CoTrade: Confident Co-Training with Data Editing

Min-Ling Zhang and Zhi-Hua Zhou, *Senior Member, IEEE*

***Abstract*—Co-training is one of the major semi-supervised learning paradigms which iteratively trains two classifiers on two different views, and uses the predictions of either classifier on the unlabeled examples to augment the training set of the other. During the co-training process, especially in initial rounds when the classifiers have only mediocre accuracy, it is quite possible that one classifier will receive labels on unlabeled examples *erroneously* predicted by the other classifier. Therefore, the performance of co-training style algorithms is usually unstable. In this paper, the problem of how to *reliably* communicate labeling information between different views is addressed by a novel co-training algorithm named CoTrade. In each labeling round, CoTrade carries out the label communication process in two steps. Firstly, confidence of either classifier's predictions on unlabeled examples is explicitly estimated based on specific data editing techniques. Secondly, a number of predicted labels with higher confidence of either classifier are passed to the other one, where certain constraints are imposed to avoid introducing undesirable classification noise. Experiments on several real-world data sets across three domains show that CoTrade can effectively exploit unlabeled data to achieve better generalization performance.**

***Index Terms*—Machine learning, semi-supervised learning, co-training, data editing, bias-variance decomposition.**

I. Introduction

SEMI-supervised learning is one of the prominent ways to learn from both labeled and unlabeled data, which automatically exploit unlabeled data in addition to labeled data to improve learning performance without human intervention [11], [50]. Roughly speaking, existing semi-supervised learning algorithms can be categorized into several paradigms [48], including generative parametric models, semi-supervised support vector machines (S3VMs), graph-based approaches. Specifically, Blum and Mitchell's seminal work on *co-training* [4] started the research on the fourth paradigm of semi-supervised learning, i.e. disagreement-based semi-supervised learning [48]. Standard co-training deals with tasks whose input space has two different views (i.e. two independent sets of attributes) and works in an iterative manner. In each co-training round, two classifiers are trained separately on the different views and the predictions of either classifier on unlabeled examples are used to augment the training set of the other.

Following the work on standard co-training, a number of relevant approaches have been developed under different names [5], [6], [16], [25], [30], [45]–[47], [49]. Considering that their key learning process is to maintain a large disagreement between base learners, the name of *disagreement-based semi-supervised learning* was then coined to characterize their essential commonalities [48]. Standard co-training and its variants have chosen to measure the labeling confidence on unlabeled examples *implicitly*, e.g. by simply using the classifier's posteriori probability outputs [4], by repeatedly performing cross-validation on the original labeled examples [16], [44], or by additionally employing a third classifier [46].

In this paper, a new co-training style algorithm named CoTrade, i.e. Confident cO-TRAining with Data Editing, is proposed. Generally, data editing techniques aim to improve the quality of the training set through identifying and eliminating training examples wrongly generated in the labeling process, which are incorporated into CoTrade to facilitate reliable labeling information exchange between different views. Comparative experiments across three real-world domains clearly validate the effectiveness of CoTrade in exploiting unlabeled data to achieve strong generalization ability.

Generally, the major contributions of the proposed CoTrade approach are two-fold. Firstly, in each co-training round, CoTrade utilizes specific data editing techniques to *explicitly* obtain reliable estimates of either classifier's labeling confidence on unlabeled examples. Specifically, on either view, a weighted graph is constructed over the labeled and unlabeled examples based on k-nearest neighbor criterion. The labeling confidence for each unlabeled example is estimated by resorting to the *cut edge weight statistic* [27], [51], which reflects the *manifold assumption* [36] that examples with high similarities in the input space should share similar labels.

Secondly, in each co-training round, CoTrade employs certain mechanisms to sequentially augment the training set of one classifier with labels predicted by the other one in the order of *descending* labeling confidence. Specifically, labels predicted by either classifier are assumed to come from a classification process with random noise. The theoretical results on *learning from noisy examples* [1] are adopted to determine the appropriate amount of labeling information to be communicated between different views, so as to prevent performance degradation due to classification noise accumulation.

The rest of this paper is organized as follows. Section II reviews related work. Section III introduces basic notations

Min-Ling Zhang is with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China. Email: zhangml@seu.edu.cn.

Zhi-Hua Zhou is with the National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China. Email:zhouzh@lamda.nju.edu.cn.

This research was supported by the National Science Foundation of China (61073097, 61021062, 60805022), the National Fundamental Research Program of China (2010CB327903), Ph.D. Programs Foundation of Ministry of Education of China for Young Faculties (200802941009), and the Startup Foundation of Southeast University.

and sketch of CoTrade. After that, Section IV presents algorithmic details of CoTrade. Section V reports experimental results on a number of real-world data sets. Section VI further analyzes the underlying reasons for CoTrade's good performance. Finally, Section VII concludes and indicates several issues for future work.

II. Related Work

A great deal of research results have been achieved on semi-supervised learning. In this section we will focus on reviewing previous work related to co-training, while comprehensive reviews on semi-supervised learning can be found in [11], [48] and [50].

Standard co-training algorithm requires two *sufficient* and *redundant* views, i.e. the input space can be naturally partitioned into two sets of attributes, each of which is sufficient for learning and is conditionally independent to the other given the class label [4]. Dasgupta *et al.* [12] showed that when the above requirement is met, the co-trained classifiers could make few generalization errors by maximizing their agreement over the unlabeled data. Later, Balcan *et al.* [3] proved that given appropriately strong PAC [2] learners on each view, a weaker "*expansion*" assumption on the underlying data distribution is sufficient for iterative co-training to succeed. Wang and Zhou [38] provided one *sufficient* condition for co-training style algorithms to work, i.e. the two base learners should have large difference. It presents the first theoretical support to the success of some *single-view* co-training algorithms which do not require two views. Later, they [40] further provided the first *sufficient and necessary* condition for co-training to succeed, and also established connection between two major semi-supervised learning paradigms, i.e. graph-based methods and disagreement-based methods. Wang and Zhou [39] also showed that by combining co-training style algorithms with active learning (as in the style of the SSAIRA method [45]), the sample complexity can be improved further than pure semi-supervised learning or pure active learning.

