Online Transfer Learning with Heterogeneous Source

Anonymous Author(s)
Affiliation
Address
email

Abstract

Heterogeneous transfer learning aims to build the model in the target domain by leveraging knowledge extracted from other heterogeneous source domains. In this paper, we investigate online heterogeneous transfer learning problem, which considers a classification task on the target domain in an online fashion. The motivation of our work is to utilize data in the heterogeneous source domain to promote the ability of classifier. The challenges in online heterogeneous transfer mainly come from two aspects. The first one is that the feature spaces of the source and target domains are different, which leads to the difficulty for knowledge transfer. The second one is that we do not have enough prior training data from the target domain to build a precise relationship across the source and target domains. With regard to aforementioned problems, we propose effective algorithms based on the ensemble strategy. We first construct a connection between the source and target domains using co-occurrence data, and then build a weighted K nearest neighbor classifier via heterogeneous data. By combining with the hypothesis trained in the target domain, we obtain the ensemble classifier. We also theoretically analyze the mistake bound of our proposed algorithms under the $\mathbf{Hedge}(\beta)$ framework. The experiments are conducted for image classification by exploiting the knowledge learnt from the text data. Encouraging results demonstrate the effectiveness of our algorithms.

1 Introduction

Transfer learning [**?**], which devotes to transferring knowledge extracted from the source domain to the target domain, has been actively studied in recent years. According to the relationship between the feature spaces of the source and target domains, we can categorize transfer learning into two groups: homogeneous transfer and heterogeneous transfer. A great number of works in the last decade deal with homogeneous transfer learning problem [**?**, **?**, **?**, **?**], which assumes that the data spaces of the source and target domains are identical. Nevertheless, more and more studies consider heterogeneous transfer learning problem. Heterogeneous transfer addresses the situation that either the feature spaces or the output spaces of the source and target domains are diverse.

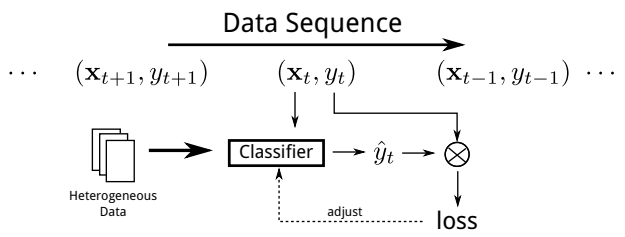


Figure 1: The illustration of online heterogeneous transfer learning problem

Most works regarding heterogeneous transfer address an offline learning problem. In this paper, however, we investigate online heterogeneous transfer learning problem where test data instances arrive sequentially. At the same time, we consider the situation that the feature spaces of the source and target domains are completely different. Data instance in the source and target domains could be image and text, or English document and Chinese document. Figure **??** illustrates the framework of online heterogeneous transfer. At t-th trial, the classifier receives an instance $\mathbf{x}_t$ and predicts a class label $\hat{y}_t$ immediately. Then the true class label y_t arrives and a loss value is calculated. According to the loss value, the classifier adjust itself to promote the ability of classification. Our motivation is to achieve better performance on a classification task in the target domain using knowledge learnt from the data in the heterogeneous source domain.

There are two principal challenges in online heterogeneous transfer learning problem. The first one is the fact that the feature spaces of the source and target domains are completely diverse, which makes it difficult to construct relationship across domains. The second one comes from the characteristic of online learning problem. As the data instances arrive in an sequential manner, we do not have enough training data from the target domain in advance. Thus, we cannot adopt a training method to build a precise relationship between the source and target domains.

In order to solve aforementioned problems, we propose **O**nline **H**eterogeneous **T**ransfer (OHT) algorithms based on the ensemble strategy. Our idea is to transfer knowledge via co-occurrence information across the source and target domains. Firstly, by calculating the similarity between the test instance and the heterogeneous data instance using co-occurrence information, we construct a connection across the domains. Next, we adopt weighted K nearest neighbor algorithm to obtain hypothesis in the heterogeneous source domain. At the same time, a traditional online learning algorithm is applied in the target domain to produce another hypothesis. By combining two hypotheses, we get the ensemble classifier. We also analyze the theoretical bound of our proposed algorithms.

The experiments are conducted for image classification by taking advantage of knowledge in the text data. We make use of NUS-WIDE dataset to generate binary classification tasks. Experimental results show that our algorithms outperform the baseline methods.

2 Related work

The topics that our work is related to are mainly heterogeneous transfer learning and online learning. In this section, we review some related studies and discuss the differences between our work and the existing works.

