Transfer Learning From Multiple Source Domains via Consensus Regularization

Ping Luo, Fuzhen Zhuang, Hui Xiong, Yuhong Xiong, Qing He

HP Laboratories
HPL-2008-115

Keyword(s):
Classification, Transfer Learning, Consensus Regularization

Abstract:

Recent years have witnessed an increased interest in transfer learning. Despite the vast amount of research performed in this field, there are remaining challenges in applying the knowledge learnt from multiple source domains to a target domain. First, data from multiple source domains can be semantically related, but have different distributions. It is not clear how to exploit the distribution differences among multiple source domains to boost the learning performance in a target domain. Second, many real-world applications demand this transfer learning to be performed in a distributed manner. To meet these challenges, we propose a consensus regularization framework for transfer learning from multiple source domains to a target domain. In this framework, a local classifier is trained by considering both local data available in a source domain and the prediction consensus with the classifiers from other source domains. In addition, the training algorithm can be implemented in a distributed manner, in which all the source-domains are treated as slave nodes and the target domain is used as the master node. To combine the training results from multiple source domains, it only needs share some statistical data rather than the full contents of their labeled data. This can modestly relieve the privacy concerns and avoid the need to upload all data to a central location. Finally, our experimental results show the effectiveness of our consensus regularization learning.

External Posting Date: September 21, 2008 [Fulltext] Approved for External Publication
Internal Posting Date: September 21, 2008 [Fulltext]

To be published in ACM 17th Conference on Information and Knowledge Management, (CIKM'08), October 26-30, 2008, Napa Valley, California, USA

Transfer Learning From Multiple Source Domains via Consensus Regularization

Ping Luo[1], Fuzhen Zhuang [2,3], Hui Xiong[4], Yuhong Xiong[1], Qing He[2]

[1] HP Labs China, {ping.luo, yuhong.xiong}@hp.com
[2] The Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, {zhuangfz, heq}@ics.ict.ac.cn
[3] Graduate University of Chinese Academy of Sciences
[4] MSIS Department, Rutgers University, hxiong@rutgers.edu

ABSTRACT

Recent years have witnessed an increased interest in transfer learning. Despite the vast amount of research performed in this field, there are remaining challenges in applying the knowledge learnt from multiple source domains to a target domain. First, data from multiple source domains can be semantically related, but have different distributions. It is not clear how to exploit the distribution differences among multiple source domains to boost the learning performance in a target domain. Second, many real-world applications demand this transfer learning to be performed in a distributed manner. To meet these challenges, we propose a consensus regularization framework for transfer learning from multiple source domains to a target domain. In this framework, a local classifier is trained by considering both local data available in a source domain and the prediction consensus with the classifiers from other source domains. In addition, the training algorithm can be implemented in a distributed manner, in which all the source-domains are treated as slave nodes and the target domain is used as the master node. To combine the training results from multiple source domains, it only needs share some statistical data rather than the full contents of their labeled data. This can modestly relieve the privacy concerns and avoid the need to upload all data to a central location. Finally, our experimental results show the effectiveness of our consensus regularization learning.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Induction

General Terms

Algorithms, Experimentation, Performance.

Keywords

Classification, Transfer Learning, Consensus Regularization.

CIKM'08, October 26-30, 2008, Napa Valley, California, USA.

1. INTRODUCTION

Classification plays an important role in many real-world applications, such as Web-page classification and video concept detection. Traditional classification techniques have the assumption that training and test data are drawn from the same distribution, and thus fail to deal with the situation when the new unlabeled data are obtained from fast evolving, related but different information sources. This leads to the new learning problem about how to build the classifier when the distributions of training and test data are different.

Previous works in this area deal with the knowledge transfer from only one *source-domain* $\mathcal{D}_s$ to a *target-domain* $\mathcal{D}_t$ [1, 2, 3]. The Source-domain $\mathcal{D}_s$ owns labeled data, while the target-domain $\mathcal{D}_t$ contains plenty of unlabeled data. In this paper, we investigate the problem of transfer learning from multiple source-domains to a target domain. More precisely, we have m source-domains as $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$ and a target-domain $\mathcal{D}_t$ (Note that $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$ and $\mathcal{D}_t$ also represent their corresponding data sets throughout this paper). The labeled source-domains and the unlabeled target-domain may be geographically distributed. We assume that the class labels in $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$ and the labels to be predicted in $\mathcal{D}_t$ are drawn from the same class-label set $\mathcal{C}$. Furthermore, while these source-domains and the target-domain have different distributions, we assume that they are semantically related to each other in the sense that similar features would describe similar categories. Under this assumption, we aim to adapt the knowledge learnt from these m source-domains for classifying the data in the target domain.

Motivating Examples. We provide the following two application scenarios to motivate the above problem formulation. In the first example, assuming we have downloaded all the Web pages of a university, and we want to use text classification to find the course main pages. To do this, we create training data by manually labeling a collection of training course pages from different universities. However, different universities usually use different course page templates, in which the terms used can also be different. For example, the terms indicating the reading materials may include "Required Reading List", "Textbooks", "Reference" and so on. Thus, the data distribution of the main course Web pages from different universities are likely to be different. If we consider a university as a domain, the manually labeled Web pages come from multiple domains. Now, the goal is to find new course Web pages in a target university. This is a transfer learning problem discussed in this paper.

In the second example, let us consider the problem of **video concept detection**, which aims to generalize models built for detecting semantic concepts from multiple-source video data to other domains. Here, a domain is a TV channel, such as CCTV, CBS, CNN, and NBC. For instance, the TRECVID collection [4] is such a multi-domain video corpus which has news video from different TV channels. Due to the large data variance and the "semantic gap" between visual features and the semantic content, the problem of mismatch among the distributions of the multiple training domains and the test domain is particularly severe in multimedia area [5]. As shown in Figure 1, video shots of easily recognizable anchors from four different famous TV channels exhibit dissimilar visual features. As a result, concept classifiers trained from only one source-domain might perform poorly on the target domain.

Figure 1: Anchor Shots from Major Media including CCTV, CBS, CNN and NBC

The common ground of the above two applications is that the training data are from multiple related but different source domains. One can argue that, if we merge the multiple source-domains into one source-domain, this problem can be solved by existing transfer learning algorithms. However, the important information, such as the distribution differences among the source-domains, is lost during the merging process. This information is the key to understand the common nature of these source-domains. In addition, the training data from different source-domains might be distributed geographically, and it is difficult to put them into a central location due to the network bandwidth, central disk space, and data copyright considerations. Under this circumstance, the learning algorithm must be performed in a distributed and modest privacy-preserving manner.

In our formulation, given m source-domains: $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$, each of which is drawn from a distribution that might be different from the distribution of the target domain $\mathcal{D}_t$, a classifier h^l $(l = 1, \cdots, m)$ can be trained locally on each of these data sets $\mathcal{D}_s^l$, since $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$ are fully labeled. However, to classify $\mathcal{D}_t$, simply applying one h^l may not achieve good performance due to the mismatched distribution. Moreover, any classifier h^l trained on the limited data $\mathcal{D}_s^l$ may suffer from the high variance problem, and give different predictions on the data set $\mathcal{D}_t$. Since each instance in the target-domain represents a unique piece of ground truth, this disagreement of predictions on the target domain inspires a direction for optimization. That is, to achieve better prediction performance, one should leverage both the knowledge in the labeled data from the multiple source-domains and the unlabeled data in the target-domain.

In this paper we develop a *consensus regularization* framework that aims to transfer the knowledge from multiple source-domains to a target domain. This regularization framework is applied into the model of *Logistic Regression*. Note that it also can be implemented into other classification models. To the end, the following two challenges have been addressed.