Besides those theoretical analyses, researchers have also proposed several practical co-training style algorithms. Goldman and Zhou [16] presented an algorithm which does not require two views on the input space but instead needs two different supervised learning algorithms which can partition the input space into a set of *equivalence classes*. Later, they [44] extended this work by using a set of different algorithms instead of two domain-partition algorithms and predicting labels for unlabeled data by weighted majority voting. Zhou and Li [46] exploited unlabeled data using three classifiers and in each training round, an unlabeled example is labeled for a classifier only if the other two classifiers agree on the labeling. Thereafter, Li and Zhou [25] generalized their work by including more base classifiers in the ensemble. Du *et al.* [14] studied empirically on whether it is possible to discover the existence of two views in a single-view data upon which co-training can work reliably well, and showed that this is very difficult when there are only a few labeled examples. Currently, co-training style algorithms have been successfully applied to many real-world tasks, such as natural language processing [31], [33], [37], information retrieval [23], [45], computer-aid medical diagnosis [25], email spam detection [26], etc.

For any co-training style algorithm, one key to its success lies in how to choose each classifier's *confident* predictions on unlabeled examples to augment the training set of the other. Blum and Mitchell [4] employed classifiers capable of yielding probabilistic outputs (e.g. Naïve Bayes) and simply treated the classifier's posteriori outputs as the labeling confidence. However, it is quite possible that erroneous predictions would also have large posteriori outputs especially when the classifier has only moderate accuracy. Goldman and Zhou [16], [44] measured the labeling confidence (and also classifier accuracy) by frequently using ten-fold cross validation on the original labeled set. However, the process of cross validation is rather time-consuming and even may fail to obtain reliable estimates when there are only a small number of labeled examples. Zhou and Li [46] utilized a third classifier to help determine how to choose unlabeled examples to label. However, the labeling confidence is only implicitly qualified (instead of explicitly quantified) by whether two classifiers agree on the labeling or not. Furthermore, this method trains initial classifiers via bootstrap sampling [15] from labeled data set, where the training process could fail if only few labeled examples are available, e.g. possibly encountering bootstrapped samples with pure positive or negative examples.

In next two sections, we will present CoTrade which is capable of *explicitly* and *reliably* estimating the labeling confidence, and making use of them in an effective way.

III. Preliminaries and Algorithm Sketch

Let $\mathbb{X}$ be the input space and $\mathbb{Y} = \{0, 1\}$ be the output space. Under standard co-training setting, the input space is partitioned into two different views $\mathbb{X}^1$ and $\mathbb{X}^2$, i.e. $\mathbb{X} = \mathbb{X}^1 \times \mathbb{X}^2$. For any example $x \in \mathbb{X}$, we use x^1 and x^2 to denote its two portions under the first view $\mathbb{X}^1$ and the second view $\mathbb{X}^2$ respectively, i.e. $x = \langle x^1, x^2 \rangle$. Suppose $\mathcal{L} = \{(v_i, y_i)|i = 1, 2, \ldots, L\}$ contains L *labeled* training examples and $\mathcal{U} = \{u_j|j = 1, 2, \ldots, U\}$ contains U *unlabeled* training examples, where $v_i = \langle v_i^1, v_i^2 \rangle \in \mathbb{X}$, $u_j = \langle u_j^1, u_j^2 \rangle \in \mathbb{X}$ and $y_i \in \mathbb{Y}$.

The goal of CoTrade is to learn some hypothesis from the training set $\mathcal{L} \cup \mathcal{U}$ to classify unseen examples. Generally, the number of labeled examples in the training set is much smaller than that of unlabeled ones, i.e. $L \ll U$. Furthermore, let $\mathcal{L}_1$ and $\mathcal{U}_1$ denote respectively the labeled and unlabeled training set with respect to view $\mathbb{X}^1$, i.e. $\mathcal{L}_1 = \{(v_i^1, y_i)|i = 1, 2, \ldots, L\}$ and $\mathcal{U}_1 = \{u_j^1|j = 1, 2, \ldots, U\}$. The corresponding sets $\mathcal{L}_2$ and $\mathcal{U}_2$ with respect to view $\mathbb{X}^2$ can be defined in similar ways.

Similar to standard co-training algorithm, CoTrade also learns from the labeled and unlabeled training examples in an iterative manner. In each co-training round, labels predicted under each view are selected to augment the labeled training set under another view to help update the current classifiers. As for CoTrade, two core steps are employed to enable effective communications of labeling information between different views.

The first core step is to utilize data editing techniques to explicitly obtain reliable estimates of either classifier's labeling confidence on unlabeled examples. Most data editing techniques rely on specific learning procedures to improve the quality of the training set [9], [17], [20], [32], [41]. Recently, different to learning-based data editing techniques, Muhlenbach *et al.* [27] proposed a statistical approach based on *cut edge weight statistic* [51]. In this paper, CoTrade explicitly evaluates the confidence of whether an example has been correctly labeled from this cut edge weight statistic. Here, the statistic is derived from a graph constructed over the labeled and unlabeled examples based on k-nearest neighbor criterion. Note that similar strategies have also been successfully used to improve the self-training method [24].

The second core step is to appropriately choose a number of predicted labels of either view to augment the labeled training set of the other one. The predicted labels of either view could be regarded as *noisy* labels as the current classifiers used to make predictions are usually imperfect. In this paper, CoTrade treats the task of updating classifier from the augmented labeled training set as the process of learning from examples with classification noise, where the theoretical finding of Angluin and Laird [1] is adopted to optimize the expected error rate of the updated classifier based on the classification noise rate. Here, this rate is derived from the labeling confidence of predicted labels used for training set augmentation. Note that similar strategies have also been successfully incorporated into other co-training style algorithms [16], [46].