2.1 Heterogeneous transfer learning

The objective of heterogeneous transfer learning is to solve a problem in the target domain by leveraging knowledge extracted from other heterogeneous source domains with different feature space. wei2011heterogeneouswei2011heterogeneous applied restricted Boltzmann machine to perform heterogeneous transfer learning task. And deep learning technique is introduced into heterogeneous transfer in [**?**]. There are also some existing works studying heterogeneous transfer using co-occurrence data. yang2009heterogeneousyang2009heterogeneous leveraged text data for image clustering, and zhu2011heterogeneouszhu2011heterogeneous utilized text data for image classification task. wang2009buildingwang2009building proposed a algorithm to build two separate classifiers based on text features and image features, and then the final classifier is trained to combine them. And in [**?**, **?**, **?**] transition probabilities based on co-occurrence information is used to deal with classification task.

However, above-mentioned studies require training data in the source and target domains. HTLIC algorithm [**?**] utilizes training data to learn high-level features which help to construct extra representation for target images. And the transition probabilities matrix used in [**?**, **?**] also need prior training data in the source and target domains. In our problem setting, since all the data instance in the target domain arrive sequentially, we do not have enough data for training process. To deal with this problem, we adopt a lazy classification algorithm using heterogeneous data without batch training procedure.

2.2 Online learning

Online learning has been actively studied for many years. Unlike the offline machine learning problem where training data are given first, online learning learner receives data instances sequentially. An online learning learner make a prediction immediately and adjust itself during the process of the task. One of the classic algorithms of online learning is Perceptron algorithm [**?**], which move the decision bound towards the direction of an instance which is labeled wrong. Passive-Aggressive algorithm and its variants take the criterion of maximum margin into account [**?**].

However, only a few works address transfer learning in an online fashion. In [**?**, **?**], the ensemble strategy is adopted to deal with online homogeneous transfer learning problem, and the multi-view approach is used to handle online heterogeneous transfer learning problem. Nevertheless, the multi-view approach for online heterogeneous transfer in [**?**, **?**] requires that the feature space of the source domain is a subset of that of the target domain. Instead, we consider a situation that the feature spaces of the source and target domains do not share any common subset. Co-occurrence data are used to help us transfer knowledge from the heterogeneous source domain to the target domain.

3 Problem definition

Suppose that we are given some instances $\{\mathbf{x}_i^s, y_i^s\}_{i=1}^{n^s}$ in the source data space $\mathcal{X}^s \times \mathcal{Y}^s$, where $\mathcal{X}^s = \mathbb{R}^{d^s}$ and $\mathcal{Y}^s = \{+1, -1\}$. The objective of online heterogeneous transfer is to learn a prediction function $f(\mathbf{x}_t)$ to classify the instance on the target domain in an online fashion. The data space of the target domain is $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \{+1, -1\}$. Specifically, the task of online heterogeneous transfer learning is a sequential process, during which an instance $\mathbf{x}_t$ comes at the t-th trial, and the classifier generates a predicted class label $\hat{y}_t$. Then the classifier receives the correct class label y_t and update itself to obtain a better classification ability.

In our problem setting, $\mathcal{Y} = \mathcal{Y}^s$, which means the same class label in two domains indicates the same class. For instance, $+1$ indicates vehicle class and -1 represents tree class. While the feature spaces of the source and target domains are completely different. Formally, $\mathcal{X} \cap \mathcal{X}^s = \varnothing$. As We cannot directly transfer knowledge from a completely different source domain, a sophisticated method of knowledge transfer is required.

4 Methods

In this section, we describe the details of our proposed algorithms. We first introduce our approach of how to establish relationship between two different domains to build classifier using heterogeneous source data. Then we present online heterogeneous transfer learning algorithms.

4.1 Heterogeneous knowledge transfer

In order to transfer knowledge in the heterogeneous source to the target domain, we construct a connection between two domains, and then design a hypothesis to make prediction using auxiliary data in the heterogeneous source domain.

Our idea is to bridge two domains via their co-occurrence information. Given some co-occurrence data instances $\{\mathbf{x}_i^o\}_{i=1}^{n^o}$ in the data space $\mathcal{X}^o$, and $\mathcal{X}^o$ can be split into two part $\mathcal{X}^{(1)}$ and $\mathcal{X}^{(2)}$. Then the co-occurrence data instance $\mathbf{x}_i^o$ can also be represented by $(\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)})$. The first part of the feature space of co-occurrence data is the same as that of the target domain, and the second part is the same as that of the source domain. Formally, we have $\mathcal{X}^{(1)} = \mathcal{X}$, and $\mathcal{X}^{(2)} = \mathcal{X}^s$.

It is convenience to collect co-occurrence data from the Internet. For example, tagged images from Flickr can be used as the co-occurrence data between the data spaces of image and text. When $\mathcal{X}$ is the feature space of English document and $\mathcal{X}^s$ is the feature space of Chinese document, the co-occurrence data can be collected from Wikipedia web pages titled by the corresponding term.

