations can be drawn from the experimental results. First of all, for most of the datasets, we found that the PA algorithm performs the worst, which implies the necessity of studying online transfer learning. Secondly, PAIO achieves better performance than PA on the first four datasets, while has no much improvement on the final two datasets, which demonstrates the importance of developing more sophisticated algorithms. In addition, the proposed HomOTL-I and HomOTL-II algorithms achieve the best performance among all datasets, which implies that the exploiting learnt knowledge from source domain is able to boost the performance of traditional online learning algorithms, and the two kinds of weight updating methods are generally comparable. Furthermore, we found that HomOTL-I outperforms HomOTL(fixed) on all the datasets, which shows that the proposed weight updating strategy can effectively transfer the knowledge. Finally, for the time efficiency evaluation, the proposed two OTL strategies are generally comparable to PAIO, and PA is the most efficient because it does not exploit data in the target domain.

Fig. 1 also shows the details of average mistake rates varying over the learning processes on the six data sets, respectively. Similar observations show that the two OTL algorithms achieve the best performance after receiving a small number of examples (e.g., less than 100 examples), which implies these two strategies can efficiently transfer the well-learnt knowledge from the source task to the target task. This again verifies the high learning efficacy of the proposed methodology.

5.1.3. *Sensitivity evaluation of parameter C for homogeneous OTL*

Fig. 2 evaluates the online prediction performance of the compared algorithms with varied C values across all the homogeneous learning tasks. Several observations can be drawn from the results. First of all, it is clear that the proposed

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

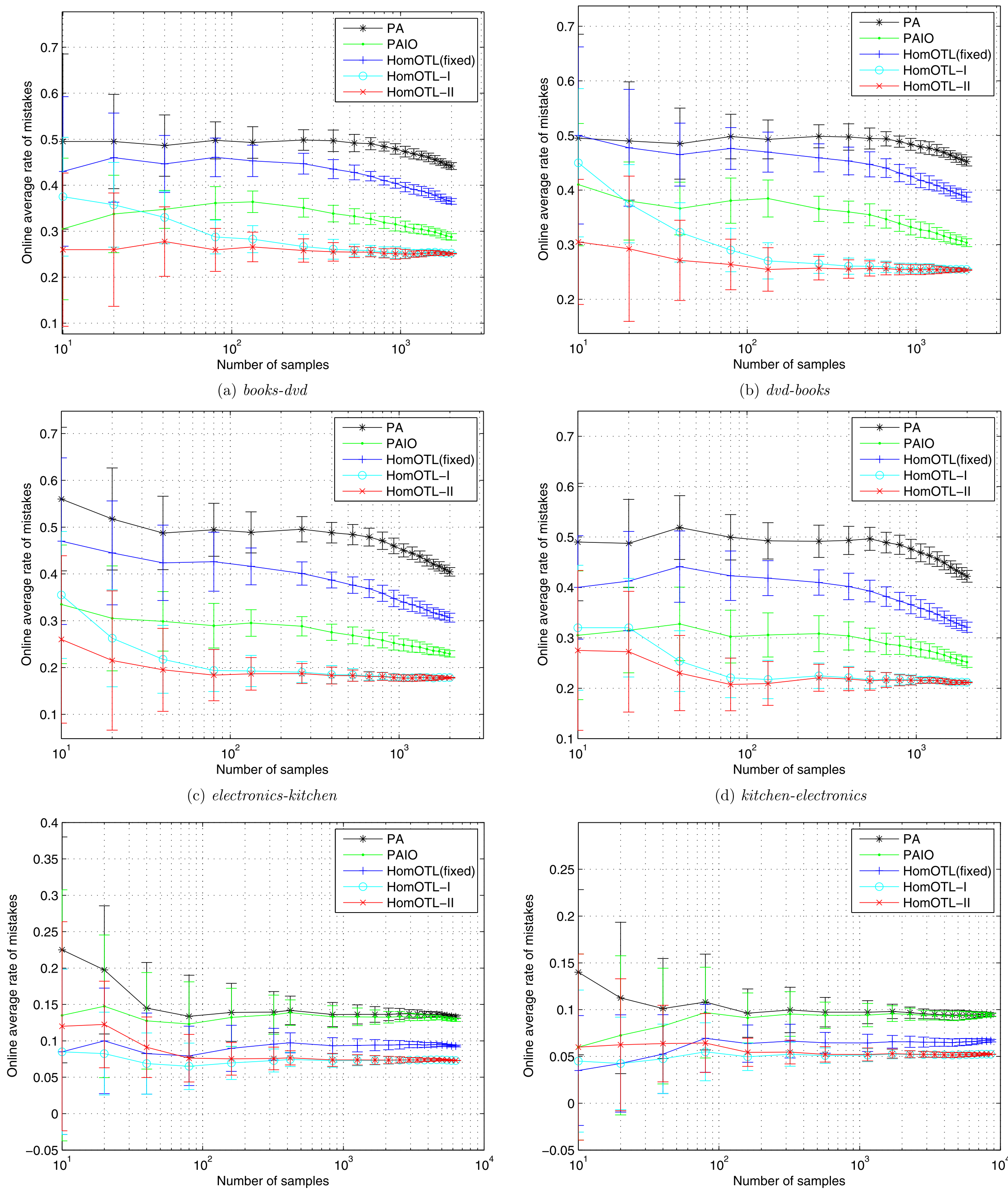
Table 2
Results on the datasets of homogeneous domain for classification.

Algorithm	books-dvd		
	Mistake (%)	Support vectors (#)	Time (s)
PA	44.1475 ± 0.7696	1626.5500 ± 20.9271	0.0519 ± 0.0018
PAIO	28.7625 ± 0.6825	3446.2500 ± 9.6729	0.1386 ± 0.0021
HomOTL(fixed)	36.4750 ± 0.6294	3384.5500 ± 20.9271	0.1318 ± 0.0011
HomOTL-I	**25.2325** ± **0.1029**	3384.5500 ± 20.9271	0.1417 ± 0.0009
HomOTL-II	**25.1200** ± **0.0377**	3384.5500 ± 20.9271	0.1366 ± 0.0011
Algorithm	dvd-books		
	Mistake (%)	Support vectors (#)	Time (s)
PA	45.2050 ± 0.8041	1633.1000 ± 15.1446	0.0522 ± 0.0008
PAIO	30.3525 ± 0.7192	3470.8500 ± 14.7444	0.1391 ± 0.0012
HomOTL(fixed)	38.6975 ± 0.8973	3400.1000 ± 15.1446	0.1339 ± 0.0012
HomOTL-I	**25.4400** ± **0.1165**	3400.1000 ± 15.1446	0.1434 ± 0.0014
HomOTL-II	**25.3350** ± **0.0432**	3400.1000 ± 15.1446	0.1379 ± 0.0014
Algorithm	electronics-kitchen		
	Mistake (%)	Support vectors (#)	Time (s)
PA	40.4200 ± 0.8904	1552.9000 ± 17.4865	0.0505 ± 0.0007
PAIO	22.8950 ± 0.6770	3106.4000 ± 13.2005	0.1232 ± 0.0015
HomOTL(fixed)	30.6200 ± 0.9379	3173.9000 ± 17.4865	0.1261 ± 0.0016
HomOTL-I	**17.8350** ± **0.0860**	3173.9000 ± 17.4865	0.1369 ± 0.0024
HomOTL-II	**17.7600** ± **0.0205**	3173.9000 ± 17.4865	0.1313 ± 0.0012
Algorithm	kitchen-electronics		
	Mistake (%)	Support vectors (#)	Time (s)
PA	42.2100 ± 1.1458	1564.9000 ± 21.3810	0.0520 ± 0.0023
PAIO	25.1750 ± 1.0392	3123.8500 ± 20.8561	0.1297 ± 0.0074
HomOTL(fixed)	32.1075 ± 1.0058	3187.9000 ± 21.3810	0.1308 ± 0.0041
HomOTL-I	**21.2025** ± **0.0980**	3187.9000 ± 21.3810	0.1409 ± 0.0048
HomOTL-II	**21.1175** ± **0.0467**	3187.9000 ± 21.3810	0.1362 ± 0.0074
Algorithm	landmine1		
	Mistake (%)	Support vectors (#)	Time (s)
PA	13.3166 ± 0.2064	1676.5500 ± 31.9003	0.1620 ± 0.0067
PAIO	12.8767 ± 0.2171	3396.6500 ± 28.9233	0.4695 ± 0.0680
HomOTL(fixed)	9.2912 ± 0.1329	3356.5500 ± 31.9003	0.4344 ± 0.0138
HomOTL-I	**7.2888** ± **0.0049**	3356.5500 ± 31.9003	0.4686 ± 0.0144
HomOTL-II	**7.2880** ± **0.0036**	3356.5500 ± 31.9003	0.4524 ± 0.0108
Algorithm	landmine2		
	Mistake (%)	Support vectors (#)	Time (s)
PA	9.4599 ± 0.1709	1713.0000 ± 32.1395	0.2321 ± 0.0094
PAIO	9.4206 ± 0.1684	3378.8000 ± 30.0063	0.6755 ± 0.0942
HomOTL(fixed)	6.6837 ± 0.1350	3420.0000 ± 32.1395	0.6187 ± 0.0140
HomOTL-I	**5.2296** ± **0.0069**	3420.0000 ± 32.1395	0.6667 ± 0.0154
HomOTL-II	**5.2267** ± **0.0036**	3420.0000 ± 32.1395	0.6476 ± 0.0222

two online transfer learning algorithms are significantly more effective than the other algorithms for most cases. Second, among all the compared algorithms, we observe that the proposed HomOTL-I and HomOTL-II algorithms always achieve the best performance when C is sufficiently large (e.g. $C > 4$), which indicates a large learning rate can efficiently improve the transfer learning efficiency. Third, we observe that HomOTL-I and HomOTL-II are significantly more accurate than the other two transfer learning strategies: HomOTL(fixed) and PAIO under varied C values, which indicates the proposed algorithms are more effective for online transfer learning. Fourth, while the insensitivity of HomOTL methods to the value of C on landmine datasets indicates that the dynamic weighting strategies are very effective for these datasets, HomOTL methods improve performance on these datasets only if a suboptimal value of C is chosen. Finally, the PA algorithm performs the worst on all the datasets for varied C values as it does not exploit the knowledge from the source domain.