With the above two core steps in mind, the sketch of the CoTrade algorithm can be summarized as follows:

- Initialize classifiers f^i under view $\mathbb{X}^i$ based on $\mathcal{L}_i$ $(i = 1, 2)$;
- Repeat
  - Apply classifier f_i to predict labels of unlabeled examples in $\mathcal{U}_i$ $(i = 1, 2)$;
  - Estimate labeling confidence of either classifier with the help of data editing techniques (**core step I**);
  - Choose an appropriate set of predicted labels of either view to augment the labeled training set of the other one (**core step II**);
  - Update f_i by learning from the augmented labeled training set $(i = 1, 2)$;

  Until {Specified termination condition is satisfied}
- Return $\{f_1, f_2\}$.

IV. The Proposed Approach

Detailed descriptions and analyses of the proposed algorithm are scrutinized in this section. Firstly, the data editing techniques employed by CoTrade are introduced (core step I); Secondly, theoretical analyses on the labeling information exchange of CoTrade are discussed (core step II); Finally, the concrete learning procedure is outlined.

A. Data Editing

In each co-training round, CoTrade performs data editing in two steps consecutively. In the first step, an undirected *neighborhood graph* is constructed from a set of labeled examples $\mathcal{Z} = \{(z_p, y_p)|p = 1, 2, \ldots, Z\}$, which expresses the proximity between examples in feature space. There are numerous ways to generate this kind of graphs from examples, such as relative neighborhood graph, Gabriel graph, Delaunay triangulation, minimal spanning tree, etc [51]. Rather than using existing techniques, here we choose to construct the desired graph by employing the *k*-nearest neighbor criterion.

Concretely, each example $(z_p, y_p) \in \mathcal{Z}$ corresponds to a vertex in the graph $G_{\mathcal{Z}}$. There will be an edge $\overline{p\ q}$ connecting the two vertices of z_p and z_q if either z_p is among the *k*-nearest neighbors z_q or z_q is among the *k*-nearest neighbors of z_p. In this way, one example is *not only* related to its own neighbors, *but also* related to those ones which regard it as their neighbors. Furthermore, a weight $w_{pq} \in [0, 1]$ is associated to the edge $\overline{p\ q}$ computed as $(1 + d(z_p, z_q))^{-1}$, where $d(z_p, z_q)$ corresponds to the distance between z_p and z_q. In this paper, $d(z_p, z_q)$ is calculated with one of the most commonly-used measures, i.e. Euclidean distance.

In the second step, CoTrade evaluates the confidence of whether the label y_p associated with z_p is correct through exploring information encoded in $G_{\mathcal{Z}}$'s structure. The basic assumption is that a correctly labeled example should possess the same label to most of its neighboring examples, i.e. those sharing an edge with it in $G_{\mathcal{Z}}$. Intuitively, this coincides with the *manifold assumption* that examples with high similarity in the input space would also have high similarity in the output space [36]. An edge in $G_{\mathcal{Z}}$ is called a *cut edge* if the two vertices connected by it have different associated labels. Let H_0 be the null hypothesis that vertices of the graph are labeled independently according to distribution $\mathbf{D}(\mathbb{Y}) = \{\Pr(y = 1), \Pr(y = 0)\}$. Here, $\Pr(y = 1)$ $(\Pr(y = 0))$ denotes the prior probability of an example being positive (negative), which is usually estimated as the fraction of positive (negative) examples in $\mathcal{Z}$.

Then, the labeling confidence of each example (z_p, y_p) is estimated based on the following *cut edge weight statistic*:

$$J_p = \sum\nolimits_{z_q \in C_p} w_{pq} I_{pq} \tag{1}$$

Here, C_p corresponds to the set of examples which are connected with z_p in $G_{\mathcal{Z}}$. Under the null hypothesis, each I_{pq} corresponds to an *i.i.d.* Bernoulli random variable which takes the value of 1 (indicating a cut edge) if y_q is different to y_p. Accordingly, the probability of $\Pr(I_{pq} = 1)$ would be $1 - \Pr(y = y_p)$. When the size of C_p is sufficiently large, according to the *central limit theorem*, J_p can be approximately modeled by a normal distribution with mean $\mu_{p|H_0}$ and variance $\sigma^2_{p|H_0}$:

$$\mu_{p|H_0} = (1 - P(y = y_p)) \sum\nolimits_{z_q \in C_p} w_{pq} \tag{2}$$

$$\sigma^2_{p|H_0} = P(y = y_p)(1 - P(y = y_p)) \sum\nolimits_{z_q \in C_p} w^2_{pq} \tag{3}$$

Then, the standardized J_p, i.e. $J^s_p = (J_p - \mu_{p|H_0})/\sigma_{p|H_0}$, turns out to be a random variable governed by standard normal distribution $\mathbf{N}(0, 1)$.

Recall the manifold assumption encoded in the neighborhood graph, correctly labeled examples tend to have few

cut edges as its label should be consistent with most of its connected examples. According to the definition in Eq.(1), it is natural to assume that the *smaller* the value of J_p^s, the *higher* the confidence of y_p being the correct label of z_p. Therefore, based on the *left unilateral* p-value of J_p^s with respect to $\mathbf{N}(0,1)$, we can calculate the labeling confidence of (z_p, y_p) as follows:

$$\mathrm{CF}_{\mathcal{Z}}(z_p, y_p) = 1 - \Phi(J_p^s) \tag{4}$$

Here $\Phi(J_p^s) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{J_p^s} \mathrm{e}^{-\frac{t^2}{2}} \mathrm{d}t$ denotes the p-value of J_p^s under standard normal distribution. In this paper, for any labeled example $(z, y) \in \mathcal{Z}$, its labeling confidence estimated by conducting data editing on k-nearest neighborhood graph $G_{\mathcal{Z}}$ is denoted as $\mathrm{CF}_{\mathcal{Z}}(z, y)$.