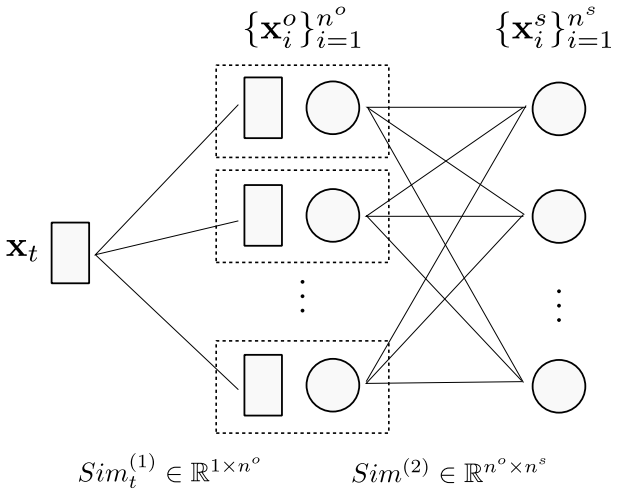


Figure 2: The illustration of our approach for heterogeneous knowledge transfer

As $\mathcal{X}^{(1)} = \mathcal{X}$, at t-th trial, we can construct a similarity vector $Sim_t^{(1)}$ to measure the affinity between $\mathbf{x}_t$ and each co-occurrence data instance. The i-th element $Sim_t^{(1)}(i)$ is the Pearson correlation between $\mathbf{x}_t$ and $\mathbf{x}_i^o$. The formula shows as follows:

$$Sim_t^{(1)}(i) = \frac{(\mathbf{x}_t - \bar{\mathbf{x}}_t)\cdot(\mathbf{x}_i^{(1)} - \bar{\mathbf{x}}_i^{(1)})}{\|\mathbf{x}_t - \bar{\mathbf{x}}_t\|\|\mathbf{x}_i^{(1)} - \bar{\mathbf{x}}_i^{(1)}\|}$$

where $\bar{\mathbf{z}} = mean(\mathbf{z})$. Also, we can construct a similarity matrix $Sim^{(2)}$ to measure the affinity between the co-occurrence data instances and the data instances in the heterogeneous source domain. The formula of the (i,j)-th element $Sim^{(2)}(i,j)$ is

$$Sim^{(2)}(i,j) = \frac{(\mathbf{x}_i^{(2)} - \bar{\mathbf{x}}_i^{(2)})\cdot(\mathbf{x}_j^{s} - \bar{\mathbf{x}}_j^{s})}{\|\mathbf{x}_i^{(2)} - \bar{\mathbf{x}}_i^{(2)}\|\|\mathbf{x}_j^{s} - \bar{\mathbf{x}}_j^{s}\|}$$

Figure **??** illustrates the calculation of $Sim_t^{(1)}$ and $Sim^{(2)}$

Finally, we obtain a similarity vector Sim_t between $\mathbf{x}_t$ and each instance in the heterogeneous source domain, with its j-th entry is calculated by

$$Sim_t(j) = \sum_i Sim_t^{(1)}(i) Sim^{(2)}(i,j)$$

Based on Sim_t, we predict the class label of instance $\mathbf{x}_t$ by weighted K nearest neighbor classifier:

$$h_t^s(\mathbf{x}_t) = \sum_{i\in N} \frac{Sim_t(i)}{\sum\limits_{i\in N} Sim_t(i)} y_i$$

where N is the set of identifiers of K nearest neighbors in the heterogeneous source domain.

4.2 Online heterogeneous transfer algorithms

Given a hypothesis obtained from the hetegeneous source domain, we can utilize the ensemble learning strategy to design classifier.

We first construct a prediction function in the target domain using online learning algorithm PA [**?**], and then combine two hypotheses to get the final classifier. According to the loss value suffered by each hypothesis seperately, we dynamically update two weights of hypotheses. Without loss of generality, we use linear function in the target domain to simplify the description. Through the kernel trick, we are able to tackle non-linear classification problem.

Algorithm 1 presents the process of the proposed online heterogeneous transfer learning algorithm 1 (OHT1). w_1^s and w_1 reflect the prior confidence we have on two hypotheses, and must sum to 1. If we have no preference on any of them, we can simply set $w_1^s = w_t = \frac{1}{2}$, which is also the setting we used in our experiments. Function $\Omega(z) = \max\{0, \min\{1, \frac{z+1}{2}\}\}$ projects $z \in \mathbb{R}$ to range $[0, 1]$. In order to adjust two weights dynamically, exponentially weighting update method [**?**] is used.