5.1.4. Sensitivity evaluation of parameter β for the HomOTL-II algorithm

In the previous experiments, we fix the value of parameter β to $\sqrt{T}/(\sqrt{T}+\sqrt{\ln 2})$ for the proposed HomOTL-II algorithm. One concern is whether if this algorithm is sensitive to the value of parameter β. Table 3 evaluates the online prediction



(a) *books-dvd* (b) *dvd-books* (c) *electronics-kitchen* (d) *kitchen-electronics* (e) *landmine1* (f) *landmine2*

Fig. 1. Evaluation of online mistake rates on homogeneous OTL classification tasks.

mistake rates of the HomOTL-II algorithm with varied values of β on six different homogeneous OTL tasks. From the results, we observe that the performance of the HomOTL-II algorithm is in general insensitive to the parameter β where HomOTL-II always outperforms PAIO consistently for all settings, validating the advantage of the proposed algorithm.

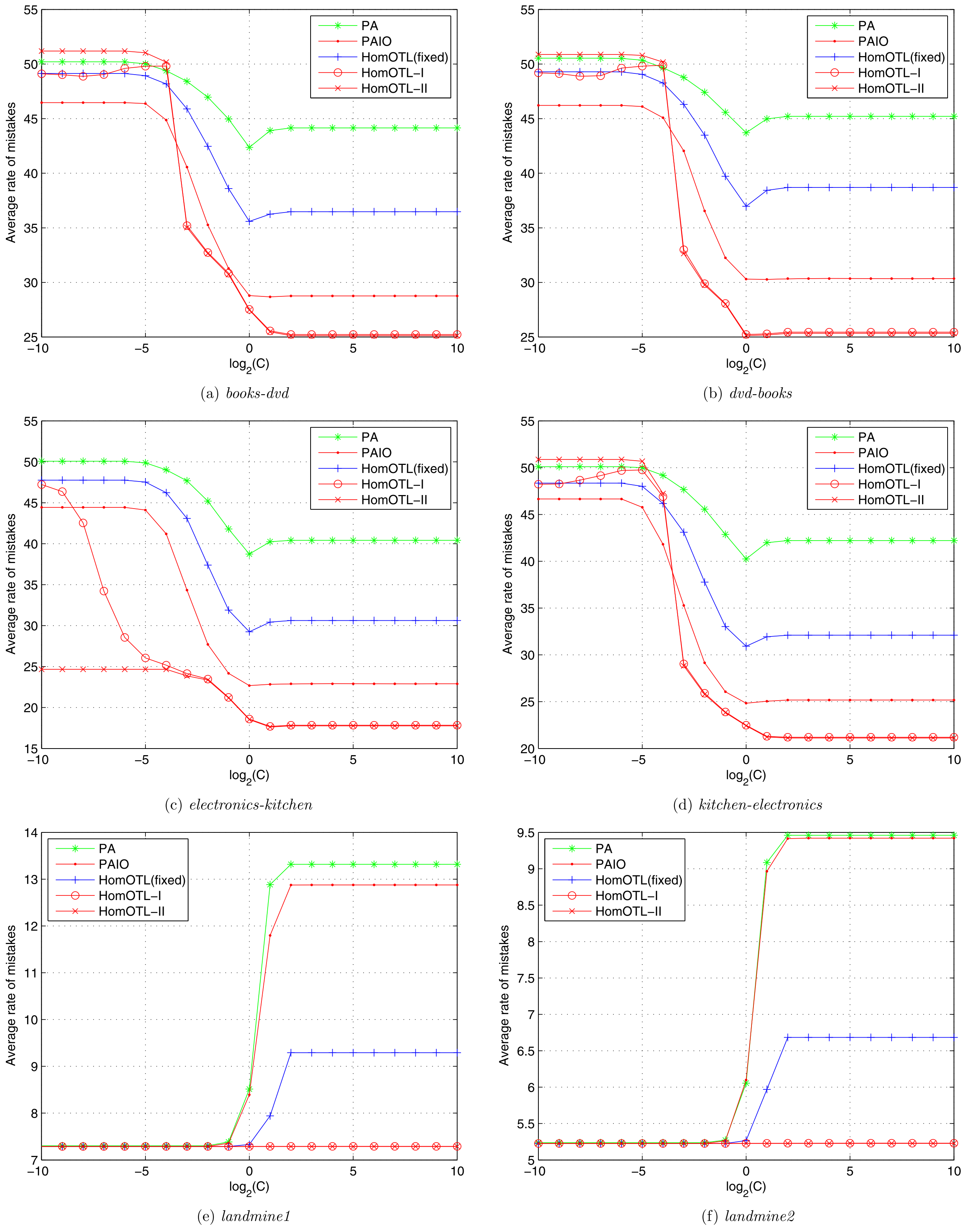


Fig. 2. Evaluation on homogeneous OTL classification tasks with varied C values.

Table 3
Evaluation of the HomOTL-II algorithm under varied values of parameter β.

Dataset	PAIO	HomOTL-II				
		$\beta = 0.125$	$\beta = 0.25$	$\beta = 0.5$	$\beta = 0.75$	$\beta = 0.875$
books-dvd	28.7625	25.1200	25.1200	25.1200	25.1225	25.1150
dvd-books	30.3525	25.3425	25.3425	25.3425	25.3375	25.3325
electronics-kitchen	22.8950	17.7700	17.7700	17.7700	17.7700	17.7700
kitchen-electronics	25.1750	21.1200	21.1200	21.1200	21.1200	21.1200
landmine1	12.8767	7.2880	7.2880	7.2880	7.2880	7.2880
landmine2	9.4206	5.2267	5.2267	5.2267	5.2267	5.2267

Table 4
Summary of data sets used for heterogeneous OTL classification tasks.

Datasets	Source/old domain		Target/new domain	
	Number	Dimension	Number	Dimension
books-dvd	2000	236,928	2000	473,857
dvd-books	2000	236,928	2000	473,857
electronics-kitchen	2000	236,928	2000	473,857
kitchen-electronics	2000	236,928	2000	473,857
landmine1	8535	5	6285	9
landmine2	6285	5	8535	9

5.2. *Experiment II: heterogeneous OTL for classification tasks*

5.2.1. *Experimental testbed and setup*

We now evaluate the empirical performance of the proposed HetOTL algorithm for heterogeneous OTL on classification tasks. We compare HetOTL with the PA algorithm, which does not exploit knowledge from the source domain. Similarly, we implement a variant of PA algorithm that uses only the first view of the data and is initialized with $\mathbf{v}$ from the source domain, denoted as “PAIO”. We also implement a variant of HetOTL, whose first-view classifier is initialized with a zero function, denoted as “HetOTL0”. This method enables us to examine the importance of engaging the $\mathbf{v}$ function learnt from the source domain. Finally, we implement another baseline algorithm that simply use HomOTL, where the source classifier only consider the first view, denoted as “Ensemble” for short. We evaluate all the algorithms extensively on several benchmark datasets, as shown in Table 4.

These datasets are the same as those used in previous homogeneous classification tasks. However, to meet the assumption and setup of heterogeneous OTL tasks, the source-domain data associate with only half of the feature space while the target-domain data include the whole feature space.

All the algorithms adopt the same Gaussian kernel. For fair comparison and simplicity, for all the datasets and algorithms, we set the regularization parameter $\gamma_1 = \gamma_2 = 1$ and kernel parameter $\sigma_1 = \sigma_2 = 4$ for the two views and $\sigma = 8$ for the whole feature. In addition, the regularization parameter C is set to 5 for all the algorithms on all the datasets. We conducted 20 runs of random permutations for each dataset and measured the average results of these 20 runs. In particular, we evaluate the performance of online learning methods by calculating the mistake rates, and evaluate the number of SVs and the time cost of the compared algorithms for efficiency evaluation.