1. How to make good use of the distribution differences among multiple source-domains to promote the prediction performance on the target-domain?
2. How to extend this consensus regularization based algorithm to a distributed algorithm while only sharing some statistical data of all source domains instead of revealing all full contents of labeled data?

For the first challenge, we propose the *maximum consensus regularization* method, which incorporates the unlabeled data from the target-domain into the learning process. For each source-domain, this consensus optimization framework will output one classifier, which is trained by considering both the local data and the prediction consensus with the other classifiers on the unlabeled target-domain data. In this mutually-affected manner, the resultant classifiers not only maintain the individuality of the corresponding source-domains, but also reveal the common nature of all the source-domains and the target-domain. For the second challenge, when multiple source-domains and the target-domain are distributed geographically, the learning algorithm is running in a distributed manner. This distributed system has a *master-slave* architecture, with the node for the target-domain being the master and the node for the source domains being the slaves. To achieve the same learning performance as a corresponding non-distributed algorithm, the training process takes multiple rounds to complete. In each round, only some statistical data (not full contents of labeled data) are shared between the slave nodes and the master node, and they cooperate in a synchronized fashion. Therefore, it is a distributed algorithm, which also addresses a modest degree of privacy concerns.

Finally, in the context of text classification, we validate the proposed method using real-world text data sets. The experimental results show that the consensus regularization learning method can effectively improve the learning performance in the target domain even if the source-domain data are geographically distributed.

2. PRELIMINARIES

In this section, we first introduce the notations used throughout this paper, and then present some preliminary concepts about logistic regression and consensus measuring.

2.1 Notations

In this paper, we use bold letters, such as $\mathbf{p}$ and $\mathbf{a}$, to represent vectors. Also, $\mathbf{p}_{(i)}$ indicates the i-th element of $\mathbf{p}$. Random variables are written in upper case, such as X and Y. Therefore, Bold upper case letters, such as $\mathbf{X}$ and $\mathbf{Y}$, represent vectors of random variables. Calligraphic letters, such as $\mathcal{A}$ and $\mathcal{D}$, represent sets. Finally, we use $\mathbb{R}$ to denote the set of real numbers and $\mathbb{R}_+$ to denote the set of nonnegative real numbers.

2.2 Logistic Regression

Logistic regression [6] is an approach to learning functions of $P(Y|\mathbf{X})$ in the case where Y is discrete-valued, and $\mathbf{X}$ is any vector containing discrete or continuous random variables. Logistic regression assumes a parametric form for the distribution $P(Y|\mathbf{X})$, then directly estimates its parameters from the training data. The parametric model assumed by logistic regression in the case where Y is Boolean is

$$P(y = \pm 1|\mathbf{x}; \mathbf{w}) = \sigma(y\mathbf{w}^T\mathbf{x}) = \frac{1}{1 + \exp(-y\mathbf{w}^T\mathbf{x})}, \quad (1)$$

where $\mathbf{w}$ is the parameter of the model. Under the principle of *Maximum A-Posteriori* (MAP), $\mathbf{w}$ is estimated under the Laplacian prior. Given a data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, we want to find the parameter $\mathbf{w}$ which maximizes:

$$\sum_{i=1}^{N} \log \frac{1}{1 + \exp(-y_i\mathbf{w}^T\mathbf{x}_i)} - \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}. \quad (2)$$

This criterion is a concave function of $\mathbf{w}$, so that the global solution can be obtained by methods of the non-linear numerical optimization. After $\mathbf{w}$ is estimated, Equation (1) can be used to compute the probabilities of an instance belonging to the positive and negative class.

2.3 Consensus Measuring

In the subsection, we first give the definition of *Shannon entropy* in a *probability distribution vector*, and then show how to measure the degree of consensus on the predictions that are made by a group of classifiers for an instance.

Definition 1 (Probability Distribution Vector). *$\boldsymbol{p} \in \mathbb{R}_+^d$ is a probability distribution vector if and only if $\sum_{i=1}^d \boldsymbol{p}_{(i)} = 1$. Each entry $\boldsymbol{p}_{(i)}$ of this vector represents the probability that this instance belongs to class i.*

Definition 2 (Shannon Entropy). *Assuming $\boldsymbol{p} \in \mathbb{R}_+^d$ is a probability distribution vector, then the Shannon entropy in $\boldsymbol{p}$ is defined as*

$$\mathbf{E}(\boldsymbol{p}) = \sum_{i=1}^{d} \boldsymbol{p}_{(i)} \log \frac{1}{\boldsymbol{p}_{(i)}}. \quad (3)$$

Given a group of m classifiers $\mathcal{H} = \{h^l\}_{l=1}^m$, each of which outputs a probability distribution vector $\mathbf{p}^l$ for an instance $\mathbf{x}$, the average probability distribution vector can be computed as:

$$\overline{\mathbf{p}} = \frac{\sum_{l=1}^m \mathbf{p}^l}{m}. \quad (4)$$

Then, using the Shannon entropy, we can measure the degree of consensus in these prediction results as shown in the examples of Table 1. For 3-class classification problem, Table 1 records the probability distribution vectors of $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ predicted by the classifiers h^1, h^2 and h^3 respectively, and their corresponding average probability distribution vectors. For the first instance $\mathbf{x}_1$, all the classifiers reach a perfect consensus that it belongs to class 1 with 100% probability. Therefore, the degree of consensus on these results reaches its maximum, while the entropy $\mathbf{E}(1, 0, 0)$ of the average distribution vector reaches its minimum for any 3-entry distribution vectors. On the other hand, for the third instance $\mathbf{x}_3$, the three classifiers predict that it belongs to class 1, 2 and 3 respectively with 100% probability. Thus, these prediction results totally disagree with each other, and their degree of consensus reaches its minimum. However, the entropy $\mathbf{E}(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ of the average distribution vector reaches its maximum. Therefore, the negative of the entropy in the average probability distribution vector can be the consensus measure for the different prediction results. Formally, the definition of the entropy based consensus measure is:

Definition 3 (Entropy based Consensus Measure). *Given m probability distribution vectors $\boldsymbol{p}^1, \cdots, \boldsymbol{p}^m$, the consensus measure for these vectors is defined as*

$$\mathbf{C}_e(\boldsymbol{p}^1, \cdots, \boldsymbol{p}^m) = -\mathbf{E}(\overline{\boldsymbol{p}}), \quad (5)$$

where $\mathbf{E}$ is the Shannon entropy in Definition 2 and $\overline{\boldsymbol{p}}$ is the average of these vectors defined by (4).

Since we only consider the relative magnitude of two consensus measures, it is acceptable that the value of this consensus measure is negative. Thus, by this definition, the consensus degree for the prediction results of the second instance $\mathbf{x}_2$ is $-\mathbf{E}(0.7, 0.2, 0.1)$.

Due to the computing complexity in the entropy, for 2-entry probability distribution vectors, this consensus measure can be simplified as:

$$\begin{aligned} \mathbf{C}_s(\mathbf{p}^1, \cdots, \mathbf{p}^m) &= (\overline{\mathbf{p}}_{(1)} - \overline{\mathbf{p}}_{(2)})^2 \\ &= (\overline{\mathbf{p}}_{(1)} - (1 - \overline{\mathbf{p}}_{(1)}))^2 = (2\overline{\mathbf{p}}_{(1)} - 1)^2. \end{aligned} \quad (6)$$

It is clear that, when comparing the relative magnitude of two degrees of consensus, $\mathbf{C}_e$ and $\mathbf{C}_s$ are equivalent for 2-entry probability distribution vectors in the sense that they always give the same answer.

3. PROBLEM FORMULATION AND CONSENSUS REGULARIZATION

In this section, we first formulate the problem of transfer learning from multiple domains and then describe the principle of consensus regularization. Next, we analyze why and when this consensus regularization works for this problem formulation. Finally, we show how to adapt this principle into the model of logistic regression.