It is worthwhile to note that the equation in step 8, which calculate the updating factor of linear classifier trained in the target domain, can be replaced by two variants. In [**?**], PA-I algorithm introduces a non-negative linear slack variable into PA, and PA-II algorithm introduces a quadratic slack variable. The corresponding equations of calculation of learning rate τ are $\tau_t = \min\{C, \frac{\ell_t^*}{\|\mathbf{x}_t\|^2}\}$ and $\tau_t = \frac{\ell_t^*}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}}$, respectively. Likewise, by replacing the equation in Step 8, we obtain two variants OHT1-I and OHT1-II.

Algorithm 1 Online Heterogeneous Transfer Algorithm 1 (OHT1)

Input: aggressiveness parameter $C > 0$
$\eta = \frac{1}{2}$
data in heterogeneous source

Initialize: $\mathbf{v} = \mathbf{0}$, $w_1^s \in (0, 1)$, $w_1 \in (0, 1)$, where $w_1^s + w_1 = 1$

1: **for** $t = 1$ to T **do**
2: receive instance: $\mathbf{x}_t \in \mathcal{X}$
3: normalize: $\theta_t^s = \frac{w_t^s}{w_t^s + w_t}, \theta_t = \frac{w_t}{w_t^s + w_t}$
4: predict: $\hat{y}_t = \text{sign}\left(\theta_t^s \Omega(h_t^s(\mathbf{x}_t)) + \theta_t \Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \frac{1}{2}\right)$
5: receive correct label: $y_t \in \mathcal{Y}$
6: compute:

$$w_{t+1}^s = w_t^s \exp\left\{-\eta\big(\Omega(h_t^s(\mathbf{x}_t)) - \Omega(y_t)\big)^2\right\}$$
$$w_{t+1} = w_t \exp\left\{-\eta\big(\Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t)\big)^2\right\}$$

7: suffer loss: $\ell_t^* = \max\{0, 1 - y_t(\mathbf{v}_t \cdot \mathbf{x}_t)\}$
8: set: $\tau_t = \frac{\ell_t^*}{\|\mathbf{x}_t\|^2}$
9: update: $\mathbf{v}_{t+1} = \mathbf{v}_t + \tau_t y_t \mathbf{x}_t$
10: **end for**

Algorithm 2 summarizes the online heterogeneous transfer learning 2 (OHT2). OHT2 holds a more polarized attitude to two hypotheses. Indication function $I(\cdot)$ represents whether the hypothesis makes a wrong prediction or not. The mistake made by a hypothesis will decrease its weight in the next trial.

Similar to OHT1 algorithms, we also have two variants OHT2-I and OHT2-II.

5 Theoretical analysis of OHT algorithms

In this section, we utilize $\mathbf{Hedge}(\beta)$ algorithm framework to help us understand OHT algorithms and derive mistake bound of OHT algorithms.

At the t-th trial of online learning problem, $\mathbf{Hedge}(\beta)$ algorithm framework synthesizes opinions from different experts based on a weight vector $\mathbf{weight}_t$. After the loss value of each expert has been received, the weight vector is updated using rule

$$weight_{t+1}^i = weight_t^i \cdot \beta^{loss_t^i}$$

where i is the index of expert, $\beta \in [0, 1]$ and $loss_t^i \in [0, 1]$.

In our proposed OHT algorithms, we refer to two hypotheses from two domains as the experts. We denote the loss value at the t-th trial of the source and targe domains by ℓ_t^s and ℓ_t respectively. For OHT1 algorithms, we take $\ell_t^s = (\Omega(h_t^s(\mathbf{x}_t)) - \Omega(y_t))^2$, $\ell_t = (\Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t))^2$, and $\beta = \exp\{-\eta\}$ where $\eta > 0$. For OHT2 algorithms we take $\ell_t^s = I(y_t h_t^s(\mathbf{x}_t) \leq 0)$, $\ell_t = I(y_t(\mathbf{v}_t \cdot$

Algorithm 2 Online Heterogeneous Transfer Algorithm 2 (OHT2)

Input: aggressiveness parameter $C > 0$
discount parameter $\alpha \in (0, 1)$
data in heterogeneous source domain

Initialize: $\mathbf{v} = \mathbf{0}$, $w_1^s \in (0, 1)$, $w_1 \in (0, 1)$, where $w_1^s + w_1 = 1$

1: **for** $t = 1$ to T **do**
2: receive instance: $\mathbf{x}_t \in \mathcal{X}$
3: normalize: $\theta_t^s = \frac{w_t^s}{w_t^s + w_t}, \theta_t = \frac{w_t}{w_t^s + w_t}$
4: predict: $\hat{y}_t = \text{sign}\left(\theta_t^s \,\text{sign}(h_t^s(\mathbf{x}_t)) + \theta_t \,\text{sign}(\mathbf{v}_t \cdot \mathbf{x}_t)\right)$
5: receive correct label: $y_t \in \mathcal{Y}$
6: compute:

$$w_{t+1}^s = w_{t+1}^s \alpha^{I(y_t h_t^s(\mathbf{x}_t) \le 0)}$$
$$w_{t+1} = w_{t+1} \alpha^{I(y_t(\mathbf{v}_t \cdot \mathbf{x}_t) \le 0)}$$