Note that $\mathrm{CF}_{\mathcal{Z}}(z, y)$ represents only a *heuristic* way to estimate the labeling confidence of (z, y) based on the p-value of the cut edge weight statistic, which measures how smoothly the labels change with respect to the nearest neighbor graph. Although $\mathrm{CF}_{\mathcal{Z}}(z, y)$ should *by no means* be deemed to represent the *ground-truth probability* of y being the correct label of z, experimental results reported in the following sections validate the usefulness of this heuristic confidence estimation strategy in discriminating correctly labeled examples from incorrectly labeled ones.

B. Labeling Information Exchange

In each co-training round, CoTrade chooses to use the predictions on the unlabeled examples of current classifier under either view to augment the labeled training set of the other. Let f_1 be the current classifier under view $\mathbb{X}^1$, whose prediction $f_1(u^1)$ on an unlabeled example $u^1 \in \mathcal{U}_1$ may be communicated to the other view by generating a newly labeled example $(u^2, f_1(u^1))$ $(u^2 \in \mathcal{U}_2)$. As f_1 is usually away from errorless, whose predicted labels on unlabeled examples would be considered to be *noisy*.

In other words, $f_1(u^1)$ can be decomposed as $f_1(u^1) = f_1^*(u^1) + \zeta(u^1)$. Here, f_1^* corresponds to the target function which always yields the ground-truth label of each example, and $\zeta(u^1) \in \{-1, 0, 1\}$ is the random classification noise which would affect the predicted label of f_1. Therefore, to update the current classifier by exploiting the noise-prone labels communicated from the other view, the amount of labeling information to be exchanged between either view should be carefully controlled to avoid introducing too much classification noise.

In this paper, we adopt the theoretical finding of Angluin and Laird [1] on *learning from noisy examples* to facilitate labeling information exchange. Conceptually speaking, their results tackle the problem of PAC (Probably Approximately Correct) learning [2] under the condition of random classification noise. Next, we firstly describe the formal results of Angluin and Laird's finding, and then illustrate how to adopt their results for help fulfill CoTrade's labeling information exchange.

Let $\mathbb{Z}$ be the instance space with probability distribution function $\mathbf{D}$, namely $\int_{z \in \mathbb{Z}} \mathbf{D}(z)\, \mathrm{d}z = 1$. In addition, let $\mathcal{H} = \{H_i | i = 1, 2, \ldots, N\}$ be the finite hypothesis space of size N, where each hypothesis H_i maps from the input space $\mathbb{Z}$ to the output space $\{0, 1\}$. Let $H_\theta \in \mathcal{H}$ be the target (ground-truth) hypothesis to be learned, and $\rho = \{(z_p, y_p) | 1 \le p \le m\}$ be a sequence of m labeled instances with random classification noise. Here, each z_p is *independently* drawn from $\mathbb{Z}$ with respect to distribution $\mathbf{D}$. Each label y_p is assumed to be subject to a classification noise process with *noise rate* η, i.e. y_p takes the correct label $H_\theta(z_p)$ with probability $1 - \eta$ while the wrong label $1 - H_\theta(z_p)$ with probability η.

Let $dis(H_i, H_\theta) = \Pr_{z \sim \mathbf{D}}\{z | H_i(z) \neq H_\theta(z)\}$ denote the error rate of H_i with respect to H_θ. Furthermore, let $H_* \in \mathcal{H}$ be the hypothesis which has minimum disagreement with the sequence ρ, i.e. $H_* = \arg\min_{H_i \in \mathcal{H}} \sum_{p=1}^{m} [\![H_i(z_p) \neq y_p]\!]$.[1] Then, given the *tolerance* parameter ϵ, the *confidence* parameter δ, and the upper bound on the noise rate η^b (all smaller than 0.5), the following theorem states the PAC property of learning from noisy examples:

Theorem 1 (Angluin & Laird [1], 1988)
Given a sequence ρ of m independently drawn labeled instances, when the sample size m satisfies:

$$m \ge \frac{2}{\epsilon^2 (1 - 2\eta^b)^2} \ln\left(\frac{2N}{\delta}\right) \tag{5}$$

Then, the hypothesis H_* which minimizes the disagreement with ρ will have the PAC property:

$$\Pr\left[dis(H_*, H_\theta) \ge \epsilon\right] \le \delta. \tag{6}$$

Here $\Pr[\cdot]$ is evaluated over all the possible sequences of ρ with length m.

In other words, under specific level of noise rate (i.e. η^b), Eq.(5) specifies *how many* noisy labeled instances (i.e. m) are needed to learn a classifier with expected low error rate ϵ at high probability $1 - \delta$. Next we will show how this theorem could be adopted to guide the process of CoTrade's labeling information exchange.

Given current classifiers f_1 and f_2 under view $\mathbb{X}^1$ and view $\mathbb{X}^2$ respectively, let $f^\circ(\mathcal{S}) = \{(s, f(s)) | s \in \mathcal{S})\}$ denote the labeled set obtained by applying classifier f to predict labels of the unlabeled examples in $\mathcal{S}$. Accordingly, labeling confidence of the newly labeled examples in $f_1^\circ(\mathcal{U}_1)$ and $f_2^\circ(\mathcal{U}_2)$ will be explicitly estimated by conducting data editing on $\mathcal{L}_1 \cup f_1^\circ(\mathcal{U}_1)$ and $\mathcal{L}_2 \cup f_2^\circ(\mathcal{U}_2)$ respectively. After that, labels predicted by one classifier can be successively used to augment the training set of the other, in the order of *descending* labeling confidence.