3.1 Problem Formulation and Principle of Consensus Regularization

Let $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$ be m (m > 1) source-domains of labeled data , and the labeled data set from the l-th source-domain is represented by $\mathcal{D}_s^l = \{(\mathbf{x}_i^l, y_i^l)\}|_{i=1}^{n^l}$, where y_i^l is the label of $\mathbf{x}_i^l$ and n^l is the number of data in this domain[1]. The unlabeled target-domain is denoted by $\mathcal{D}_t = \{(\mathbf{x}_i)\}|_{i=1}^n$, where n is the number of data objects in the target-domain. Under the assumption that the distributions of $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m, \mathcal{D}_t$ are different but closely related, we aim to train the classification models on these labeled source-domains to accurately classify the unlabeled data in the target-domain.

If we train m classifiers $h^1, \cdots, h^m$ locally, each of which is based only on the data from one source-domain data, the ideal situation is that these m classifiers make a perfect consensus that they predict an instance from the target-domain to be its ground truth with 100% confidence. However, since the distributions of $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m, \mathcal{D}_t$ are different, these initial m classifiers usually disagree with each other to some

[1]In the following, the upper index of a letter denotes the index of source-domain, while the lower index of a letter, if existing, denotes the index of the data in the data set.

Table 1: The Entropy and Consensus of Probability Distribution Vectors

instance	$\mathbf{p}^1$ by h^1	$\mathbf{p}^2$ by h^2	$\mathbf{p}^3$ by h^3	$\overline{\mathbf{p}}$	Entropy	Consensus
$\mathbf{x}_1$	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)	(1, 0, 0)	0	0
$\mathbf{x}_2$	(0.7, 0.25, 0.05)	(0.8, 0.1, 0.1)	(0.6, 0.25, 0.15)	(0.7, 0.2, 0.1)	$(\frac{7}{10}\log\frac{10}{7}+\frac{1}{5}\log 5+\frac{1}{10}\log 10)$	$-(\frac{7}{10}\log\frac{10}{7}+\frac{1}{5}\log 5+\frac{1}{10}\log 10)$
$\mathbf{x}_3$	(1, 0, 0)	(0, 1, 0)	(0, 0, 1)	$(\frac{1}{3},\frac{1}{3},\frac{1}{3})$	log 3	-log 3

degree on the prediction results of a certain instance. Thus, there is room to further maximize the consensus of these models on the prediction results of the data in the target-domain. Therefore, we can incorporate this consensus measure into the standard framework of supervised learning as follows. This adapted supervised learning framework with consensus regularization will output m models $h^1, \cdots, h^m$, which maximize the following equation:

$$\sum_{l=1}^{m} P(h^l|\mathcal{D}_s^l) + \theta \cdot Consensus(h^1, \cdots, h^m|\mathcal{D}_t), \tag{7}$$

where $P(h^l|\mathcal{D}_s^l)$ is the probability of the hypotheses h^l given the observed data set $\mathcal{D}_s^l$, and $Consensus(h^1, \cdots, h^m|\mathcal{D}_t)$ is the consensus measure of these m models $h^1, \cdots, h^m$ on the prediction results of the data in the target-domain $\mathcal{D}_t$.

In the first term of (7), each model h^l is applied to its local source-domain, while the second term in (7) is used as a bridge to link all these models, and realizes a mutual coupling optimization. In this way, each of these resultant models not only keeps the individuality of the corresponding local source-domain, but also reveals the common nature of the target-domain. Thus, this regularization framework maximizes not only the posteriori in each source-domain, but also the consensus degree of these models.

Given a source-domain data set $\mathcal{D}_s^l = \{(\mathbf{x}_i^l, y_i^l)\}|_{i=1}^{n^l}$ of *independent and identically-distributed* (IID) observations, the maximization of $P(h^l|\mathcal{D}_s^l)$ in the first term of (7) can be expanded further as follows:

$$\begin{aligned}&\max P(h^l|\mathcal{D}_s^l) = \max \frac{P(\mathcal{D}_s^l|h^l)P(h^l)}{P(\mathcal{D}_s^l)} = \max P(\mathcal{D}_s^l|h^l)P(h^l)\\ &= \max P(h^l)\cdot\prod_{i=1}^{n^l} P(y_i^l|\mathbf{x}_i;h^l) = \max(\log P(h^l) + \sum_{i=1}^{n^l}\log P(y_i^l|\mathbf{x}_i;h^l)).\end{aligned} \tag{8}$$

As to $Consensus(h^1, \cdots, h^m|\mathcal{D}_t)$, it is defined as the sum of the consensus measures of these m models on all the data in $\mathcal{D}_t$ and is shown as follows:

$$Consensus(h^1, \cdots, h^m|\mathcal{D}_t) = \sum_{i=1}^{n} \mathbf{C}_e(\mathbf{p}_i^1, \cdots, \mathbf{p}_i^m), \tag{9}$$

where $\mathbf{C}_e$ is the consensus measure in Definition 3, and $\mathbf{p}_i^l$ is the probability distribution vector predicted by the l-th model h^l for the i-th instance in the target-domain $\mathcal{D}_t$. Actually, this consensus measure has two sides of effect. One is to promote the degree of agreement on all the models. The other is to minimize the entropy of the prediction results on unlabeled data.

3.2 Why Consensus Regularization

In this subsection we will theoretically show that maximizing agreement between any two individual classifiers could lead to the performance improvement of the individual classifiers.

In this study we focus on binary classification problems with the labels 1 and -1. We can train m models $h^1, \cdots, h^m$ for m source-domains. Let Y be the target label, and the disagreement of any two individual models be $P(h^i \neq h^j)$ $(i, j \in \{1, \cdots, m\}, i \neq j)$. In the following the number of the individual models is set to 3 for convenience. Note that the results in this subsection can be extended to any number of individual models similarly. We also have the following two definitions.

Definition 4 (non-trivial classifier). *If a classifier h satisfies the condition*

$$P(h = u|Y = u) > P(h = \overline{u}|Y = u),$$

where $u \in \{-1, 1\}$ and $\overline{u}$ is the complement of u. Then we call classifier h is a non-trivial classifier.

In other words, we can restate the non-trivial condition as

$$P(h = u|Y = u) > 1/2 \text{ or } P(h \neq u|Y = u) \leq 1/2.$$

Definition 5 (conditional independent classifiers). *The conditional independence of models h^1, h^2, h^3 is shown as follows,*

$$P(h^1 = u|h^2 = v, Y = y) = P(h^1 = u|Y = y), \tag{10}$$

$$P(h^1 = u|h^2 = v, h^3 = w, Y = y) = P(h^1 = u|Y = y), \tag{11}$$

where $u, v, w, y \in \{-1, 1\}$.

According to the assumption of non-trivial and conditional independent classifiers, We obtain the following theorem.