5.2.2. *Performance evaluation results*

Table 5 summarizes the evaluation results of heterogeneous OTL tasks.

Several observations can be drawn from the above results. First of all, we found that among all the algorithms, the PA algorithm without exploiting knowledge from source domain achieved very high mistake rate in most cases. This shows that it is important for studying knowledge transfer in an OTL task. Second, for all the datasets, we found that the HetOTL algorithm has the smallest mistake rate. This validates the proposed OTL technique is effective for knowledge transfer in the online learning tasks. By examining the running time cost, we found that the HetOTL techniques usually consumes comparable time with the other baselines except the PA algorithm, which is actually resulted in by the number of SVs stored by each algorithm. This clearly demonstrates the efficiency of the proposed HetOTL technique. Finally, Fig. 3 shows the details of the HetOTL processes. Firstly, the standard deviations are high for most of the algorithms, which reduces the significance of the improvement of the proposed algorithm. Secondly, similar observations on the average mistakes from the results, in a way, verify the proposed HetOTL method is effective for the challenging heterogeneous OTL tasks.

5.2.3. *Sensitivity evaluation of parameter C for heterogeneous OTL*

Fig. 4 evaluates the online prediction performance of different algorithms with varied values of parameter C across all the heterogeneous learning tasks. Several observations can be drawn from the results. First of all, it is clear to observe that the proposed HetOTL algorithm is significantly more effective than the other algorithms for most cases. Second, among all the compared algorithms, we observe that the proposed HetOTL algorithm always achieves the best performance (or at least

Table 5
Results on the datasets of heterogeneous domain for classification.

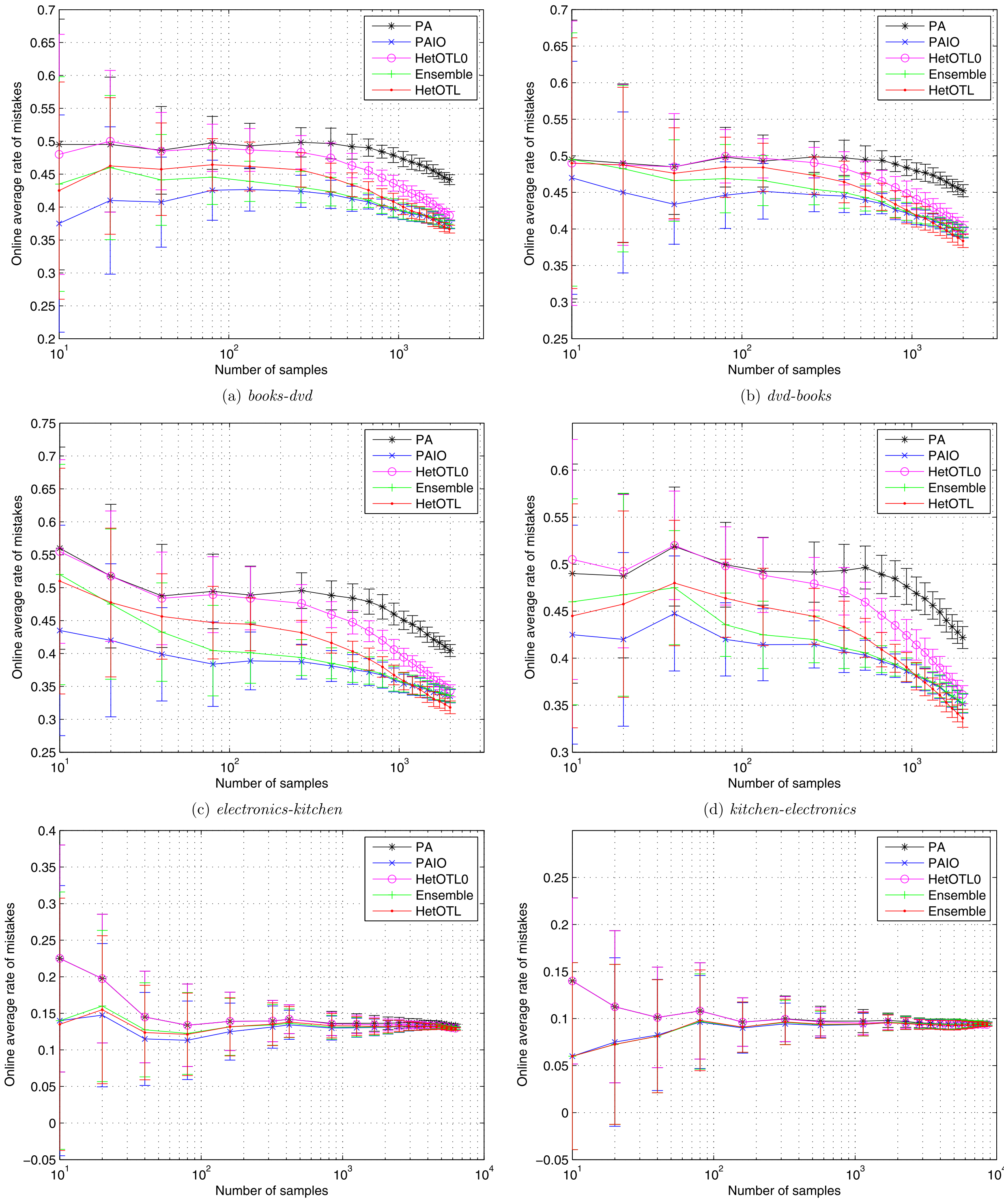
Algorithm	books-dvd		
	Mistake (%)	Support vectors (#)	Time (s)
PA	44.1475 ± 0.7696	1626.5500 ± 20.9271	0.0524 ± 0.0008
PAIO	37.2975 ± 0.5961	3318.2500 ± 20.5782	0.1356 ± 0.0064
HetOTL0	38.6775 ± 0.6463	3372.5000 ± 30.5795	0.1037 ± 0.0022
Ensemble	37.3800 ± 0.5921	4944.8000 ± 37.1251	0.2037 ± 0.0026
HetOTL	**36.6250** ± **0.6142**	5003.0000 ± 31.7788	0.1936 ± 0.0017
Algorithm	dvd-books		
	Mistake (%)	Support vectors (#)	Time (s)
PA	45.2050 ± 0.8041	1633.1000 ± 15.1446	0.0539 ± 0.0011
PAIO	39.4525 ± 0.6862	3395.0500 ± 14.8234	0.1346 ± 0.0028
HetOTL0	40.1425 ± 0.8027	3398.9000 ± 26.6476	0.1057 ± 0.0033
Ensemble	39.5500 ± 0.6827	5028.1500 ± 26.7685	0.2139 ± 0.0305
HetOTL	**38.3400** ± **0.8879**	5075.3000 ± 30.8496	0.1951 ± 0.0051
Algorithm	electronics-kitchen		
	Mistake (%)	Support vectors (#)	Time (s)
PA	40.4200 ± 0.8904	1552.9000 ± 17.4865	0.0525 ± 0.0022
PAIO	33.4925 ± 1.0033	3062.9000 ± 15.7577	0.1213 ± 0.0030
HetOTL0	34.2825 ± 0.8917	3141.7000 ± 27.5855	0.0994 ± 0.0015
Ensemble	33.5750 ± 1.0213	4615.8000 ± 29.6197	0.1892 ± 0.0025
HetOTL	**31.7775** ± **0.9496**	4612.2000 ± 27.6702	0.1824 ± 0.0189
Algorithm	kitchen-electronics		
	Mistake (%)	Support vectors (#)	Time (s)
PA	42.2100 ± 1.1458	1564.9000 ± 21.3810	0.0519 ± 0.0018
PAIO	35.1925 ± 1.0015	3150.2000 ± 32.0700	0.1250 ± 0.0039
HetOTL0	36.1275 ± 0.9450	3165.7000 ± 32.8459	0.1006 ± 0.0020
Ensemble	35.2625 ± 1.0089	4715.1000 ± 49.3813	0.1946 ± 0.0058
HetOTL	**33.6325** ± **0.9501**	4689.0000 ± 39.1677	0.1819 ± 0.0028
Algorithm	landmine1		
	Mistake (%)	Support vectors (#)	Time (s)
PA	13.3166 ± 0.2064	1676.5500 ± 31.9003	0.1659 ± 0.0076
PAIO	12.9737 ± 0.1896	3431.0000 ± 28.3382	0.4856 ± 0.0680
HetOTL0	12.9626 ± 0.1733	3402.8000 ± 63.2535	0.3205 ± 0.0151
Ensemble	12.9881 ± 0.1902	5107.5500 ± 55.1032	0.7076 ± 0.0522
HetOTL	**12.8361** ± **0.1897**	5150.8000 ± 59.3079	0.6689 ± 0.0518
Algorithm	landmine2		
	Mistake (%)	Support vectors (#)	Time (s)
PA	9.4599 ± 0.1709	1713.0000 ± 32.1395	0.2242 ± 0.0078
PAIO	9.5636 ± 0.2040	3422.9000 ± 31.6625	0.6121 ± 0.0446
HetOTL0	9.3538 ± 0.1840	3355.0000 ± 57.1996	0.4205 ± 0.0149
Ensemble	9.4487 ± 0.2118	5135.9000 ± 59.5950	0.9200 ± 0.0593
HetOTL	**9.3146** ± **0.1813**	5032.0000 ± 62.4112	0.8312 ± 0.0191

very close to the best performance, e.g., “landmine1” and “landmine2”) when C is sufficiently large (e.g., $C > 4$). This shows that setting a large learning rate can improve the transfer learning efficacy. Third, we observe that HetOTL is significantly more accurate than the other transfer learning strategies under varied C values, which again validates the efficacy of the proposed OTL strategy. Fourth, HomOTL improves performance on the landmine datasets only if a suboptimal value of C is chosen. Finally, the PA algorithm performs the worst on all the datasets under varied C values.