7: suffer loss: $\ell_t^* = \max\{0, 1 - y_t(\mathbf{v}_t \cdot \mathbf{x}_t)\}$
8: set: $\tau_t = \frac{\ell_t^*}{\|\mathbf{x}_t\|^2}$
9: update: $\mathbf{v}_{t+1} = \mathbf{v}_t + \tau_t y_t \mathbf{x}_t$
10: **end for**

$\mathbf{x}_t) \le 0)$, and $\beta = \alpha$. It is easy to validate that the weight updating strategies in OHT algorithms obey the rule of $\mathbf{Hedge}(\beta)$ algorithm framework.

Notice that the loss value ℓ_t^* in step 7 of OHT algorithms is calculated based on the hypothesis trained in the target domain independently. The calculation of ℓ_t^*, which comes from PA algorithm, aims to update the hypothesis in the target domain. Instead, we use ℓ_t^s and ℓ_t to represent the loss suffered by each hypothesis seperately, and adjust the weights of two hypotheses from different domains.

Before we present the mistake bound of our two OHT algorithms, we first give a proposition which can be used to derive mistake bound of both OHT algorithms.

Proposition 1. *Given loss $\ell_t^s \in [0, 1]$, $\ell_t \in [0, 1]$ and decay factor $\beta \in (0, 1)$, for any sequence of loss vectors $\{(\ell_t^s, \ell_t) | t = 1, 2, \cdots T\}$, we have*

$$\sum_{t=1}^{T} (\theta_t^s \ell_t^s + \theta_t \ell_t) \le \frac{1}{1 - \beta} \min\left(\Delta^s, \Delta\right)$$

where $\Delta^s = \ln(\frac{1}{w_1^s}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t^s$ *and* $\Delta = \ln(\frac{1}{w_1}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t$

The technique of the proof can be found in [**?**].

Remark. The physical meaning of the Proposition 1 is that the entire loss value, which sums up the combined loss at all T trials, is not much larger than the loss value made by the better single hypothesis. At the right side of the inequation, $\ln(\frac{1}{w_1^s})$ and $\ln(\frac{1}{w_1})$ measure the prior confidence we have on two hypotheses. If we have no reason to prefer any of them, we simply set $w_1^s = w_1 = \frac{1}{2}$. At this time, the upper bound only depends on the summation of loss made by two hypotheses seperately. On the other hand, if we accidently trust the better expert, we could get a tighter bound. For instance, if we choose to believe that $\sum_{t=1}^{T} \ell_t < \sum_{t=1}^{T} \ell_t^s$, and fortunately it is the truth at the same time, then we set $w_1 > w_1^s$. Consequently, we get a better bound than the one when we set $w_1^s = w_1$.

Based on Proposition 1, we can analyze the mistake bound of the proposed algorithms.

5.1 Mistake bound of OHT1

In OHT1 algorithms, $\ell_t^s = (\Omega(h_t^s(\mathbf{x}_t)) - \Omega(y_t))^2$, $\ell_t = (\Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t))^2$, and $\beta = \exp\{-\eta\}$. According to Proposition 1, we derive the mistake bound of OHT1 as follows.

Theorem 1. *Let M be the number of mistakes made by OHT1 algorithm, then we have*

$$M \leq \frac{4}{1-\beta} \min(\Delta^s, \Delta)$$

where $\Delta^s = \ln(\frac{1}{w_1^s}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t^s$ *and* $\Delta = \ln(\frac{1}{w_1}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t$.

Proof. Whenever there is a mistake made by classifier at t-th trial, we should have

$$|\theta_t^s \Omega(h_t^s(\mathbf{x}_t)) + \theta_t \Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t)| \geq \frac{1}{2}$$

Since $\theta_t^s + \theta_t = 1$, we have

$$\begin{aligned}
\frac{1}{2} \leq & |\theta_t^s \Omega(h_t^s(\mathbf{x}_t)) + \theta_t \Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t)| \\
= & |\theta_t^s \Omega(h_t^s(\mathbf{x}_t)) + \theta_t \Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \theta_t^s \Omega(y_t) - \theta_t \Omega(y_t)| \\
= & |\theta_t^s \big(\Omega(h_t^s(\mathbf{x}_t)) - \Omega(y_t)\big) + \theta_t \big(\Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t)\big)|
\end{aligned}$$