Theorem 1. *If the conditions that conditional independent assumptions are satisfied, it holds that the disagreement upper bounds the misclassification error for nontrivial classifier.*

Proof. The classification error of h^1 is

$$\begin{aligned}&P(h^1 \neq Y)\\ &= P(h^1 = 1, Y = -1) + P(h^1 = -1, Y = 1)\\ &= P(h^1 = 1, h^2 = -1, Y = -1)\\ &\quad + P(h^1 = 1, h^2 = 1, Y = -1)\\ &\quad + P(h^1 = -1, h^2 = -1, Y = 1)\\ &\quad + P(h^1 = -1, h^2 = 1, Y = 1),\end{aligned}$$

and the disagreement between h^1 and h^2 is

$$\begin{aligned}&P(h^1 \neq h^2)\\ &= P(h^1 = 1, h^2 = -1) + P(h^1 = -1, h^2 = 1)\\ &= P(h^1 = 1, h^2 = -1, Y = -1)\\ &\quad + P(h^1 = 1, h^2 = -1, Y = 1)\\ &\quad + P(h^1 = -1, h^2 = 1, Y = -1)\\ &\quad + P(h^1 = -1, h^2 = 1, Y = 1).\end{aligned}$$

To validate that $P(h^1 \neq Y) \leq P(h^1 \neq h^2)$, we only have to

proof the following inequation,

$$\begin{aligned} &P(h^1 = 1, h^2 = 1, Y = -1) \\ &\quad + P(h^1 = -1, h^2 = -1, Y = 1) \\ &\leq P(h^1 = 1, h^2 = -1, Y = 1) \\ &\quad + P(h^1 = -1, h^2 = 1, Y = -1). \end{aligned} \tag{12}$$

According to equation (10) and the Bayes Principle, Inequation (12) can also be written as follows,

$$\begin{aligned} &P(h^1 = 1|Y = -1)P(h^2 = 1, Y = -1) \\ &\quad + P(h^1 = -1|Y = 1)P(h^2 = -1, Y = 1) \\ &\leq P(h^1 = 1|Y = 1)P(h^2 = -1, Y = 1) \\ &\quad + P(h^1 = -1|Y = -1)P(h^2 = 1, Y = -1). \end{aligned} \tag{13}$$

From Definition 4, the following inequations (14), (15) hold,

$$P(h^1 = 1|Y = -1) \leq P(h^1 = -1|Y = -1), \tag{14}$$

$$P(h^1 = -1|Y = 1) \leq P(h^1 = 1|Y = 1). \tag{15}$$

Therefore, Inequation (13) holds.

Finally we obtain that the disagreement upper bounds the misclassification error as

$$P(h^1 \neq Y) \leq P(h^1 \neq h^2). \tag{16}$$

Similarly, we can prove that the following inequations also hold,

$$\begin{aligned} &P(h^1 \neq Y) \leq P(h^1 \neq h^3), \\ &P(h^2 \neq Y) \leq P(h^2 \neq h^1), \\ &P(h^2 \neq Y) \leq P(h^2 \neq h^3), \\ &P(h^3 \neq Y) \leq P(h^3 \neq h^1), \\ &P(h^3 \neq Y) \leq P(h^3 \neq h^2). \end{aligned}$$

□

Theorem 1 shows that the disagreement upper bounds the misclassification error for nontrivial classifier. Thus, minimizing the disagreement means to decrease the classification error.

3.3 Implementation of Consensus Regularization by Logistic Regression

Here, we introduce how to adapt and integrate the principle of consensus regularization into the model of logistic regression.

According to the problem formulation in Section 3.1, this consensus regularization framework outputs m logistic models $\mathbf{w}^1, \cdots, \mathbf{w}^m$, which maximize:

$$\begin{aligned} g_e(\mathbf{w}^1, \cdots, \mathbf{w}^m) &= \sum_{l=1}^{m} (\sum_{i=1}^{n^l} \log P(y_i^l|\mathbf{x}_i^l; \mathbf{w}^l) - \frac{\lambda^l}{2} {\mathbf{w}^l}^T \mathbf{w}^l) \\ &- \theta \cdot \sum_{i=1}^{n} \mathbf{E}(\frac{\sum_{l=1}^{m} P(y = -1|\mathbf{x}_i; \mathbf{w}^l)}{m}, \frac{\sum_{l=1}^{m} P(y = 1|\mathbf{x}_i; \mathbf{w}^l)}{m}), \end{aligned} \tag{17}$$

where the conditional probability P is the logistic function defined in (1), and $\mathbf{E}$ is the Shannon entropy. Note that this regularization framework works for multi-class problems.

For the 2-class classification problem, the entropy based consensus measure can be substituted with the equivalent form $\mathbf{C}_s$, defined in Equation (6). Thus, the new objective function is

$$\begin{aligned} g_s(\mathbf{w}^1, \cdots, \mathbf{w}^m) = &\sum_{l=1}^{m} (\sum_{i=1}^{n^l} \log P(y_i^l|\mathbf{x}_i^l; \mathbf{w}^l) - \frac{\lambda^l}{2} {\mathbf{w}^l}^T \mathbf{w}^l) + \\ &\theta \cdot \sum_{i=1}^{n} (2 \frac{\sum_{l=1}^{m} P(y = 1|\mathbf{x}_i; \mathbf{w}^l)}{m} - 1)^2, \end{aligned} \tag{18}$$

where the conditional probability P is the logistic function defined in Equation (1).

To simplify the discussion, in this paper we only describe this regularization framework in Equation (18) for 2-class classification problems, but it can be extended to multi-class problems using the framework in Equation (17). Thus, The partial differential of the objective g_s is

$$\nabla_{\mathbf{w}^l}(g_s) = \frac{\partial g_s}{\partial \mathbf{w}^l} = \nabla_{sn}^l(\mathbf{w}^l, \mathcal{D}_s^l) + \nabla_{mn}^l(\mathbf{w}^1, \cdots, \mathbf{w}^m, \mathcal{D}_t), \tag{19}$$

where the function σ is defined in (1), and

$$\nabla_{sn}^l(\mathbf{w}^l, \mathcal{D}_s^l) = \frac{\partial g_s}{\partial \mathbf{w}^l} = \sum_{i=1}^{n^l} (1 - \sigma(y_i^l {\mathbf{w}^l}^T)) y_i^l \mathbf{x}_i^l - \lambda^l \mathbf{w}^l, \tag{20}$$

$$\begin{aligned} &\nabla_{mn}^l(\mathbf{w}^1, \cdots, \mathbf{w}^m, \mathcal{D}_t) \\ =&\frac{4\theta}{m^2} \sum_{i=1}^{n} (2 \sum_{k=1}^{m} \sigma({\mathbf{w}^k}^T \mathbf{x}_i) - m)(1 - \sigma({\mathbf{w}^l}^T \mathbf{x}_i)) \sigma({\mathbf{w}^l}^T \mathbf{x}_i) \mathbf{x}_i. \end{aligned} \tag{21}$$

Though the objective function in Equation (18) is neither concave nor convex, for given initial values, the local optimization solution can also be obtained by any non-linear optimization technique. In this study, we adopt the *conjugate gradient* method [7] as the optimization technique (the reason why we adopt conjugate gradient is described in the experimental section), and the initial models are set to the ones trained on each local source-domain separately. The pseudo-code of our method is shown in Algorithm 1. To solve the sub-problem in Step 4 of Algorithm 1, any optimization technique can be used. In our implementation the function *fminunc* provided by Matlab is adopted for Step 4.

4. CONSENSUS REGULARIZATION IN A DISTRIBUTED MANNER

In this section, we investigate how to extend this centralized consensus regularization method into a distributed learning algorithm, which can work in the situation that the source-domains $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$ and the target domain $\mathcal{D}_t$ are all geographically separate distributed. In this distributed setting, the data nodes containing the source-domain data are used as *slave nodes*, denoted by $sn^1, \cdots, sn^m$, and the data node containing the target-domain data is used as the *master node*, denoted by mn.

Let us first revisit the partial differential of the objective g_s in (19), which consists of two parts. It is clear that the computation of the first term $\nabla_{sn}^l(\mathbf{w}^l, \mathcal{D}_s^l)$ needs only the local model $\mathbf{w}^l$ and the data set $\mathcal{D}_s^l$. Thus, it can be computed locally. The computation of the second term $\nabla_{mn}^l(\mathbf{w}^1, \cdots, \mathbf{w}^m, \mathcal{D}_t)$ involves all the models $\mathbf{w}^1, \cdots, \mathbf{w}^m$ and the target-domain data set $\mathcal{D}_t$. Therefore, if the slave nodes sn^l $(l = 1, \cdots, m)$ sends $\mathbf{w}^l$ and ∇_{sn}^l to the master node mn, the master node can compute $\nabla_{\mathbf{w}^l}(g_s)$ by $\nabla_{\mathbf{w}^l}(g_s) = \nabla_{sn}^l + \nabla_{mn}^l$ in a straightforward manner.