5.3. Experiment III: OTL for learning with concept-drifting data streams

5.3.1. Experimental testbed and setup

We now evaluate the empirical performance of the proposed technique for online learning tasks with concept-drifting data streams. We compare our CDOL algorithm with the standard Perceptron (denoted as “PE”) and PA algorithms. In addition, we also compare with two popular algorithms for concept-drifting online learning, i.e., the Modified Perceptron algorithm (denoted as “ModiPE” for short) [10], and the shifting Perceptron algorithm (denoted as “ShiftPE” for short) [6]. In addition, we also implement a variant of the CDOL which will set $P_i = P\ \forall i$, and term it as CDOL(fixed).



(a) *books-dvd* (b) *dvd-books*

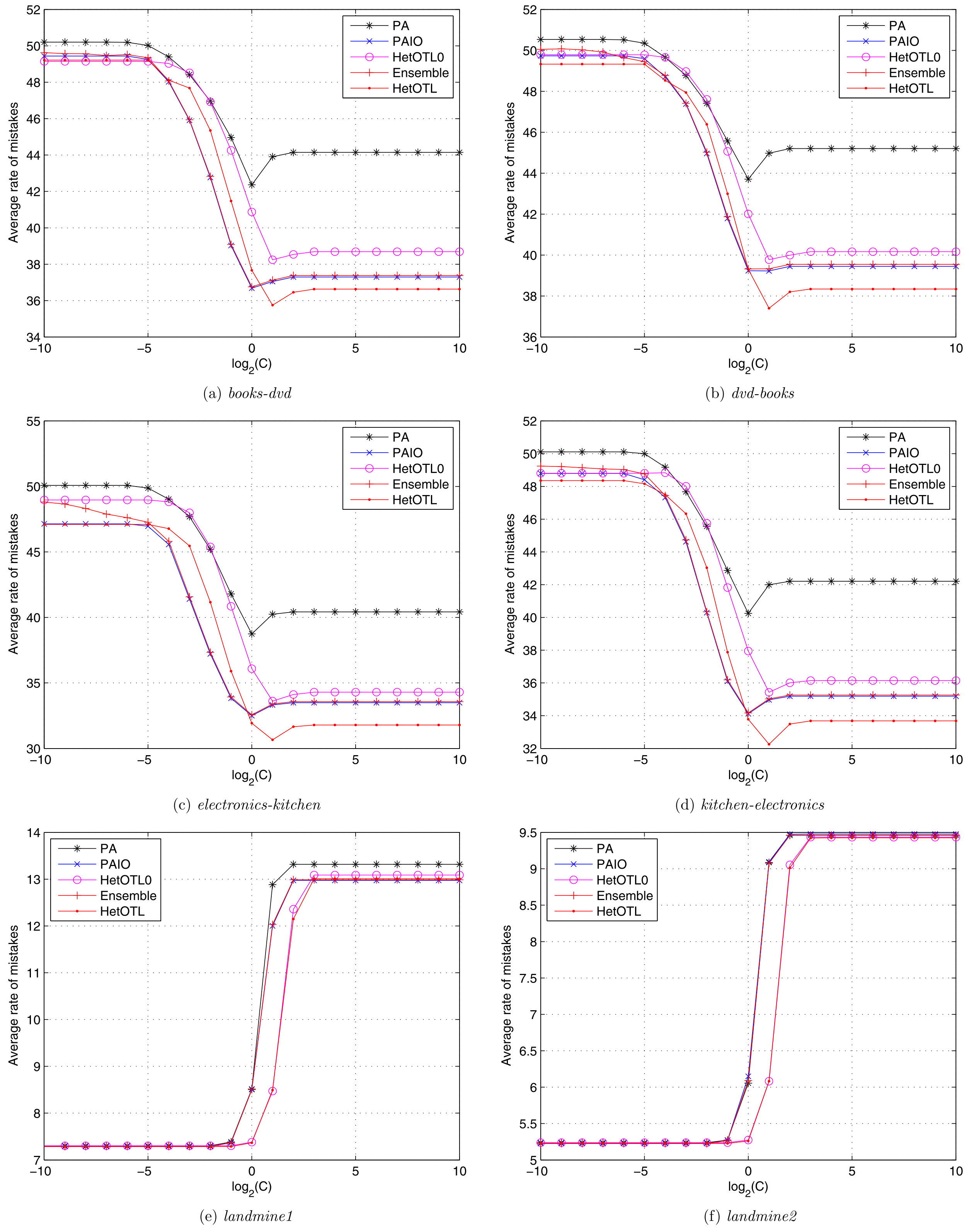
(c) *electronics-kitchen* (d) *kitchen-electronics*

(e) *landmine1* (f) *landmine2*

Fig. 3. Evaluation of online mistake rates on heterogeneous OTL classification tasks.

We evaluate the performance of all the algorithms on six benchmark datasets, as shown in Table 6, where the datasets “emaildata”, “usenet1” and “usenet2”, are downloaded from Concept Drift Datasets website,[2] while the “MITface”, “news-

[2] http://mlkd.csd.auth.gr/concept_drift.html.



(a) *books-dvd* (b) *dvd-books* (c) *electronics-kitchen* (d) *kitchen-electronics* (e) *landmine1* (f) *landmine2*

Fig. 4. Evaluation on heterogeneous OTL classification tasks with varied C values.

Table 6
Datasets used in the concept drift tasks.

Dataset	emaildata	MITface	newsgroup4	usenet1	usenet2	usps
# examples	1500	6977	1600	1500	1500	2930
# features	913	361	62,061	99	99	256

Table 7
The class distribution of dataset *MITface*.

Example id	1–1500	1501–3000	3001–4500	4501–6000	6000–6977
face	+	−	+	−	+
non-face	−	+	−	+	−

Table 8
The class distribution of dataset *newsgroup4*.

Example id	0–400	401–800	801–1200	1201–1600
comp.windows.x	+	−	−	+
rec.sport.hockey	+	+	−	−
sci.space	−	+	+	−
talk.politics.mideast	−	−	+	+

Table 9
The class distribution of dataset *usps*.

Original \ changed label	1–400	401–1200	1201–2400	2401–2930
1	+	−	+	−
2, 3	−	+	−	+

group4" and "usps" are created by ourself by using the datasets downloaded from MIT website[3] and the LIBSVM website, respectively. The details of "MITface", "newsgroup4" and "usps" are shown in Tables 7, 8 and 9, respectively.

All the algorithms employ the same Gaussian kernel, where the kernel width is set as $\sigma = 8$. Similarly, for fair comparison, we set the parameter C to 5 for all the algorithms on all the datasets. In addition, parameter λ is set to 1 for the Shift Perceptron algorithm, and parameter $P = 30$ is used for CDOL(fixed) and OWA, which is used to determine P_i for CDOL automatically. We conducted 20 different runs of random permutations (the examples will be permutated within every period) to obtain the average results. We evaluate the accuracy of online learning algorithms by measuring the mistake rate, their model sparsity by measuring the number of SVs, and their efficiency by measuring time cost.

5.3.2. *Performance evaluation results*

Table 10 summarizes the results for concept-drift online learning.

Several observations can be drawn from the results. First of all, we found that for the first two algorithms without considering concept drift, the PA algorithm achieved better performance for most cases. This shows that PA can learn new knowledge more effectively than the passive PE algorithm. Second, among all the algorithms, the ModiPE and ShiftPE algorithms designed for learning with concept drift seldom outperform the simple PA algorithm, which indicates that concept drift learning is in general hard to solve and the existing techniques are still not effective enough. In addition, CDOL almost always outperforms CDOL(fixed), which implies that the proposed OWA algorithm could find proper P_is effectively. Fourth, among all the evaluated algorithms, we found that the proposed CDOL algorithm achieved the smallest mistake rates for most of the datasets. This validates CDOL is effective for knowledge transfer in the concept-drift learning tasks. Of course, there is some cost for performing knowledge transfer for the gain achieved by the proposed CDOL method. By examining the running time cost, we found that CDOL usually took more time cost than the other algorithms, since it need take time to find the best P_is. Finally, Fig. 5 shows the details of the concept-drift online learning processes. Similar observations can be found from the results, which again verify the proposed OTL technique is effective and promising for resolving the challenging tasks of online learning with concept drift.