According to Jensen's inequality, we have

$$\begin{aligned}
\frac{1}{4} \leq & \Big(\theta_t^s \big(\Omega(h_t^s(\mathbf{x}_t)) - \Omega(y_t)\big) + \theta_t \big(\Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t)\big)\Big)^2 \\
\leq & \theta_t^s \Big(\Omega(h_t^s(\mathbf{x}_t)) - \Omega(y_t)\Big)^2 + \theta_t \Big(\Omega(\mathbf{v}_t \cdot \mathbf{x}_t) - \Omega(y_t)\Big)^2 \\
= & \theta_t^s \ell_t^s + \theta_t \ell_t
\end{aligned}$$

By adding the inequalities of all mistakes and combining with Proposition 1, we have

$$\frac{1}{4} M \leq \sum_{t=1}^{T} (\theta_t^s \ell_t^s + \theta_t \ell_t) \leq \frac{1}{1-\beta} \min\big(\Delta^s, \Delta\big)$$

where $\Delta^s = \ln(\frac{1}{w_1^s}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t^s$ and $\Delta = \ln(\frac{1}{w_1}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t$. The theorem follows immediately. □

5.2 Mistake bound of OHT2

In OHT2 algorithms, $\ell_t^s = I(y_t h_t^s(\mathbf{x}_t) \leq 0)$, $\ell_t = I(y_t(\mathbf{v}_t \cdot \mathbf{x}_t) \leq 0)$, and $\beta = \alpha$. The mistake bound of OHT2 is given by following theorem.

Theorem 2. *Let M be the number of mistakes made by OHT2 algorithm, then we have*

$$M \leq \frac{2}{1-\beta} \min(\Delta^s, \Delta)$$

where $\Delta^s = \ln(\frac{1}{w_1^s}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t^s$ *and* $\Delta = \ln(\frac{1}{w_1}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t$.

Proof. Whenever there is a mistake made by classifier at t-th trial, we should have

$$\begin{aligned}
\frac{1}{2} \leq & \theta_t^s I(y_t h_t^s(\mathbf{x}_t)) + \theta_t I(y_t(\mathbf{v}_t \cdot \mathbf{x}_t)) \\
= & \theta_t^s \ell_t^s + \theta_t \ell_t
\end{aligned}$$

By adding the inequalities of all mistakes and combining with Proposition 1, we have

$$\frac{1}{2} M \leq \sum_{t=1}^{T} (\theta_t^s \ell_t^s + \theta_t \ell_t) \leq \frac{1}{1-\beta} \min\big(\Delta^s, \Delta\big)$$

where $\Delta^s = \ln(\frac{1}{w_1^s}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t^s$ and $\Delta = \ln(\frac{1}{w_1}) + (\ln \frac{1}{\beta}) \sum_{t=1}^{T} \ell_t$. The theorem follows immediately. □

A recommended value is given by following theorem.

Theorem 3. *When*

$$\beta = \frac{\sqrt{T}}{\sqrt{T} + \sqrt{2\max(\ln(\frac{1}{w_1^s}), \ln(\frac{1}{w_1}))}}$$

we have

$$M \leq 2\min(\Lambda^s, \Lambda) + \sqrt{2T\max(\ln(\frac{1}{w_1^s}), \ln(\frac{1}{w_1}))}$$

where $\Lambda^s = \sum_{t=1}^{T} \ell_t^s + \ln(\frac{1}{w_1^s})$ *and* $\Lambda = \sum_{t=1}^{T} \ell_t + \ln(\frac{1}{w_1})$.

Theorem 3 can be proved by similar technique in [**?**].

6 Experimental results

In this section, we empirically evaluate the performance of proposed online heterogeneous transfer learning algorithms and classic online Passive-Aggressive algorithms, which consists of a original version PA and its two variations PA-I and PA-II. Encouraging results demonstrate that the proposed algorithms outperform baseline methods.

6.1 Dataset

Our experiments are conducted for image classification by leveraging knowledge from text data. We use NUS-WIDE dataset [**?**] to generate learning tasks. The NUS-WIDE dataset is extracted from Flickr. It includes 269,648 images and the associated tags, with a total number of 5,018 unique tags. An image instance is represented by a feature vector based on SIFT descriptions, and a text instance is represented by a feature vector based on tags. There are 81 ground-truth class labels in the dataset. We randomly selected 10 classes (bird, boat, car, flower, food, rock, sun, toy, tree) and built $C_{10}^2 = 45$ binary classification tasks.

We refer the images as the data in the target domain, and the tags as the text data in the heterogeneous source domain. Each binary classification task has 500 image instances in the target domain, 1,200 text instances in the heterogeneous source domain, and 1,500 co-occurred image-text pairs. In order to obtain stable results, we draw 100 times of random permutation of the image instances in the target domain and evaluate the performance of learning algorithms based on average rate of mistakes.