As a result, if each round of the optimization process performs this synchronous communication of the statistic data between the slave nodes and the master node, the gradient

Algorithm 1 Centralized Version of Consensus Regularization by Conjugate Gradient Ascent

Input: The labeled data sets $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$, the unlabeled data set $\mathcal{D}_t$, the element matrix $Q \in \mathbb{R}^{k\times k}$ where $k = |\mathbf{x}|$ is the dimension of the data in the source-domain, the error threshold $\varepsilon > 0$, and the maximum iterating number max.
Output: m classifiers $\mathbf{w}^1, \cdots, \mathbf{w}^m$.

1. Each source-domain calculates the initial $\mathbf{w}_0^l$ by logistic regression based on its local data set $\mathcal{D}_s^l$
2. $k := 0$.
3. For $l = 1, \cdots, m$, compute the gradients $\nabla_{\mathbf{w}_k^l}(g_s)$ by (19), and set the searching directions as
$$\begin{cases} \mathbf{d}_0^l = \nabla_{\mathbf{w}_0^l}(g_s) \\ \mathbf{d}_{k+1}^l = \nabla_{\mathbf{w}_{k+1}^l}(g_s) + \alpha_k \mathbf{d}_k^l \\ \alpha_k = -\nabla^T_{\mathbf{w}_{k+1}^l}(g_s) Q \mathbf{d}_k^l / {\mathbf{d}_k^l}^T Q \mathbf{d}_k^l \end{cases} \quad (22)$$
If $\sum_{l=1}^m \|\nabla_{\mathbf{w}_k^l}(g_s)\| < \varepsilon$, then turn to Step 6.
4. Compute the best searching step $\gamma \geq 0$, which maximizes
$$u_s(\gamma) = g_s(\mathbf{w}_k^1 + \gamma \mathbf{d}_k^1, \cdots, \mathbf{w}_k^m + \gamma \mathbf{d}_k^m). \quad (23)$$
Then for $l = 1, \cdots, m$, compute $\mathbf{w}_{k+1}^l$ by
$$\mathbf{w}_{k+1}^l = \mathbf{w}_k^l + \gamma \mathbf{d}_k^l. \quad (24)$$
5. $k := k + 1$. If $k \leq max$, then turn to Step 3.
6. Output $\mathbf{w}_k^1, \cdots, \mathbf{w}_k^m$.

$\nabla_{\mathbf{w}^l}(g_s)$ can be computed accurately. However, the maximization in the Step 4 of Algorithm 1 involves all the data from the source-domains and the target domain, which is hard to be solved distributively. In order to extend Algorithm 1 to a distributed version, the searching step γ in it can be set to a constant. Then, the method of distributed consensus regularization is described in Algorithm 2, which is an approximation of Algorithm 1.

In each round of Algorithm 2, each slave node sn^l sends a vector ∇_{sn}^l to the master node (in the first round the slave node should also send the initial model to the master node), and the master node sends back the updated model. Therefore, if this process terminates after k iterations, the total communication overhead will be
$$(2k+1)\sum_{l=1}^{m} |\mathbf{w}^l|. \quad (25)$$

Note that this distributed process communicates only some statistic values, such as ∇_{sn}^l $(l = 1, \cdots, m)$ and the classification models, without sending the raw source-domain data. Therefore, this also can modestly alleviate the privacy-concerns.

5. EXPERIMENTAL EVALUATION

The experiments performed in this section evaluate the performance of the proposed methods. In the experiments, we focus on the problem of binary classification, however, it is straightforward to extend the proposed methods for multi-class classification. Additionally, in this study the number of the source-domains for transfer learning is set to 3.

Algorithm 2 The Distributed Version of Consensus Regularization by Conjugate Gradient

Input: The labeled data sets $\mathcal{D}_s^1, \cdots, \mathcal{D}_s^m$ on the separated slave nodes $sn^1, \cdots, sn_m$ respectively, the unlabeled data set $\mathcal{D}_t$ on the master node mn, the error threshold $\varepsilon > 0$, the maximum iterating number max, and the step constant γ.
Output: m classifiers $\mathbf{w}^1, \cdots, \mathbf{w}^m$.

1. Each slave node sn^l $(l = 1, \cdots, m)$ calculates the initial $\mathbf{w}_0^l$ by logistic regression based on its local data set $\mathcal{D}_s^l$. Then they send this initial model $\mathbf{w}_0^l$ and the value of $\nabla_{sn}^l(\mathbf{w}_0^l, \mathcal{D}_s^l)$ $(l = 1, \cdots, m)$ to the master node mn.
2. $k := 0$.
3. The master node computes the gradients $\nabla_{\mathbf{w}_k^l}(g_s)$ $(l = 1, \cdots, m)$ for each model by (19), and sets the searching direction $\mathbf{d}_k^l$ as (22). If $\sum_{l=1}^m \|\nabla_{\mathbf{w}_k^l}(g_s)\| < \varepsilon$, then turn to Step 6; Otherwise, using the input constant γ, compute $\mathbf{w}_{k+1}^l$ as (24) for $l = 1, \cdots, m$.
4. The master node sends $\mathbf{w}_{k+1}^l$ $(l = 1, \cdots, m)$ to each slave node. Then each slave node computes $\nabla_{sn}^l(\mathbf{w}_{k+1}^l, \mathcal{D}_s^l)$ and sends it back to the master nodes.
5. $k := k + 1$. If $k \leq max$, then turn to Step 3.
6. Output $\mathbf{w}_k^1, \cdots, \mathbf{w}_k^m$.

5.1 Data Preparation

We follow the data preparation method in [2] to construct the problems of transfer learning from multiple source-domains, which will be detailed following. Since the public data collections are not originally designed for transfer learning from multiple source-domains, we need to make some modifications on them to fit the problem settings. It requires that each of these data sets has at least a two-level hierarchical structure. In this paper, we assume $\mathcal{A}$ and $\mathcal{B}$ are two root categories in a data set, and $\mathcal{A}_1, \cdots, \mathcal{A}_4$ and $\mathcal{B}_1, \cdots, \mathcal{B}_4$ are the four sub-level categories of $\mathcal{A}$ and $\mathcal{B}$ respectively. These sub-level categories are used for the three source-domains and one target-domain. Now we construct the training and test data as follows. For $i = 1, \cdots, 4$, let $\mathcal{A}_{a_i}$ and $\mathcal{B}_{a_i}$ be the positive and negative instances in the i-th domain $\mathcal{D}_{a_i} = \mathcal{A}_{a_i} \cup \mathcal{B}_{a_i}$ respectively; and $\mathcal{A}_{a_i}$ and $\mathcal{B}_{a_i}$ appear once and only once in these domains. In this way, the positive (negative) data from different domains are similar since they belong to the same top category $\mathcal{A}$ ($\mathcal{B}$), and the positive (negative) data from different domains are still different since they belong to different sub-categories. Thus, these four domains own different but similar data distributions. We can then select any one of these four domains as the target domain, and the other three domains as the source-domains. Therefore, given $\mathcal{A}_1, \cdots, \mathcal{A}_4$ and $\mathcal{B}_1, \cdots, \mathcal{B}_4$, we can construct 96 $(4 \cdot P_4^4)$ instances of 3-source-domain transfer learning problems.