5.3.3. *Sensitivity evaluation of parameter C for concept-drifting learning*

Fig. 6 examines the online prediction performance of different algorithms with varied values of C for concept drifting online learning tasks. Several observations can be drawn from the results. First of all, it is clear to see that the proposed CDOL algorithm is considerably more effective than the other algorithms for most cases. Second, among all the compared algorithms, we observe that the proposed CDOL algorithm often achieves the best performance when C is sufficiently large

[3] http://cbcl.mit.edu/software-datasets/FaceData2.html.

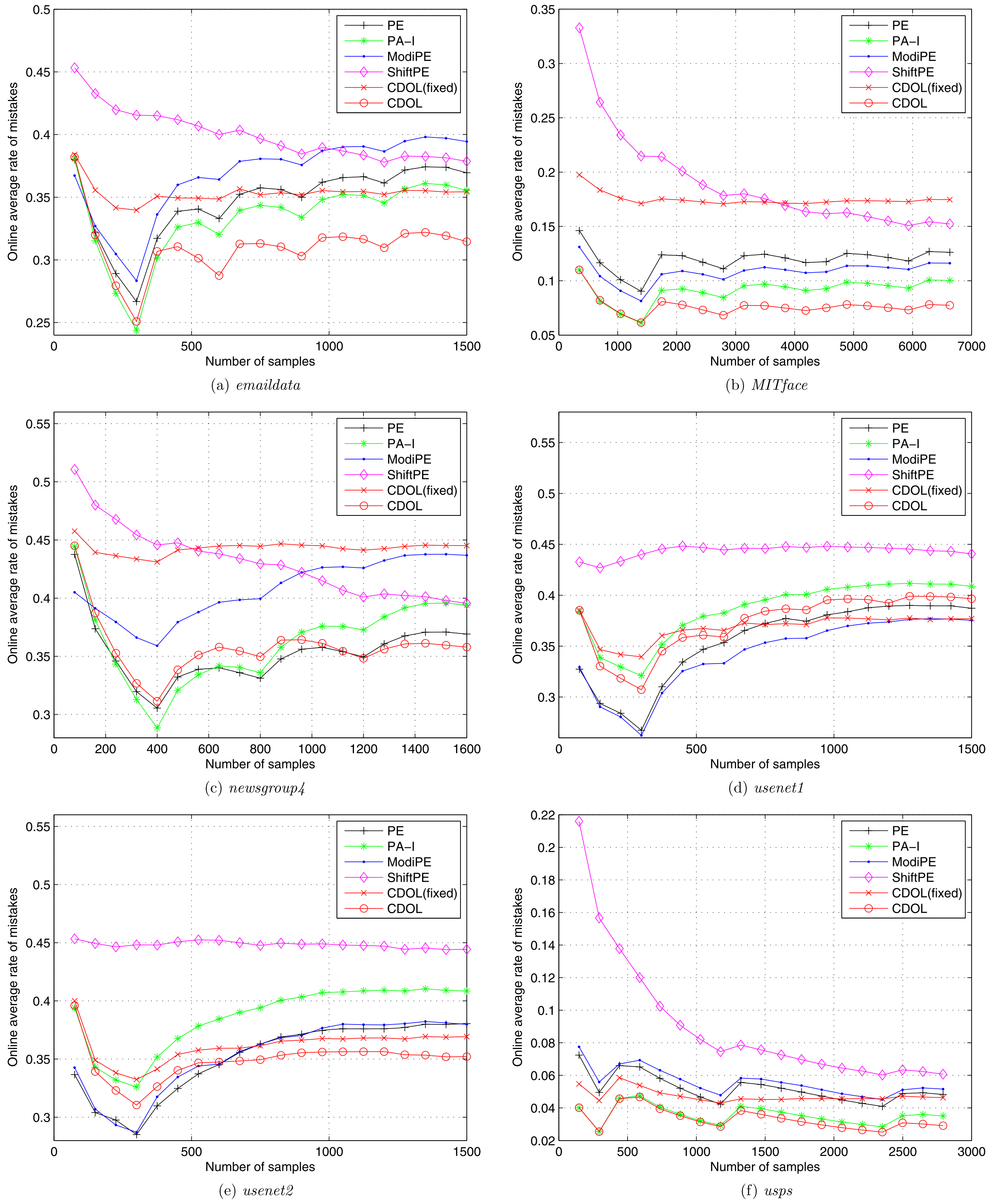
Table 10
Evaluation results of online learning with concept-drifting datasets.

Algorithm	emaildata		
	Mistake (%)	Support vectors (#)	Time (s)
PE:	36.9533 ± 1.1080	554.3000 ± 16.6199	0.0162 ± 0.0005
PA-I:	35.5333 ± 0.7849	1216.6000 ± 12.4495	0.0284 ± 0.0005
ShiftPE:	37.8567 ± 1.0473	567.8500 ± 15.7088	0.0173 ± 0.0005
ModiPE:	39.4367 ± 1.1184	591.5500 ± 16.7755	0.0176 ± 0.0011
CDOL(fixed):	35.4333 ± 1.0551	28.9500 ± 0.8870	0.0298 ± 0.0009
CDOL:	**31.4533** ± **1.6738**	338.9000 ± 132.1458	0.0701 ± 0.0019
Algorithm	MITface		
	Mistake (%)	Support vectors (#)	Time (s)
PE:	12.3979 ± 0.2360	865.0000 ± 16.4637	0.0921 ± 0.0024
PA-I:	9.8603 ± 0.1413	2023.6500 ± 29.0612	0.1931 ± 0.0042
ShiftPE:	14.9226 ± 0.3977	1041.1500 ± 27.7475	0.1158 ± 0.0030
ModiPE:	11.5021 ± 0.3681	802.5000 ± 25.6812	0.0874 ± 0.0023
CDOL(fixed):	17.4129 ± 1.7939	40.0000 ± 2.2942	0.1336 ± 0.0009
CDOL:	**7.6251** ± **0.2854**	714.2000 ± 196.6963	0.3656 ± 0.0168
Algorithm	newsgroup4		
	Mistake (%)	Support vectors (#)	Time (s)
PE:	36.9062 ± 1.1317	590.5000 ± 18.1064	0.0182 ± 0.0005
PA-I:	39.3875 ± 1.0398	1513.4500 ± 8.7748	0.0350 ± 0.0006
ShiftPE:	39.5562 ± 0.8385	632.9000 ± 13.4160	0.0205 ± 0.0008
ModiPE:	43.6781 ± 1.5975	698.8500 ± 25.5596	0.0206 ± 0.0009
CDOL(fixed):	44.5156 ± 1.2001	39.8000 ± 0.5231	0.0326 ± 0.0013
CDOL:	**35.7906** ± **1.5934**	432.4000 ± 211.6244	0.0809 ± 0.0025
Algorithm	usenet1		
	Mistake (%)	Support vectors (#)	Time (s)
PE:	38.7200 ± 1.3958	580.8000 ± 20.9375	0.0164 ± 0.0004
PA-I:	40.8833 ± 0.7807	958.8000 ± 19.3462	0.0240 ± 0.0005
ShiftPE:	44.0600 ± 1.1075	660.9000 ± 16.6129	0.0187 ± 0.0004
ModiPE:	**37.5333** ± **0.8463**	563.0000 ± 12.6948	0.0166 ± 0.0003
CDOL(fixed):	37.6967 ± 1.0132	22.3500 ± 2.9069	0.0281 ± 0.0002
CDOL:	39.6533 ± 1.6469	177.7500 ± 103.2941	0.0646 ± 0.0018
Algorithm	usenet2		
	Mistake (%)	Support vectors (#)	Time (s)
PE:	38.0467 ± 0.9167	570.7000 ± 13.7500	0.0172 ± 0.0025
PA-I:	40.8467 ± 0.8414	960.0000 ± 14.2349	0.0251 ± 0.0033
ShiftPE:	44.4367 ± 0.9498	666.5500 ± 14.2477	0.0197 ± 0.0024
ModiPE:	37.9833 ± 1.1255	569.7500 ± 16.8831	0.0171 ± 0.0003
CDOL(fixed):	36.9333 ± 1.6990	22.0500 ± 3.3321	0.0286 ± 0.0003
CDOL:	**35.2100** ± **1.3295**	193.7000 ± 100.1757	0.0646 ± 0.0017
Algorithm	usps		
	Mistake (%)	Support vectors (#)	Time (s)
PE:	4.7218 ± 0.2176	138.3500 ± 6.3766	0.0188 ± 0.0003
PA-I:	3.4266 ± 0.2289	443.3000 ± 9.4874	0.0295 ± 0.0005
ShiftPE:	5.9164 ± 0.3227	173.3500 ± 9.4550	0.0202 ± 0.0003
ModiPE:	5.0529 ± 0.3978	148.0500 ± 11.6550	0.0192 ± 0.0004
CDOL(fixed):	4.6143 ± 0.4057	34.0500 ± 3.3635	0.0530 ± 0.0002
CDOL:	**2.8208** ± **0.2188**	258.8500 ± 52.3584	0.1152 ± 0.0008

(e.g. $C > 4$) (except “usenet1”), which indicates a large learning rate can efficiently improve the transfer learning efficacy. Third, we observe that CDOL is significantly more accurate than the other two concept drift learning strategies: ModiPE and ShiftPE, under varied values of C, which again indicates the CDOL algorithm is more effective for concept drifting online learning tasks.