Note that classification noise encoded in the predicted labels would keep increasing when more and more labeling information is exchanged between the classifiers. Therefore, in order to prevent performance degradation caused by accumulated classification noise, CoTrade has to carefully choose *appropriate amount* of labels to be transferred from one classifier to the other.

Suppose f_1 passes its predicted labels on a chosen subset of examples $\mathcal{U}_1' \subseteq \mathcal{U}_1$ to their counterparts $\mathcal{U}_2' \subseteq \mathcal{U}_2$. Then, f_2 will be updated to another classifier learned from $\mathcal{L}_2 \cup f_1^\triangle(\mathcal{U}_1')$.

[1] For any predicate π, $[\![\pi]\!] = 1$ if π holds. Otherwise, $[\![\pi]\!] = 0$.

Here, $f_1^{\triangle}(\mathcal{U}_1')$ represents the set of labeled examples under view $\mathbb{X}^2$ through passing f_1's predictions on $\mathcal{U}_1'$ to $\mathcal{U}_2'$:

$$f_1^{\triangle}(\mathcal{U}_1') = \{(u^2, f_1(u^1))|(u^1, u^2) \in \mathbb{X}, u^1 \in \mathcal{U}_1', u^2 \in \mathcal{U}_2'\} \quad (7)$$

The set of labeled examples $f_2^{\triangle}(\mathcal{U}_2')$ is defined in similar ways.

As $f_1^{\triangle}(\mathcal{U}_1')$ usually contains noisy labels communicated from f^1, the task of updating f^2 based on $\mathcal{L}_2 \cup f_1^{\triangle}(\mathcal{U}_1')$ can be treated as the process of learning from examples with classification errors. Resorting to Eq.(5), by fixing N, δ and let $c = 2\ln(\frac{2N}{\delta})$, the *least* accommodable hypothesis classification error ϵ given m and η^b will be:

$$\epsilon = \sqrt{\frac{c}{m(1-2\eta^b)^2}} \quad (8)$$

When learning from $\mathcal{L}_2 \cup f_1^{\triangle}(\mathcal{U}_1')$, the sample size m as shown in Eq.(8) becomes:

$$m_{\mathcal{U}_1'} = |\mathcal{L}_2 \cup f_1^{\triangle}(\mathcal{U}_1')| = L + |\mathcal{U}_1'| \quad (9)$$

Furthermore, to make Eq.(8) be practical for the guidance of labeling information exchange, the noise rate upper bound η^b should be reasonably set. Here, we propose to *heuristically* deriving η^b by utilizing current estimated labeling confidence:

$$\eta^b_{\mathcal{U}_1'} = \frac{1}{m_{\mathcal{U}_1'}} \sum\nolimits_{u^1 \in \mathcal{U}_1'} \left(1 - \mathrm{CF}_{\mathcal{L}_1 \cup f_1^{\circ}(\mathcal{U}_1)}(u^1, f_1(u^1))\right) \quad (10)$$

Here, $\mathrm{CF}_{\mathcal{L}_1 \cup f_1^{\circ}(\mathcal{U}_1)}(u^1, f_1(u^1))$ corresponds to the labeling confidence of a newly labeled example $(u^1, f_1(u^1)) \in \mathcal{L}_1 \cup f_1^{\circ}(\mathcal{U}_1)$ as defined in Eq.(4). Therefore, Eq.(10) reflects the assumption that the *higher* the labeling confidence of an example the *lower* the possibility of it being a noisy example. Note that the original labeled set $\mathcal{L}_2$ is assumed to be noise-free.

Substituting Eqs.(9) and (10) into Eq.(8), we can evaluate the expected classification error $\epsilon_{\mathcal{U}_1'}$ when f_1's predicted labels on $\mathcal{U}_1'$ are used to augment the training set of f_2:

$$\epsilon_{\mathcal{U}_1'} = \frac{\sqrt{c} \cdot \sqrt{L + |\mathcal{U}_1'|}}{L + |\mathcal{U}_1'| - 2\sum_{u^1 \in \mathcal{U}_1'} \left(1 - \mathrm{CF}_{\mathcal{L}_1 \cup f_1^{\circ}(\mathcal{U}_1)}(u^1, f_1(u^1))\right)} \quad (11)$$

By keeping class distribution in $f_1^{\circ}(\mathcal{U}_1')$ the same as that in $\mathcal{L}_1$, we can generate a series of *candidate* unlabeled data sets $\Xi_1 = \{\mathcal{U}_1^{\xi} | \xi \in \mathbb{N}\}$ to constitute supplementary labeled examples $f_1^{\triangle}(\mathcal{U}_1^{\xi})$ for f_2. Let γ be the ratio of the number of negative examples to the number of positive examples in $\mathcal{L}_1$. Without loss of generality, we can assume that γ is greater than 1. Then, $\mathcal{U}_1^{\xi}$ is formed by choosing ξ examples in $\mathcal{U}_1$ with highest labeling confidence of being *positive* and $\lceil \gamma \cdot \xi \rceil$ examples in $\mathcal{U}_1$ with highest labeling confidence of being *negative*, if exist. CoTrade identifies optimal choice $\mathcal{U}_1^* \in \Xi_1$ for labeling information exchange which would yield *smallest* expected classification error ϵ_1:

$$\mathcal{U}_1^* = \arg\min_{\mathcal{U}_1^{\xi} \in \Xi_1} \epsilon_{\mathcal{U}_1^{\xi}}, \quad \epsilon_1 = \epsilon_{\mathcal{U}_1^*} \quad (12)$$

Here $\epsilon_{\mathcal{U}_1^{\xi}}$ is calculated based on Eq.(11).