20 Newsgroup[2] is one of the public available data collec-

[2]http://people.csail.mit.edu/jrennie/20Newsgroups/

tion, which satisfies the problem requirement that the top category contains at least four sub-categories. In this section, we list the evaluation results on the data sets of *sci* and *talk*, which are regarded as the top categories, denoted by $\mathcal{A}$ and $\mathcal{B}$ respectively. The four sub-categories in *sci* are *sci.crypt*, *sci.electronics*, *sci.med* and *sci.space*, denoted by $\mathcal{A}_1, \cdots, \mathcal{A}_4$ respectively. The four sub-categories in *talk* are *talk.politics.guns*, *talk.politics.mideast*, *talk.politics.misc* and *talk.religion.misc*, denoted by $\mathcal{B}_1, \cdots, \mathcal{B}_4$ respectively. The threshold of *Document Frequency* with the value of 5 is used to select the features. Then, these corresponding 96 problem instances are evaluated by the benchmark classification methods in Section 5.2.

We also evaluate our algorithm on other text data sets, including *comp* vs. *talk* and *comp* vs. *sci* etc. Furthermore, we construct and evaluate the transfer learning problems of image classification. These results are quite similar to those in this section. Due to the space limitations, we omit these detail results.

5.2 Benchmark Classification Methods

Under the assumption that the source-domains and the target domain are geographically separate distributed, we introduce the following benchmark classification methods for comparison. They can be grouped into two types: distributed and centralized classification algorithms.

Distributed Approach: in this approach, the algorithm is implemented in a distributed manner. The simplest distributed approach is Distributed Ensemble (DE), where a classifier is trained based on the local data on each source-domain. The other distributed approach is Distributed Consensus Regularization (DCR) described in Algorithm 2. In both DE and DCR, the prediction is made by majority voting on the probability distribution vectors of the resultant classifiers.

Centralized Approach: in this approach, all the data from the source-domains and the target-domain are accumulated and processed on a central node. In this case, the simplest method is Centralized Training (CT) which trains a global classifier on all the data. Meanwhile, if all the data from the source-domains are put together as one labeled data set, Centralized Consensus Regularization (CCR) in Algorithm 1 with $m = 1$, denoted by CCR_1, can be used. The transfer learning method CoCC [2] and the semi-supervised techniques TSVM [8] as well as SGT [9] can also be applied to this situation. On the other hand, in order to explicitly consider that the centralized data are from three different source-domains, CCR with $m = 3$, denoted by CCR_3, is also adopted.

In summary, our proposed method DCR, CCR_1 and CCR_3 are compared with DE, CT, CoCC, TSVM and SGT. Note that when the parameter θ in the objective function (Equation 18) is set to 0, DE is equivalent to DCR, and CT is equivalent to CCR_1. Also, DCR with a small step constant achieves the same accuracy performance as CCR_3 at the cost of some communication overhead. Therefore, these two algorithms DCR and CCR_3 are denoted by CCR_3 only.

Details of Implementation: Some initial experiments show that several popular non-linear optimization techniques output similar models for the proposed optimization problem and conjugate gradient is the fastest one. Therefore, we use conjugate gradient in this paper. After some preliminary test, we find that λ^l is not very sensitive in the value range [10,300], so the λ^l in the objective function (Equation 18) is set to 145 (for $l = 1, \cdots, m$). And the value range of θ is [0, 0.25]. In the algorithms of consensus regularization, the maximal iterating number max is set to 200, and the error threshold ε is set to 0.1. Before conducting the optimization for consensus regularization, logistic regression[3] is performed on each source-domain to obtain the initial values of the model for further optimization. The parameters of CoCC, TSVM and SGT are the same as those in [2].

5.3 Evaluation Metrics

The performance of the comparison methods is evaluated by accuracy. Let c be the function which maps each instance to its true class label, and f be the function which maps each instance to its prediction label given by the classifier. Accuracy is defined as

$$a = \frac{|\{\mathbf{d}|\mathbf{d} \in \mathcal{D}_t \wedge c(\mathbf{d}) = f(\mathbf{d})\}|}{|\mathcal{D}_t|}. \tag{26}$$

We also measure the consensus degree of multiple classifiers on the target-domain as follows. Let h be the function which maps each instance to the probability distribution vector predicted by the classifier. This consensus degree of m classifiers $h_1, \cdots, h_m$ is defined as

$$c = \frac{\sum_{\mathbf{d} \in \mathcal{D}_t} \sqrt{\mathbf{C}_s(h_1(\mathbf{d}), \cdots, h_m(\mathbf{d}))}}{|\mathcal{D}_t|}, \tag{27}$$

where $\mathbf{C}_s$ is defined in (6). It is clear that these m classifiers reach perfect consensus when c reaches its maximal value 1.

5.4 Experimental Results

5.4.1 Comparison of CCR_3, CCR_1, DE and CT

For each of the 96 problem instances described in Section 5.1, we record the values of accuracy and consensus for the resultant classifiers of Algorithm CCR_3 and CCR_1 on different values of θ. Table 2 gives an example of these measures for one of these problem instances. Due to the space limitation, we cannot list all the 96 tables. However, the properties in these tables are similar, as can be seen later in this section.

For each θ, Algorithm CCR_3 outputs three classifiers on the corresponding three source-domains. These classifiers are tested on their own source-domains and the target-domain, and the results are recorded from the 2nd to 7th column of Table 2. Then, the 8th column records the consensus measure of three classifiers. Finally, the accuracy performances of CCR_3 and CCR_1 are shown in the 9th and 10th column of this table respectively. As mentioned above, when $\theta = 0$ the accuracy for CCR_3 (73.9451 at the 1st row and 9th column of Table 2) is that of DE, and the accuracy for CCR_1 (71.9937 at the 1st row and 10th column of Table 2) is that of CT.

The results in Table 2 show that: 1) When $\theta \neq 0$, CCR_3 always outperforms DE and CT. Additionally, CCR_3 consistently outperforms CCR_1, which shows that exploiting the distribution differences among source-domains can improve the performance. 2) When $\theta \neq 0$, the performances of the local classifiers tested on their own source-domains are stable (always near 100%). Additionally, under this situation the performances of the local classifiers tested on the target-domain increase significantly. For example, the performance

[3] http://research.microsoft.com/∼minka/papers/logreg/

Table 2: The Accuracy (%) and Consensus Measure in an Example Problem

θ	The classifier on $\mathcal{D}_s^1$		The classifier on $\mathcal{D}_s^2$		The classifier on $\mathcal{D}_s^3$		Consensus	CCR_3	CCR_1
	Acc. on $\mathcal{D}_s^1$	Acc. on $\mathcal{D}_t$	Acc. on $\mathcal{D}_s^2$	Acc. on $\mathcal{D}_t$	Acc. on $\mathcal{D}_s^3$	Acc. on $\mathcal{D}_t$			
0	100	71.10	99.95	55.75	100	72.10	0.2775	73.94	71.99
0.05	100	92.14	99.95	90.61	99.94	93.20	0.6459	93.46	74.47
0.1	100	93.14	99.95	92.67	99.94	92.93	0.7385	93.30	76.11
0.15	100	93.83	99.95	93.46	99.89	93.88	0.7861	93.72	77.90
0.2	100	93.25	99.95	92.99	99.89	93.20	0.8146	93.25	80.43
0.25	100	93.35	99.95	92.99	99.89	93.14	0.8349	93.20	81.12

Source-domains: $\mathcal{D}_s^1(\mathcal{A}_2,\mathcal{B}_4)$, $\mathcal{D}_s^2(\mathcal{A}_1,\mathcal{B}_2)$, $\mathcal{D}_s^3(\mathcal{A}_4,\mathcal{B}_3)$; Target-domain: $\mathcal{D}_t(\mathcal{A}_3,\mathcal{B}_1)$

of the classifier on $\mathcal{D}_s^2$ increases from 55.7489% to more than 90% when it is applied to the target-domain. 3) When θ increases, the consensus measure of the resultant three classifiers increases. When this consensus measure reaches some extent, the classifiers always output the same results for an instance. So the performances of these classifiers tested on the target-domain are almost equal to that of CCR_3 when $\theta \neq 0$.