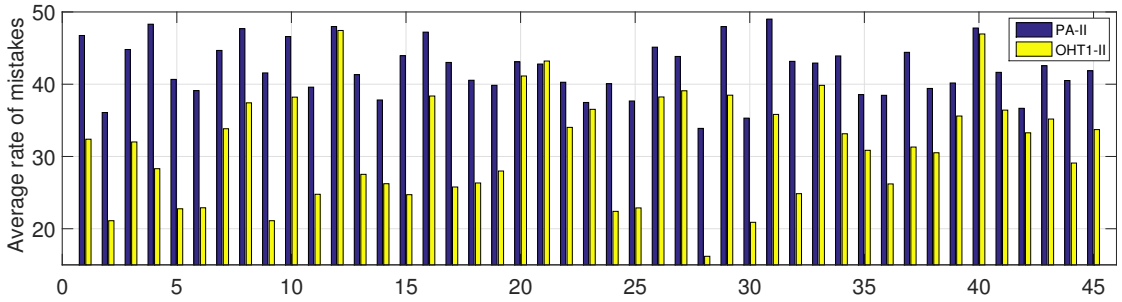
6.2 Baseline methods

We compare the proposed methods with Passive-Aggressive online learning algorithms. PA algorithms proposed in [**?**] does not exploit knowledge from the source domain. It deals with a traditional online learning problem. As each of PA and OHT algorithms have three versions, we conduct three sets of experiments separately. Each set of experiments compares two OHT methods against the Passive-Aggressive algorithm which use the same calculation method of learning rate τ.

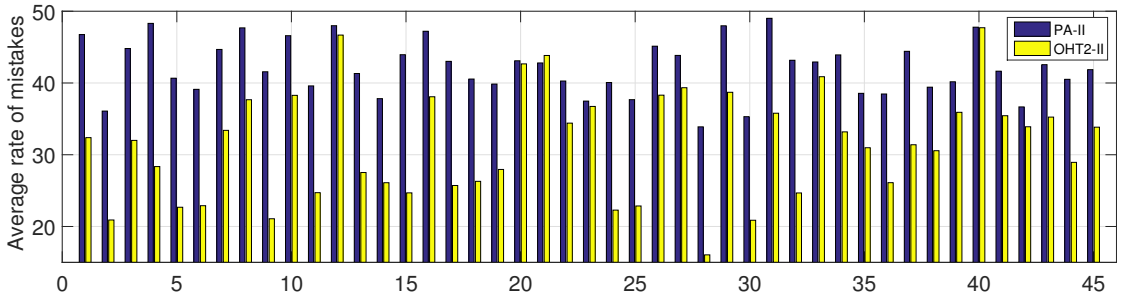
For fair comparison and simplicity, we adopt Gaussian kernel function on all the algorithms and tasks, and the kernel parameter $\sigma = 8$. The regularization parameter $C = 5$, and $\beta = \frac{\sqrt{T}}{\sqrt{T}+\sqrt{2\ln 4}}$ for OHT2 algorithm. In addition, we set the number of nearest neighbors to be considered $K = 100$. Sensitivity of parameters will be examined in subsequent sections.

6.3 Results and discussion

Figure **??** summarizes the mistake rates of all 45 binary classification tasks in the third set, which compares OHT1-II, OHT2-II and PA-II. The x-axis of the figure refers to the 45 tasks. We see that on most tasks, PA-II has the very high mistake rate, which prove the difficulty of image classification task without any auxiliary source information and the necessity of knowledge transfer. The observation that our proposed OHT methods in general outperform Passive-Aggressive algorithm validates



(a) PA-II vs. OHT1-II



(b) PA-II vs. OHT2-II

Figure 3: Average rate of mistakes on all 45 tasks

the effectivity of heterogeneous transfer learning. Similar experimental results are observed in other two sets. Because of the restricted space, we are not able to report them. In order to facilitate the description, we denote PA-II, OHT1-II and OHT2-II by PA, OHT1 and OHT2 respectively in the following discussion.

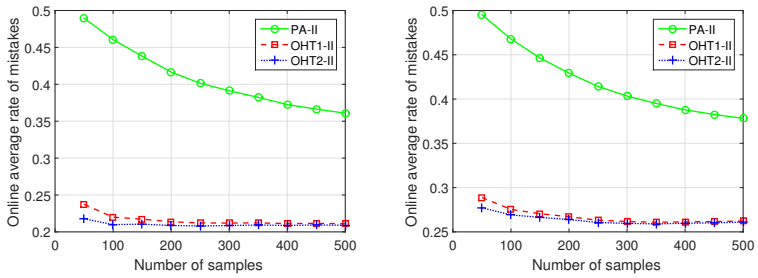
Figure **??** illustrates the dynamic process of several representative classification tasks. We observe that OHT algorithms usually achieve better performance at the beginning stage. On some tasks(e.g., 7, 16 and 33), the online mistake rates of all three algorithms decrease during the period, and OHT methods always obtain better performance than PA algorithm. These observations verifies that the OHT algorithms indeed transfer useful knowledge from the heterogeneous source domain to the target domain.