Thereafter, if ϵ_1 is smaller than its previous value ϵ_1' determined in preceding round, f_2 will be updated based on $\mathcal{L}_2$ together with the identified optimal choice, i.e. $\mathcal{L}_2 \cup f_1^{\triangle}(\mathcal{U}_1^*)$. Note that in Eq.(11), the constant term $\sqrt{c}$ will have no impact on CoTrade's training procedure and thus is dropped from the numerator. The initial value of ϵ_1 before CoTrade launches its co-training iteration is set to be $1/\sqrt{L}$ (i.e. $\epsilon_{\emptyset}$). Similar notations and statements can be made when analyzing how f_2 uses its predictions to augment the training set of f_1.

Note that the theoretical results of Theorem 1 hold for the case of *finite* hypothesis space, while analysis for the case of infinite hypothesis space can be conducted with the help of *Vapnik-Chervonenkis* (VC) dimension [22]. Although the hypothesis space studied in this paper is actually infinite, we still choose to adopt the finite version of Theorem 1 due to its clarity and simplicity. Furthermore, this choice is also inspired by previous success in applying Theorem 1 to design co-training style algorithms which deal with infinite hypothesis space too [16], [46].

TABLE I
THE COTRADE ALGORITHM.

$[f_1, f_2]$ = CoTrade($\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{U}_1$, $\mathcal{U}_2$, $Learner$, k)

Inputs:

$\mathcal{L}_1$: labeled set $\{(v_i^1, y_i)|1 \le i \le L\}$ under view $\mathbb{X}^1$
$\mathcal{L}_2$: labeled set $\{(v_i^2, y_i)|1 \le i \le L\}$ under view $\mathbb{X}^2$
$\mathcal{U}_1$: unlabeled set $\{u_1^j|1 \le j \le U\}$ under view $\mathbb{X}^1$
$\mathcal{U}_2$: unlabeled set $\{u_2^j|1 \le j \le U\}$ under view $\mathbb{X}^2$
$Learner$: the learning procedure which takes a labeled training set and induces a binary classifier
k : the number of nearest neighbors considered in neighborhood graph construction

Outputs:

f_1: the returned classifier under view $\mathbb{X}^1$
f_2: the returned classifier under view $\mathbb{X}^2$

Process:

1 $f_1' \leftarrow Learner(\mathcal{L}_1)$;
$f_2' \leftarrow Learner(\mathcal{L}_2)$; % initializing classifiers
2 $e_1' \leftarrow MeasureError(f_1', \mathcal{L}_1)$;
$e_2' \leftarrow MeasureError(f_2', \mathcal{L}_2)$; % measuring predictive error
3 $\epsilon_1' \leftarrow 1/\sqrt{L}$; $\epsilon_2' \leftarrow 1/\sqrt{L}$;
4 $Iter \leftarrow 1$;
While ($Iter \le 50$) **do**
5 Generate $f_1'^{\circ}(\mathcal{U}_1)$ and $f_2'^{\circ}(\mathcal{U}_2)$;
% making predictions on unlabeled data
6 Construct neighborhood graphs from $\mathcal{L}_1 \cup f_1'^{\circ}(\mathcal{U}_1)$ as well as $\mathcal{L}_2 \cup f_2'^{\circ}(\mathcal{U}_2)$, and estimate the labeling confidence based on the constructed graphs; % **core step I**
7 Identify optimal choices $(\mathcal{U}_1^*, \epsilon_1)$ and $(\mathcal{U}_2^*, \epsilon_2)$ for labeling using Eq.(12); % **core step II**
8 $f_1 \leftarrow Learner(\mathcal{L}_1 \cup f_2'^{\triangle}(\mathcal{U}_2^*))$
$f_2 \leftarrow Learner(\mathcal{L}_2 \cup f_1'^{\triangle}(\mathcal{U}_1^*))$;
9 $e_1 \leftarrow MeasureError(f_1, \mathcal{L}_1)$;
$e_2 \leftarrow MeasureError(f_2, \mathcal{L}_2)$;
10 **if** ($e_1 > e_1'$ || $e_2 > e_2'$) **then** GOTO **15**;
11 **if** ($\epsilon_1 \ge \epsilon_1'$ && $\epsilon_2 \ge \epsilon_2'$) **then** GOTO **15**;
12 $Iter \leftarrow Iter + 1$;
13 **if** ($\epsilon_1 < \epsilon_1'$) **then** $\epsilon_1' \leftarrow \epsilon_1$; $f_1' \leftarrow f_1$;
14 **if** ($\epsilon_2 < \epsilon_2'$) **then** $\epsilon_2' \leftarrow \epsilon_2$; $f_2' \leftarrow f_2$;
End of While
15 $f_1 \leftarrow f_1'$; $f_2 \leftarrow f_2'$;

C. Iterative Procedure

To sum up, Table I presents the pseudo-code of the proposed algorithm. As for input parameters, $\mathcal{L}_1$, $\mathcal{L}_2$ $\mathcal{U}_1$, and $\mathcal{U}_2$ correspond to the labeled and unlabeled data sets under either view, $Learner$ specifies the learning procedure used to induce binary classifier from labeled training set, and k sets the number of nearest neighbors used for neighborhood graph construction; As for output parameters, f_1 and f_2 return the trained classifiers under either view.

As shown in Table I, CoTrade initializes itself on the labeled training sets $\mathcal{L}_1$ and $\mathcal{L}_2$ (steps 1 to 4), and then iteratively refines two classifiers by confidently exchanging labeling information between each other (steps 5 to 15). The co-training process automatically stops when either classifier's predictive error on original labeled set increases (step 10), or the expected predictive errors of both classifiers won't decrease (step 11). The maximum number of co-training rounds is set to 50, while empirical results show that in most cases CoTrade terminates within no more than 10 iterations.