To further validate this on the other 95 tables, for each of these tables, we measure six values: 1) the performance of DE; 2) the performance of CT; 3) the average performance of CCR_3 when θ is sampled in [0.05,0.25], denoted by $\overline{CCR_3}$; 4) the best performance of CCR_3 when θ is sampled in [0.05,0.25], denoted by CCR_3^{max}; 5) the average performance of CCR_1 when θ is sampled in [0.05,0.25], denoted by $\overline{CCR_1}$; 6) the best performance of CCR_1 when θ is sampled in [0.05,0.25], denoted by CCR_1^{max}. For each of the six numbers we can average its values on all the 96 problem instances. These results are shown in Figure 2 and Table 3. In Figure 2(a) and Figure 2(b), the 96 problem instances are sorted by the increase order of the performances by CT.

Figure 2(a) shows that CCR_3^{max} outperforms DE and CT on every problem instance. Figure 2(b) shows that except Problem 45 CCR_3^{max} outperforms CCR_1^{max} on each problem instance, which further proves that explicitly exploiting the distribution differences among the source-domains increases the performance. In Figure 2(c) the x-axis represents the accuracy of DE while the y-axis represents the performance difference between CCR_3^{max} and DE. Figure 2(c) shows that this performance improvement decreases when the performance of DE increases. The reason is that if the accuracy of DE is high, these original classifiers usually output the same right results, and the consensus measure of the classifiers is big. In this case, the room for futher increasing this consensus measure is very limited, and thus the improvement by consensus regularization is small.

Table 3: Avg. Values(%) on 96 Problem Instances

CCR_3^{max}	$\overline{CCR_3}$	CCR_1^{max}	$\overline{CCR_1}$	DE	CT
92.6571	90.4172	83.7201	81.9982	79.2111	77.1886

Table 3 lists the average values of the six accuracy measures over the 96 problem instances in a decreasing order. Note that the same performance of CCR_3^{max} can be achieved by DCR in a distributed manner. Compared with DE, the accuracy of DCR increases from 79.2111 to 92.6571.

5.4.2 Comparison of TSVM, SGT, CoCC and CCR_3

To compare CCR_3 with CoCC, TSVM and SGT, we select the data sets in [2] which can be modified to fit our problem setting. We divide the single source-domain of the original problem into multiple source-domains. Four data sets, which are described in Table 4, are selected for this comparison. We measured the performance of CCR_3 on these problems by the two values $\overline{CCR_3}$ and CCR_3^{max}. The experimental results in Table 5 show that both $\overline{CCR_3}$ and CCR_3^{max} outperform TSVM and SGT on these four data sets. Except that CoCC slightly outperforms CCR_3 on the fourth data set, both $\overline{CCR_3}$ and CCR_3^{max} are better than CoCC on the other three data sets.

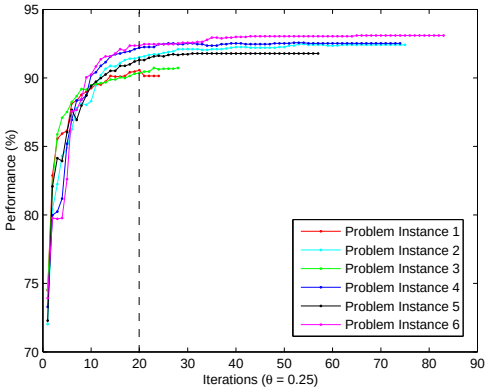


Figure 3: Iterations vs. the Performance of CCR_3.

Table 5: The Performance Comparison Results (%) among TSVM, SGT, CoCC and CCR_3

Data Set	TSVM	SGT	CoCC	$\overline{CCR_3}$	CCR_3^{max}
comp vs. sci	81.7	72.1	87.0	91.4	93.1
rec vs. talk	96.0	90.9	96.5	97.8	98.0
rec vs. sci	93.8	93.8	94.5	96.3	96.7
sci vs. talk	89.2	91.7	94.6	92.8	93.6

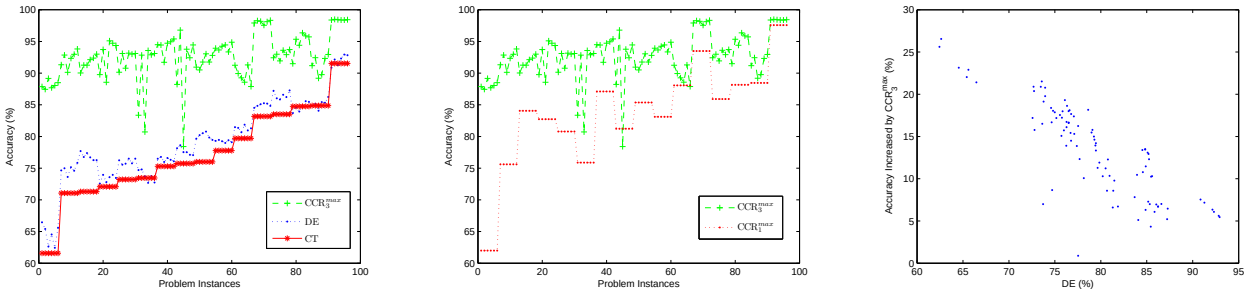
5.4.3 Algorithm Convergence

We check the convergence property of CCR_3 on 6 randomly selected problem instances. These results are shown in Figure 3, where the x-axis represents the number of iterations and the y-axis represents the accuracy performance. It shows that for each problem instance the accuracy performance increases along the number of iterations and almost converges after 20 iterations. This indicates that our algorithm owns a good property of convergence.

6. RELATED WORK

In this section, we introduce some existing works in the fields of transfer learning, self-taught learning, semi-supervised classification, and multi-view learning, which are closely related to the problem studied in this paper.

Transfer Learning aims to solve the fundamental problem of mismatched distributions between the training and



(a) CCR_3^{max} vs. DE and CT (b) CCR_3^{max} vs. CCR_1^{max} (c) DE vs. CCR_3^{max}

Figure 2: The Performance Comparison on the 96 Problem Instances

Table 4: The Data Description for the Performance Comparison among TSVM, SGT, CoCC and CCR_3

Data Set	$\mathcal{D}_s$		$\mathcal{D}_t$
	$\mathcal{D}_s^1$	$\mathcal{D}_s^2$	
comp vs. sci	comp.graphics sci.crypt	comp.os.ms-windows.misc sci.electronics	comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x sci.med, sci.space
rec vs. talk	rec.autos talk.politics.guns	rec.motorcycles talk.politics.misc	rec.sport.baseball rec.sport.hockey talk.politics.mideast talk.religion.misc
rec vs. sci	rec.autos sci.space	rec.sport.baseball sci.med	rec.motorcycles, rec.sport.hockey sci.crypt, sci.electronics
sci vs. talk	sci.electronics talk.religion.misc	sci.med talk.politics.misc	sci.crypt, talk.politics.guns sci.space, talk.politics.mideast

testing data. It is also referred to as *Cross-Domain Learning*, which adapts the knowledge from a source-domain (*in-domain*, *auxiliary-domain*) to a target domain (*out-of-domain*, *primary-domain*). In general, previous works in this area can be grouped into two categories. The first category is under the assumption that there are some labeled data from the target domain with different distribution. For instance, Liao et al. [10] estimated the degree of mismatch of each instance in the source domain with the whole target domain, and incorporated this information into logistic regression. Also, Dai et al. [1] extended boosting-based learning algorithms to transfer learning, in which the source-domain data with very different distribution are less weighted for data sampling. They also analyzed the theoretical effectiveness of this algorithm using the *Probability Approximately Correct* (PAC) theory. In addition, Yang et al. [5] studied the problem of transform the existing classifier from the source-domain to a target-domain in an incremental way. The principle behind this transforming is that the difference between the classifiers before and after adaption should be as small as possible. This work involves the data from multiple source-domains, but it does not consider the distribution difference among these source-domains. Meanwhile, Smith and Elkan [11] built generative classifiers, which address the selection bias problem in transfer learning. Finally, Raina et al. [12] focused on constructing information priors from the source-domain by a semi-definite program, and then encoded them into the model built.