We also analyze the performance difference between PA and two OHT algorithms. Statistical significance against PA was assessed by paired t-test at 0.01 level. For each task, a win (or loss) is counted when OHT algorithm is significatly better (or worse) than PA algorithm over 100 runs. Otherwise, a tie is recorded. The win/tie/loss results is 44/0/1 for competition between OHT1 and PA, and 42/2/1 for competition between OHT2 and PA. This result validates that our OHT algorithms is statistically better than PA algorithm.

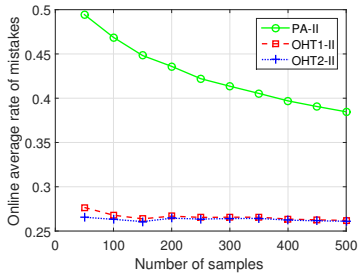
Besides, we utilize Cohen's d value to measure the improvement of our algorithms. Generally, $d > 0.8$ indicates a large promotion, and $0.2 < d < 0.8$ indicates a middle promotion. In our experiments, OHT1 algorithm achieves large improvement on 41 tasks and middle improvement on 3 tasks. For OHT2 algorithms, the numbers are 40 and 3. Combining the win/tie/loss results, we see that OHT1 is more stable than OHT2.

6.4 Parameters and running time

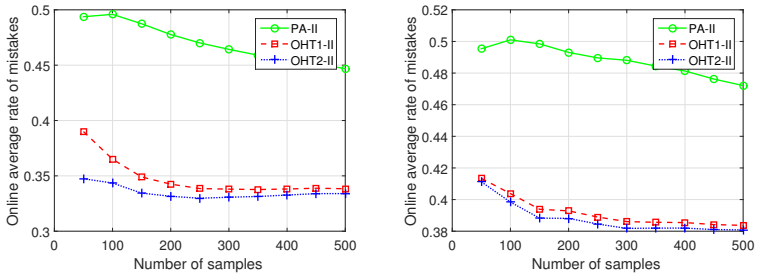
Parameters We investigate how different values of parameter K affect the mistake rates of the algorithms. Figure **??** shows the average mistake rates with varied values of parameter K over all 45 tasks. PA algorithm, whose performance is not related to the parameter K, provides a baseline rate of mistakes. We observe that the performance of the proposed methods consistently outperform



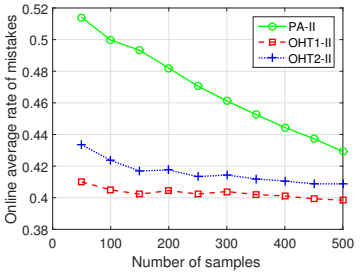
(a) Task 2 (b) Task 14



(c) Task 36



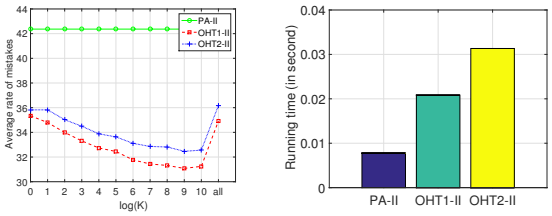
(d) Task 7 (e) Task 16



(f) Task 33

Figure 4: Online average rate of mistakes on example tasks

PA algorithm, which indicates that nearest neighbors in heterogeneous source domain do provide valuable advice for the classification task.



(a) (b)

Figure 5: (a) The average rate of mistakes under varying values of K. (b) The average running time of different algorithms when all instances in heterogeneous source are considered.

Running time All the experiments are run in Matlab over a Linux machine with 3.2 GHz CPU and 3.8 GB memory. Compared to PA algorithm without exploiting any information from the source domain, OHT algorithms are less efficient. Despite the extra computation for updating weights, the main reason of more running time for OHT algorithms is probably the searching process for the nearest neighbors. We can make use of global nearest neighbor approach in the heterogeneous source domain to get rid of overhead for searching nearest neighbors. Figure **??** shows the running time of different algorithms when all instances in the heterogeneous source domain are considered. We obtain generally comparable running time to PA, and at the same time, achieve a better performance than PA.

7 Conclusion

In this paper, we explore online heterogeneous transfer learning problem, whose objective is to tackle an online classification task in a target domain by leveraging knowledge extracted from a heterogeneous source domain. We construct a connection across the domains using co-occurrence data, and apply the ensemble strategy to train a classifier based on two hypotheses coming from different domains. We also offer the theoretical analysis of our algorithms. Experimental results show that the proposed algorithms outperform the baseline methods. In the future, we will consider the applications with other types of data, or the scenario which includes more than one source domains.