6. Conclusion

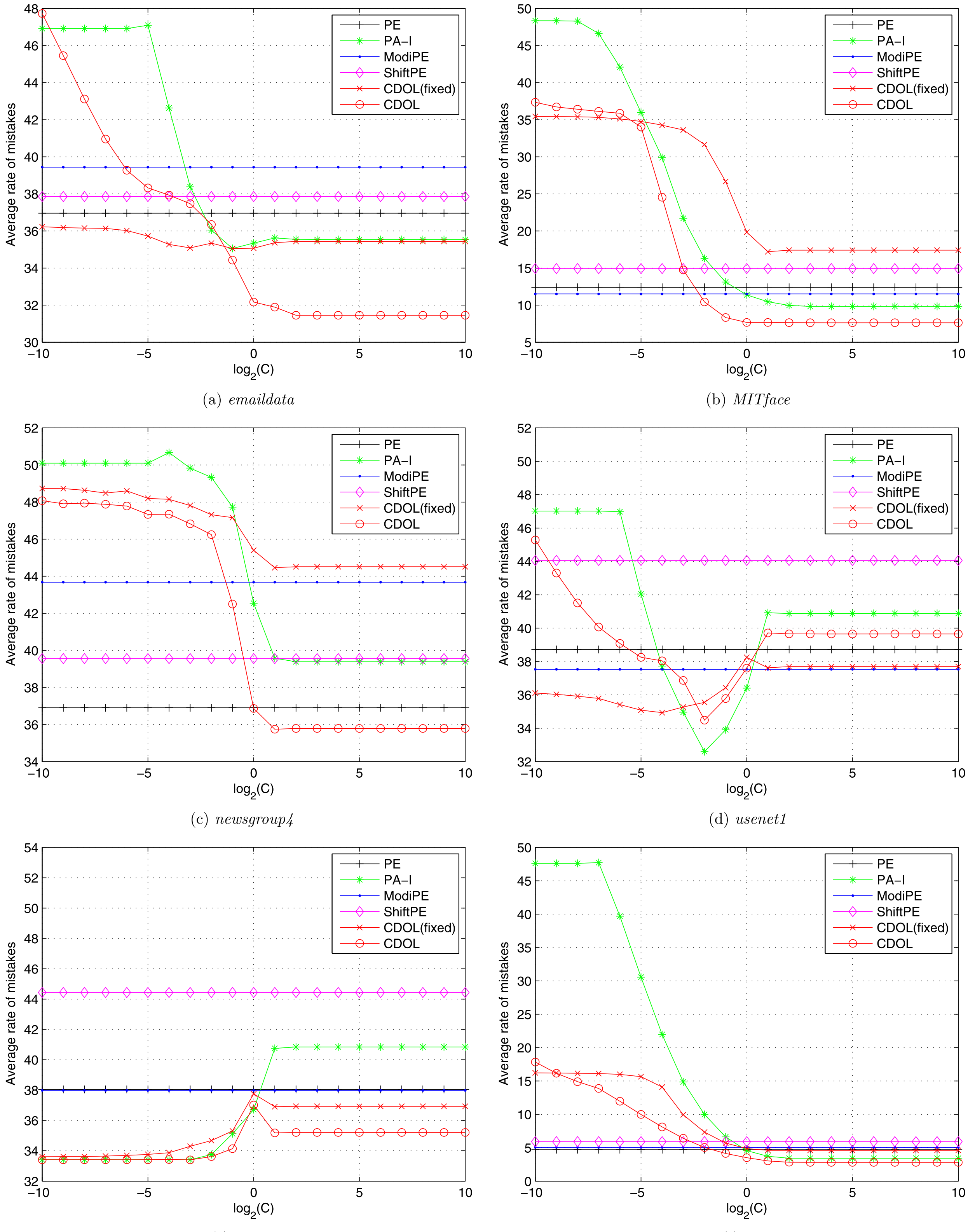
This paper presented a novel framework of *Online Transfer Learning* (OTL), which aims to attack an online learning task on a target domain by transferring knowledge from a source domain. We addressed two OTL tasks in classification setting and presented two OTL algorithms for different tasks. We offered theoretical analysis of the proposed OTL algorithms, and conducted an extensive set of experiments, in which encouraging results were obtained. Furthermore, we explored the OTL



(a) *emaildata* (b) *MITface* (c) *newsgroup4* (d) *usenet1* (e) *usenet2* (f) *usps*

Fig. 5. Evaluation of online mistake rates on the concept drift learning tasks.

technique as a natural extension to tackle the challenging task of learning over concept-drifting data streams, and proposed an effective algorithm which was validated by the promising empirical results. Through this work, we hope to encourage further research on in-depth investigations of OTL to address other hard problems, e.g., how to perform heterogeneous OTL



(a) *emaildata* (b) *MITface*

(c) *newsgroup4* (d) *usenet1*

(e) *usenet2* (f) *usps*

Fig. 6. Evaluation on the concept drift learning tasks with varied C values.

from complex data of completely diverse feature representations, how to derive better theoretical bounds of the proposed OTL algorithms, and how to develop new applications of OTL techniques to tackle other various challenges in AI community.

Acknowledgements

Bin Li is supported by "the Fundamental Research Funds for the Central Universities" and "the Academic Team Building Plans for Scholars Born in 1970's from Wuhan University".

Appendix A

A.1. Proof of Proposition 1

Proof. To facilitate the analysis, we denote $p_t = \alpha_{1,t}\Pi(\mathbf{v}^\top \mathbf{x}_t) + \alpha_{2,t}\Pi(\mathbf{w}_t^\top \mathbf{x}_t)$, $p_{1,t} = \Pi(\mathbf{v}^\top \mathbf{x}_t)$, and $p_{2,t} = \Pi(\mathbf{w}_t^\top \mathbf{x}_t)$.

It is straightforward to show that $F(z) = \exp(-\frac{1}{2}\ell^*(z, y))$ is concave with respect to z for all y. Then according to Jensen's inequality

$$\exp\left(-\frac{1}{2}\ell^*(p_t, \Pi(y_t))\right) \geq \frac{\sum_{i=1}^2 \alpha_{i,t}\exp(-\frac{1}{2}\ell^*(p_{i,t}, \Pi(y_t)))}{\sum_{i=1}^2 \alpha_{i,t}}.$$

Denoting $r_{i,t} = \ell^*(p_t, \Pi(y_t)) - \ell^*(p_{i,t}, \Pi(y_t))$ and rearranging the above inequality result in:

$$\sum_{i=1}^2 \alpha_{i,t}\exp\left(\frac{1}{2}r_{i,t}\right) \leq \sum_{i=1}^2 \alpha_{i,t}.$$

Combining the equality (2) with the above inequality, we have

$$\sum_{i=1}^2 \exp\left(-\frac{1}{2}\sum_{j=1}^{t-1}\ell^*(p_{i,j}, \Pi(y_j))\right)\exp\left(\frac{1}{2}r_{i,t}\right) \leq \sum_{i=1}^2 \exp\left(-\frac{1}{2}\sum_{j=1}^{t-1}\ell^*(p_{i,j}, \Pi(y_j))\right).$$

Multiplying the two sides of the above inequality with $\exp(\frac{1}{2}\sum_{j=1}^{t-1}\ell^*(p_j, \Pi(y_j)))$ results in

$$\sum_{i=1}^2 \exp\left(\frac{1}{2}\sum_{j=1}^{t} r_{i,j}\right) \leq \sum_{i=1}^2 \exp\left(\frac{1}{2}\sum_{j=1}^{t-1} r_{i,j}\right),$$

which further implies

$$\sum_{i=1}^2 \exp\left(\frac{1}{2}\sum_{j=1}^{T} r_{i,j}\right) \leq \sum_{i=1}^2 \exp\left(\frac{1}{2}\sum_{j=1}^{T-1} r_{i,j}\right) \leq \ldots \sum_{i=1}^2 \exp\left(\frac{1}{2}\sum_{j=1}^{0} r_{i,j}\right) = 2\ln(2).$$

Finally,

$$\sum_{t=1}^T \ell^*(p_t, \Pi(y_t)) - \min_{i=1,2}\sum_{t=1}^T \ell^*(p_{i,t}, \Pi(y_t)) = \max_{i=1,2}\sum_{t=1}^T r_{i,t} \leq 2\ln\left(\sum_{i=1}^2 \exp\left(\frac{1}{2}\sum_{t=1}^T r_{i,t}\right)\right) \leq 2\ln(2).$$

Plugging $p_t = \alpha_{1,t}\Pi(\mathbf{v}^\top \mathbf{x}_t) + \alpha_{2,t}\Pi(\mathbf{w}_t^\top \mathbf{x}_t)$, $p_{1,t} = \Pi(\mathbf{v}^\top \mathbf{x}_t)$, and $p_{2,t} = \Pi(\mathbf{w}_t^\top \mathbf{x}_t)$ into the above inequality concludes the claim. □