In the second category, for the problem that the data from the target-domain are totally unlabeled, Dai et al. [2] proposed a *Co-clustering based Classification* method (CoCC), in which the class labels are transferred through the bridge of co-clustering. Xing et al. [3] proposed a transductive learning algorithm for this problem. Their method performs a two-phase label propagation, which is based on the adjacent matrix of the data. However, the method cannot output the classifier for future unlabeled data. The proposed method in this paper falls into this category of transfer learning.

Self-Taught Learning [13] studies how to use a large number of unlabeled data to improve performance on a given classification task. The distributions of the unlabeled data and the labeled data in the given task can be totally different (not related). Raina et al. [13] proposed an approach to self-taught learning that used sparse coding to construct higher-level features using the unlabeled data. The labeled data are represented by these succinct features. The significant performance improvement is shown in their experiments.

However, none of the above existing works consider the problem of transfer learning from multiple source-domains to a target domain in a distributed manner. These methods were not developed for the situation that the training data for transfer learning are geographically separate distributed. Moreover, we explicitly leverage the distribution differences among the source-domains and the target domain in our model to further improve the learning performance in the target domain.

Semi-Supervised Classification uses a large amount of unlabeled data, together with the labeled data, to build better classifiers. Different from transfer learning, the labeled and unlabeled data in semi-supervised learning are from the same distribution. The most related work in this area is semi-supervised learning by entropy minimization [14]. To compare with this method in the same problem setting, we assume that the labeled data and unlabeled data consist of a source-domain and target-domain, respectively. The regularization framework in [14] is recognized as an instance of the objective (17) with $m = 1$ (m is the number of source-domains). In other words, our regularization approach is more general and includes this method as a special case. Furthermore, our consensus regularization method is designed in a distributed manner.

Multi-View Learning is a new and natural, but non-standard learning problem, where the data are represented by multiple independent sets of features. As an example, Yarowsky [15] as well as Blum and Mitchell [16] noticed that having multiple representations can improve the performance of semi-supervised classification. The techniques of multi-view classification, such as co-Training [16] and MixtBoost [17], often boost the agreement among two views on unlabeled data. Sindhwani et al. [18] proposed a co-regularization approach, which penalizes not only the misclassifications by any individual classifier but also the high level of disagreement between different views. The experiments in [18] are performed on two views only, and the regularization term do not have the effect of entropy minimization. In addition, Dasgupta et al. [19] and Abney [20, 21] provided PAC bounds on the error of co-Training in terms of the disagreement rate of hypotheses on unlabeled data in two independent views. This inspires the principle of consensus maximization, which says that by minimizing the disagreement rate on unlabeled data, the error rate can be minimized. Our work utilizes this principle to another problem setting: transfer learning from multiple domains. From the point view of data partition, the difference between multi-view learning and multi-domain learning is that the data are vertically partitioned for multi-view learning while they are horizontally partitioned for multi-domain learning.

7. CONCLUSIONS

In this paper, we studied the problem of transfer learning from multiple source domains to a target domain. Specifically, for the case that data from multiple source domains and the target domain are semantically related, but have different distributions, we proposed a consensus regularization framework to exploit the distribution differences and learn the knowledge among training data from multiple source domains to boost the learning performance in a target domain. In this framework, we designed a distributed learning algorithm. In other words, a local classifier is trained at each source domain by considering both local data and the prediction consensus with the classifiers from other source domains. To modestly alleviate the privacy concerns, only some statistical data are shared between the source domains and the target domain, rather than the full contents of labeled data. Our experiments on real-world text data sets have shown that our consensus regularization learning method can effectively improve the learning performance in the target domain by leveraging the knowledge learnt from multiple source domains.

8. ACKNOWLEDGMENTS

The authors Fuzhen Zhuang and Qing He are supported by the National Science Foundation of China (No. 60435010, 60675010), 863 National High-Tech Program (No.2006AA01 Z128), National Basic Research Priorities Programme (No.2 007CB311004).

9. REFERENCES

[1] W. Dai, Q. Yang, G. Xue, and Y. Yu. Boosting for transfer learning. In *Proc. of the 24th ICML*, pages 193–200, 2007.

[2] W. Dai, G. Xue, Q. Yang, and Y. Yu. Co-clustering based classification for out-of-domain documents. In *Proc. of the 13th ACM SIGKDD*, pages 210–219, 2007.

[3] D. Xing, W. Dai, G. Xue, and Y. Yu. Bridged refinement for transfer learning. In *Proc. of the 13th PKDD*, pages 324–335, 2007.

[4] A. Smeaton and P. Over. Trecvid: Benchmarking the effectiveness of information retrieval tasks on digital video. In *Proc. of the Intl. Conf. on Image and Video Retrieval*, 2003.

[5] J. Yang, R. Yan, and A. G. Hauptmann. Cross-domain video concept detection using adaptive svms. In *Proc. of the 15th ACM MM*, pages 188–197, 2007.

[6] David Hosmer and Stanley Lemeshow. *Applied Logistic Regression*. Wiley, New York, 2000.

[7] Andrzej Ruszczynski. *Nonlinear Optimization*. Princeton University Press, Princeton, 2006.

[8] T. Joachims. Transductive inference for text classification using support vector machines. In *Proc. of the 16th ICML*, 1999.

[9] T. Joachims. Transductive learning via spectral graph partitioning. In *Proc. of the 20th ICML*, 2003.

[10] X. Liao, Y. Xue, and L. Carin. Logistic regression with an auxiliary data source. In *Proc. of the 22nd ICML*, pages 505–512, 2005.

[11] A. Smith and C. Elkan. Making generative classifiers robust to selection bias. In *Proc. of the 13th ACM SIGKDD*, pages 657–666, 2007.

[12] R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In *Proc. of the 23rd ICML*, pages 713–720, 2006.

[13] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: Transfer learning from unlabeled data. *Proc. of the 24th ICML*, pages 759–766, 2007.

[14] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *Proc. of the 19th NIPS*, pages 529–536, 2005.

[15] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. of the 33th ACL*, pages 189–196, 1995.

[16] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of the 11th COLT*, pages 92–100, 1998.

[17] Y. Grandvalet, F. d'Alché-Buc, and C. Ambroise. Boosting mixture models for semi-supervised learning. In *Proc. of the intl. conf. on Artificial Neural Networks*, pages 41–48, 2001.

[18] V. Sindhwani, P. Niyogi, and M. Belkin. A co-regularization approach to semi-supervised learning with multiple views. In *Proc. ICML Workshop on Learning with Multiple Views*, 2005.

[19] S. Dasgupta, M. L. Littman, and D. A. McAllester. Pac generalization bounds for co-training. In *Proc. of the 15th NIPS*, pages 375–382, 2001.

[20] S. Abney. Bootstrapping. In *Proc. of the ACL*, 2002.

[21] S. Abney. Understanding the yarowsky algorithm. *Computational Linguistics*, 30(3):365–395, 2004.