Note that traditional co-training procedures *permanently* add pseudo-labeled examples in each round to the labeled examples, which may be problematic as classification noise in those pseudo-labels may be undesirably accumulated round by round. Therefore, in CoTrade we choose not to progressively grow the labeled training set with those predicted labels on unlabeled examples, while useful information conveyed by the predicted labels of each round is implicitly passed to the subsequent learning process via the updated classifiers.

V. Experiments

A. Data Sets

To evaluate the performance of CoTrade, we employ six data sets derived from the following three real-world domains, where each data set is associated with two naturally partitioned or artificially generated views:

• *Web page classification*: This problem focuses on 1,051 home pages collected from web sites of Computer Science departments of four universities: Cornell University, University of Washington, University of Wisconsin and University of Texas[2]. These pages have been manually labeled into a number of categories such as *student*, *faculty*, *staff*, *course*, etc., among which the category *course* home page is regarded as the target. That is, course home pages (22%) correspond to positive examples while all the other pages are negative examples. Each page has a *page-based* view (words appearing in the page itself) and a *link-based* view (words appearing in hyperlinks pointing to it), and the task is to predict whether it is a *course page* or not. The resultant data set associated with *page-based* view and *link-based* view is referred as the *course* data in the rest of this paper.

• *Advertisement image filtering*: This problem is investigated by Kushmerick [21] in his work of automatically removing advertising images in web pages. Each image is described from multiple views, such as *image properties* (height, width, aspect ratio, etc.), *image caption* (words enclosed in ⟨A⟩ tag), *image url* (words occurring in the image source's url), *base url* (words occurring in the affiliated web page's url), *destination url* (words occurring in the image anchor's url), etc. Specifically, using any two out of the three *url-based views* (i.e. *1-image url*, *2-base url*, *3-destination url*), we create three *ads* data sets named *ads12*, *ads13* and *ads23*. For any *ads* data set, each image is associated two different views and the task is predict whether it is a *advertisement* or not.

• *Newsgroup postings categorization*: This problem is considered by Muslea *et al.* [28], [29] in their study of robust multi-view learning. A total of 16 newsgroups postings from the *Mini-Newsgroup* data are used[3], and each consists of 100 postings randomly drawn from the 1,000 postings in the original *20-Newsgroup* data [18]. The 16 chosen newsgroups are divided into four groups, denoted as $A_1 - A_4$, $B_1 - B_4$, $C_1 - C_4$ and $D_1 - D_4$[4]. The first two groups form the first view while the last two groups form the second view. Following this strategy, two co-training data sets are created as follows:

1) *NG1*: A positive example is generated by randomly paring one example from $A_1 - A_4$ to another example from $C_1 - C_4$. Correspondingly, a negative example is generated by randomly paring one example from $B_1 - B_4$ to another example from $D_1 - D_4$.
2) *NG2*: A positive example is generated by randomly paring one example from $A_1 - A_2$ to another example from $C_1 - C_2$, or randomly paring one example from $A_3 - A_4$ to another example from $C_3 - C_4$. Correspondingly, a negative example is generated by randomly paring one example from $B_1 - B_2$ to another example from $D_1 - D_2$, or randomly paring one example from $B_3 - B_4$ to another example from $D_3 - D_4$.

For the first four data sets, i.e. *course*, *ads12*, *ads13* and *ads23*, each example in them bears textual representation. Accordingly, examples are described as feature vectors in feature space $\mathcal{F}$ based on *Boolean weighting* [34], where features of $\mathcal{F}$ correspond to words in the vocabulary. Each feature of one example is set to be 1 if the the example contains the corresponding word and set to be 0 otherwise. Furthermore, dimensionality reduction techniques based on *gain ratio* [43] are performed and 10% of the original features are retained. In addition to Boolean representation, for the other two data sets, i.e. *NG1* and *NG2*, examples are described as *numerical* feature vectors based on *tf-idf weighting* [43]. Furthermore, dimensionality reduction techniques based on *document frequency* [43] are performed and 2% of the original features are retained. Table II summarizes the characteristics of the experimental data sets used in this paper.

For each data set, 25% of the data are kept as test examples while the rest are used as training examples, i.e. $\mathcal{L} \cup \mathcal{U}$. Class

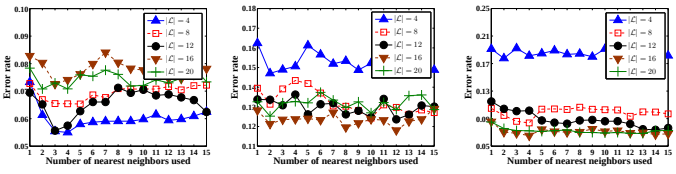
[2]Data available at http://www.cs.cum.edu/afs/cs/project/theo-11/www/wwkb/

[3]Data available at http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes/mini_newsgroup.tar.gz

[4]{A_1: comp.os.ms-windows.misc, A_2: comp.sys.ibm.pc.hardware, A_3: rec.autos, A_4: rec.sport.baseball}; {B_1: sci.crypt, B_2: sci.space, B_3: talk.politics.guns, B_4: talk.politics.misc}; {C_1: comp.windows.x, C_2: comp.sys.mac.hardware, C_3: rec.motorcycles, C_4: rec.sport.hockey}; {D_1: sci.electronics, D_2: sci.med, D_3: talk.politics.mideast, D_4: talk.religion.misc}

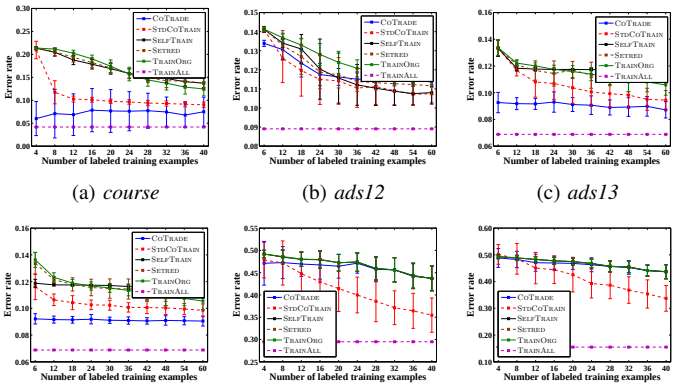
TABLE II
CHARACTERISTICS OF THE DATA SETS.