A.2. Proof of Proposition 2

Proof. It is not difficult to show the optimization in (6) is equivalent

$$\min_{\mathbf{w}_1 \mathbf{w}_2} \frac{\gamma_1}{2}\|\mathbf{w}_1 - \mathbf{w}_{1,t}\|^2 + \frac{\gamma_2}{2}\|\mathbf{w}_2 - \mathbf{w}_{2,t}\|^2 + C\xi$$
$$s.t.\quad 1 - y_t\frac{1}{2}(\mathbf{w}_1^\top \mathbf{x}_{1,t} + \mathbf{w}_2^\top \mathbf{x}_{2,t}) \leq \xi \quad and \quad \xi \geq 0$$

The Lagrangian of the above optimization problem is expressed as

$$\mathcal{L}(\mathbf{w}_1,\mathbf{w}_2,\xi,\tau_t,\lambda)$$
$$= \frac{\gamma_1}{2}\|\mathbf{w}_1-\mathbf{w}_{1,t}\|^2+\frac{\gamma_2}{2}\|\mathbf{w}_2-\mathbf{w}_{2,t}\|^2+\xi(C-\tau_t-\lambda)+\tau_t\left(1-y_t\frac{1}{2}\left(\mathbf{w}_1^\top\mathbf{x}_{1,t}+\mathbf{w}_2^\top\mathbf{x}_{2,t}\right)\right) \tag{1}$$

where $\tau_t \ge 0$ and $\lambda \ge 0$ are Lagrange multipliers. We now find the minimum of the Lagrangian w.r.t. $\mathbf{w}_1$, $\mathbf{w}_2$ and ξ by setting their partial derivatives to zeros. We get $\mathbf{w}_i = \mathbf{w}_{i,t}+\frac{\tau_t}{2\gamma_i}y_t\mathbf{x}_{i,t}$ for $i=1,2$ and $C-\tau_t-\lambda=0$. And since $\lambda\ge 0$, we conclude $C\ge\tau_t$. We thus have $\tau_t\in[0,C]$. Plugging the three equations $\mathbf{w}_i=\mathbf{w}_{i,t}+\frac{\tau_t}{2\gamma_i}y_t\mathbf{x}_{i,t}$ (where $i=1,2$) and $C-\tau_t-\lambda=0$ into Eq. (1), we have

$$\mathcal{L}(\tau_t)=-\tau_t^2\left(\frac{z_{1,t}}{8\gamma_1}+\frac{z_{2,t}}{8\gamma_2}\right)+\tau_t\ell_t,\quad \text{where } \ell_t=\ell(\mathbf{w}_{1,t},\mathbf{w}_{2,t};t).$$

Setting the derivative of the above to zero leads to:

$$\tau_t=\ell_t\Big/\left(\frac{z_{1,t}}{4\gamma_1}+\frac{z_{2,t}}{4\gamma_2}\right)=\frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2+z_{2,t}\gamma_1}.$$

Finally, combining the fact $\tau_t\in[0,C]$, we thus have the final solution:

$$\tau_t=\min\left\{C,\frac{4\gamma_1\gamma_2\ell_t}{z_{1,t}\gamma_2+z_{2,t}\gamma_1}\right\}.\qquad\square$$

A.3. Proof of Lemma 1

Proof. Let us introduce the following notation:

$$\Delta_t=\frac{\gamma_1}{2}\left(\|\mathbf{w}_{1,t}-\mathbf{w}_1\|^2-\|\mathbf{w}_{1,t+1}-\mathbf{w}_1\|^2\right)+\frac{\gamma_2}{2}\left(\|\mathbf{w}_{2,t}-\mathbf{w}_2\|^2-\|\mathbf{w}_{2,t+1}-\mathbf{w}_2\|^2\right).$$

We then have

$$\begin{aligned}\sum_{t=1}^{T}\Delta_t&=\sum_{t=1}^{T}\left\{\frac{\gamma_1}{2}\left(\|\mathbf{w}_{1,t}-\mathbf{w}_1\|^2-\|\mathbf{w}_{1,t+1}-\mathbf{w}_1\|^2\right)+\frac{\gamma_2}{2}\left(\|\mathbf{w}_{2,t}-\mathbf{w}_2\|^2-\|\mathbf{w}_{2,t+1}-\mathbf{w}_2\|^2\right)\right\}\\&=\frac{\gamma_1}{2}\left(\|\mathbf{v}-\mathbf{w}_1\|^2-\|\mathbf{w}_{1,T+1}-\mathbf{w}_1\|^2\right)+\frac{\gamma_2}{2}\left(\|\mathbf{w}_{2,1}-\mathbf{w}_2\|^2-\|\mathbf{w}_{2,T+1}-\mathbf{w}_2\|^2\right)\\&\le\frac{\gamma_1}{2}\left(\|\mathbf{v}-\mathbf{w}_1\|^2\right)+\frac{\gamma_2}{2}\left(\|\mathbf{w}_2\|^2\right)\end{aligned}$$

Second, when $\ell_t=0$, $\mathbf{w}_{i,t+1}=\mathbf{w}_{i,t}$ for $i=1,2$, it is clear $\Delta_t=0$; when $\ell_t>0$, $\mathbf{w}_{i,t+1}=\mathbf{w}_{i,t}+\frac{\tau_t}{2\gamma_i}y_t\mathbf{x}_{i,t}$, we compute Δ_t as:

$$\begin{aligned}\Delta_t&=\frac{\gamma_1}{2}\left(\|\mathbf{w}_{1,t}-\mathbf{w}_1\|^2-\|\mathbf{w}_{1,t+1}-\mathbf{w}_1\|^2\right)+\frac{\gamma_2}{2}\left(\|\mathbf{w}_{2,t}-\mathbf{w}_2\|^2-\|\mathbf{w}_{2,t+1}-\mathbf{w}_2\|^2\right)\\&=\tau_t\left\{-\frac{y_t}{2}\left(\mathbf{w}_{1,t}^\top\mathbf{x}_{1,t}+\mathbf{w}_{2,t}^\top\mathbf{x}_{2,t}\right)+\frac{y_t}{2}\left(\mathbf{w}_1^\top\mathbf{x}_{1,t}+\mathbf{w}_2^\top\mathbf{x}_{2,t}\right)-\left(\frac{z_{1,t}}{8\gamma_1}+\frac{z_{2,t}}{8\gamma_2}\right)\tau_t\right\}\end{aligned}\tag{2}$$

We also have $\ell_t=1-y_t(\frac{1}{2}(\mathbf{w}_{1,t}^\top\mathbf{x}_{1,t}+\mathbf{w}_{2,t}^\top\mathbf{x}_{2,t}))$ as $\ell_t>0$, which is equivalent to:

$$\frac{y_t}{2}\left(\mathbf{w}_{1,t}^\top\mathbf{x}_{1,t}+\mathbf{w}_{2,t}^\top\mathbf{x}_{2,t}\right)=1-\ell_t.$$

In addition,

$$\ell(\mathbf{w}_1,\mathbf{w}_2;t)=\left[1-y_t\frac{1}{2}\left(\mathbf{w}_1^\top\mathbf{x}_{1,t}+\mathbf{w}_2^\top\mathbf{x}_{2,t}\right)\right]_+\ge 1-y_t\frac{1}{2}\left(\mathbf{w}_1^\top\mathbf{x}_{1,t}+\mathbf{w}_2^\top\mathbf{x}_{2,t}\right),$$

we thus have

$$\frac{y_t}{2}\left(\mathbf{w}_1^\top\mathbf{x}_{1,t}+\mathbf{w}_2^\top\mathbf{x}_{2,t}\right)\ge 1-\ell(\mathbf{w}_1,\mathbf{w}_2;t).$$

Combining these two facts and inequality (2), we thus have the following:

$$\begin{aligned}\Delta_t&\ge\tau_t\left(-(1-\ell_t)+1-\ell(\mathbf{w}_1,\mathbf{w}_2;t)-\left(\frac{z_{1,t}}{8\gamma_1}+\frac{z_{2,t}}{8\gamma_2}\right)\tau_t\right)\\&=\tau_t\left(\ell_t-\ell(\mathbf{w}_1,\mathbf{w}_2;t)-\left(\frac{z_{1,t}}{8\gamma_1}+\frac{z_{2,t}}{8\gamma_2}\right)\tau_t\right)\end{aligned}$$

Hence, we have the following conclusion:

$$\sum_{t=1}^{T} \tau_t \left(\ell_t - \ell(\mathbf{w}_1, \mathbf{w}_2; t) - \left(\frac{z_{1,t}}{8\gamma_1} + \frac{z_{2,t}}{8\gamma_2} \right) \tau_t \right) \le \frac{\gamma_1}{2} \|\mathbf{v} - \mathbf{w}_1\|^2 + \frac{\gamma_2}{2} \|\mathbf{w}_2\|^2. \qquad \square$$

A.4. Proof of Proposition 3

Proof. First of all, we note that $\Pi(y_t) = \frac{y_t+1}{2}$ since $y_t \in \{-1, +1\}$.

Case 1. If $y_t \mathbf{w}_t^\top \mathbf{x}_t \in (-\infty, -1]$, then $\ell_t \ge 2$.

1.1. If $y_t = -1$, then $\mathbf{w}_t^\top \mathbf{x}_t \ge +1$ and $\Pi(\mathbf{w}_t^\top \mathbf{x}_t) = 1$ and $\Pi(y_t) = 0$;
1.2. If $y_t = +1$, then $\mathbf{w}_t^\top \mathbf{x}_t \le -1$ and $\Pi(\mathbf{w}_t^\top \mathbf{x}_t) = 0$ and $\Pi(y_t) = 1$;

Accordingly, we have $\ell_t^* = (\Pi(\mathbf{w}_t^\top \mathbf{x}_t) - \Pi(y_t))^2 = 1 \le \frac{\ell_t^2}{4}$ (or $\le \frac{\ell_t}{2}$).

Case 2. If $y_t \mathbf{w}_t^\top \mathbf{x}_t \in (-1, +1)$, since $y_t \in \{-1, +1\}$, then $\mathbf{w}_t^\top \mathbf{x}_t \in (-1, +1)$, and $\Pi(\mathbf{w}_t^\top \mathbf{x}_t) = \frac{\mathbf{w}_t^\top \mathbf{x}_t + 1}{2} \in (0, +1)$, as a result:

$$\ell_t^* = \left(\Pi\left(\mathbf{w}_t^\top \mathbf{x}_t\right) - \Pi(y_t)\right)^2 = \left(\frac{\mathbf{w}_t^\top \mathbf{x}_t + 1}{2} - \frac{y_t + 1}{2} \right)^2 = \left(\frac{1 - y_t \mathbf{w}_t^\top \mathbf{x}_t}{2} \right)^2 = \frac{\ell_t^2}{4} \le \frac{\ell_t}{2}.$$

Case 3. If $y_t \mathbf{w}_t^\top \mathbf{x}_t \in [+1, +\infty)$, $\ell_t = 0$.

3.1. If $y_t = -1$, then $\mathbf{w}_t^\top \mathbf{x}_t \le -1$ and $\Pi(\mathbf{w}_t^\top \mathbf{x}_t) = 0 = \Pi(y_t)$;
3.2. If $y_t = +1$, then $\mathbf{w}_t^\top \mathbf{x}_t \ge +1$ and $\Pi(\mathbf{w}_t^\top \mathbf{x}_t) = 1 = \Pi(y_t)$;

Accordingly, we have $\ell_t^* = (\Pi(\mathbf{w}_t^\top \mathbf{x}_t) - \Pi(y_t))^2 = 0 = \ell_t$.

In summary, we have $\ell_t^* \le \min\{\ell_t/2, \ell_t^2/4\}$. $\square$

References

[1] Andreas Argyriou, Andreas Maurer, Massimiliano Pontil, An algorithm for transfer learning in a heterogeneous environment, in: ECML/PKDD'08, Antwerp, Belgium, 2008, pp. 71–85.
[2] Andrew Arnold, Ramesh Nallapati, William W. Cohen, A comparative study of methods for transductive transfer learning, in: Proc. 7th IEEE ICDM Workshops, 2007, pp. 77–82.
[3] Peter L. Bartlett, Learning with a slowly changing distribution, in: COLT, 1992, pp. 243–252.
[4] Avrim Blum, Alan M. Frieze, Ravi Kannan, Santosh Vempala, A polynomial-time algorithm for learning noisy linear threshold functions, Algorithmica 22 (1/2) (1998) 35–52.
[5] Avrim Blum, Tom Mitchell, Combining labeled and unlabeled data with co-training, in: Proceedings of the Eleventh Annual Conference on Computational Learning Theory, ACM, 1998, pp. 92–100.
[6] Giovanni Cavallanti, Nicolò Cesa-Bianchi, Claudio Gentile, Tracking the best hyperplane with a simple budget perceptron, Mach. Learn. 69 (2–3) (2007) 143–167.
[7] Nicolò Cesa-Bianchi, Alex Conconi, Claudio Gentile, On the generalization ability of on-line learning algorithms, IEEE Trans. Inf. Theory 50 (9) (2004) 2050–2057.
[8] Nicolo Cesa-Bianchi, Gabor Lugosi, Prediction, Learning, and Games, Cambridge University Press, New York, NY, USA, 2006.
[9] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, Yoram Singer, Online passive–aggressive algorithms, J. Mach. Learn. Res. 7 (2006) 551–585.
[10] Koby Crammer, Yishay Mansour, Eyal Even-Dar, Jennifer Wortman Vaughan, Regret minimization with concept drift, in: COLT, 2010, pp. 168–180.
[11] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, Yong Yu, Self-taught clustering, in: ICML, Helsinki, Finland, 2008, pp. 200–207.
[12] H. Daumá III, D. Marcu, Domain adaptation for statistical classifiers, J. Artif. Intell. Res. 26 (2006) 101–126.
[13] Ofer Dekel, Philip M. Long, Yoram Singer, Online learning of multiple tasks with a shared loss, J. Mach. Learn. Res. 8 (2007) 2233–2264.
[14] Yoav Freund, Robert E. Schapire, Large margin classification using the perceptron algorithm, Mach. Learn. 37 (3) (1999) 277–296.
[15] Mohamed Medhat Gaber, Arkady B. Zaslavsky, Shonali Krishnaswamy, Mining data streams: a review, SIGMOD Rec. 34 (2) (2005) 18–26.
[16] Elad Hazan, C. Seshadhri, Efficient learning algorithms for changing environments, in: ICML, 2009, p. 50.
[17] David P. Helmbold, Philip M. Long, Tracking drifting concepts by minimizing disagreements, Mach. Learn. 14 (1) (1994) 27–45.
[18] Mark Herbster, Manfred K. Warmuth, Tracking the best linear predictor, J. Mach. Learn. Res. 1 (2001) 281–309.
[19] Steven C.H. Hoi, Jialei Wang, Peilin Zhao, LIBOL: a library for online learning algorithms, J. Mach. Learn. Res. 15 (1) (2014) 495–499.
[20] Matthew Klenk, Kenneth D. Forbus, Analogical model formulation for transfer learning in AP physics, Artif. Intell. 173 (18) (2009) 1615–1638.
[21] Ralf Klinkenberg, Learning drifting concepts: example selection vs. example weighting, Intell. Data Anal. 8 (August 2004) 281–300.
[22] Ralf Klinkenberg, Thorsten Joachims, Detecting concept drift with support vector machines, in: ICML, 2000, pp. 487–494.
[23] Yi Li, Philip M. Long, The relaxed online maximum margin algorithm, in: NIPS, 1999, pp. 498–504.
[24] J. Michaelsen, Cross-validation in statistical climate forecast models, J. Appl. Meteorol. 26 (November 1987) 1589–1600.
[25] Eirinaios Michelakis, Ion Androutsopoulos, Georgios Paliouras, George Sakkis, Panagiotis Stamatopoulos, Filtron: a learning-based anti-spam filter, in: Proceedings of The 1st Conference on Email and Anti-Spam, 2004.
[26] Alexandru Niculescu-Mizil, Rich Caruana, Inductive transfer for Bayesian network structure learning, in: AISTATS, 2007.

[27] Sinno Jialin Pan, Qiang Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (October 2010) 1345–1359.
[28] F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain, Psychol. Rev. 65 (1958) 386–407.
[29] Vikas Sindhwani, David S. Rosenberg, An RKHS for multi-view learning and manifold co-regularization, in: Proceedings of the 25th international conference on Machine learning, ACM, 2008, pp. 976–983.
[30] Jialei Wang, Peilin Zhao, Steven C.H. Hoi, Exact soft confidence-weighted learning, in: Proceedings of the 29th International Conference on Machine Learning (ICML-12), Edinburgh, Scotland, 26 June – 1 July, 2012, 2012.
[31] Zheng Wang, Yangqiu Song, Changshui Zhang, Transferred dimensionality reduction, in: ECML/PKDD'08, Antwerp, Belgium, 2008, pp. 550–565.
[32] Haiqin Yang, Zenglin Xu, Irwin King, Michael Lyu, Online learning for group Lasso, in: ICML, Haifa, Israel, 2010.
[33] Peilin Zhao, Steven C.H. Hoi, OTL: a framework of online transfer learning, in: ICML, 2010, pp. 1231–1238.
[34] Peilin Zhao, Steven C.H. Hoi, Rong Jin, Double updating online learning, J. Mach. Learn. Res. 12 (2011) 1587–1615.
[35] Peilin Zhao, Rong Jin, Tianbao Yang, Steven C. Hoi, Online AUC maximization, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 233–240.
[36] Peilin Zhao, Jialei Wang, Pengcheng Wu, Rong Jin, Steven C.H. Hoi, Fast bounded online gradient descent algorithms for scalable kernel-based online learning, in: Proceedings of the 29th International Conference on Machine Learning (ICML-12), Edinburgh, Scotland, 26 June – 1 July, 2012, 2012.
[37] Indre Zliobaite, Learning under concept drift: an overview, CoRR abs/1010.4784, 2010.