Data set	Number of examples	View content		Dimensionality		Class proportion	
		view 1	view 2	view 1	view 2	positive	negative
course	1,051	page itself	links to the page	344	42	21.88%	78.12%
ads12	983	image URL	base URL	45	49	14.04%	85.96%
ads13	983	image URL	destination URL	45	47	14.04%	85.96%
ads23	983	base URL	destination URL	49	47	14.04%	85.96%
NG1	800	groups A_1 to B_4[4]	groups C_1 to D_4[4]	303	334	50.00%	50.00%
NG2	800	groups A_1 to B_4[4]	groups C_1 to D_4[4]	303	334	50.00%	50.00%



(a) NAÏVE BAYES (b) CART (c) LIBSVM

Fig. 1. Classification error rate of COTRADE (on the *course* data set) changes as the number of nearest neighbors used (i.e. k) increases. Each sub-figure reports the experimental results with different classifier inducers, from left to right: NAÏVE BAYES, CART and LIBSVM.



(a) *course* (b) *ads12* (c) *ads13*

(d) *ads23* (e) *NG1* (f) *NG2*

Fig. 2. Classification error rate of each comparing algorithm changes as the number of labeled training examples increases, where NAÏVE BAYES is utilized as the classifier inducer.

distribution of $\mathcal{L}$, $\mathcal{U}$ and the test set are similar to that of the original data set. To simulate real-world cases where labeled examples are rarely available, $\mathcal{L}$ is set to contain only a small number of examples. For the *course* data set, we have created $\mathcal{L}$ with ten different configurations $\{\alpha\mathbf{p}\beta\mathbf{n}|\beta = 3\alpha, \alpha = 1, 2, \ldots, 10\}$, where $\alpha\mathbf{p}\beta\mathbf{n}$ denote that α positive examples and β negative examples are selected; For any of the three *ads* data sets, we have created $\mathcal{L}$ with ten different configurations $\{\alpha\mathbf{p}\beta\mathbf{n}|\beta = 5\alpha, \alpha = 1, 2, \ldots, 10\}$; Finally, for either of the *newsgroup* data sets, we have also created $\mathcal{L}$ with ten different configurations $\{\alpha\mathbf{p}\beta\mathbf{n}|\beta = \alpha, \alpha = 2, 4, \ldots, 20\}$.

B. Experimental Setup

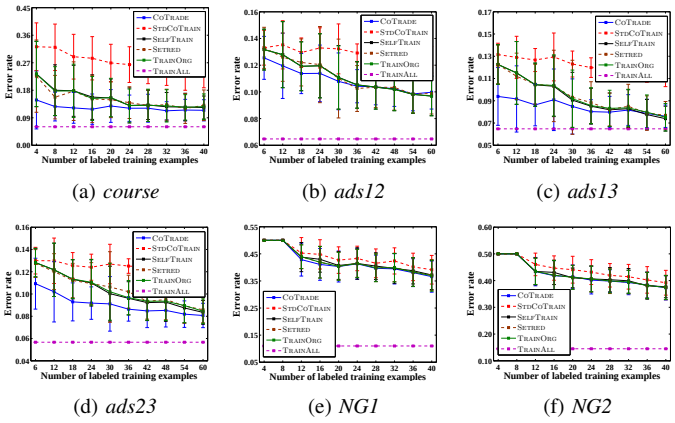
The performance of COTRADE is compared with three semi-supervised learning algorithms. The first comparing algorithm is the standard co-training algorithm (STDCOTRAIN) [4]. Furthermore, COTRADE is compared with another well-known semi-supervised learning algorithm SELFTRAIN [30]. Unlike STDCOTRAIN, SELFTRAIN initially trains a classifier on labeled data and then iteratively augment its labeled training set by adding several newly labeled unlabeled examples with most confident predictions *of its own* (instead of the other classifier). In addition to SELFTRAIN, COTRADE is further compared with SETRED [24], which is a variant of SELFTRAIN incorporated with data editing techniques[5].

For any comparing algorithm, several kinds of learning approaches are employed to perform classifier induction, aiming to investigate how each comparing algorithm behaves along with base learners bearing diverse characteristics. Specifically, the Bayesian-style method of NAÏVE BAYES, nonmetric-style method of decision trees (CART implementation [8]) and kernel-style method (LIBSVM implementation [10]) are utilized. Note that besides NAÏVE BAYES which can yield probabilistic outputs, CART and LIBSVM are also triggered to give probability estimates in order to incorporate them with STDCOTRAIN, SELFTRAIN and SETRED. Concretely, CART employs the proportion of dominating class in leaf node as probabilistic output. LIBSVM is configured to give probabilistic estimates by using the training option "-b 1"[6].

For STDCOTRAIN, the same training strategy as used by Blum and Mitchell [4] is adopted. Concretely, in each co-training round, one classifier's labeled training set is *incrementally* enlarged using the *"most confident"* outputs (labels with highest posteriori estimates) of its own and the other classifier. SELFTRAIN and SETRED also update two classifiers in their iterative training process, while in each round the *"most confident"* outputs of one classifier is only used as candidates to enlarge the labeled training set of its own. To avoid introducing too much noise, in each training round, each classifier of STDCOTRAIN, SELFTRAIN and SETRED only selects $1\mathbf{p}3\mathbf{n}$ examples for the *course* data, $1\mathbf{p}5\mathbf{n}$ examples for

[5]Note that the other two co-training style algorithms reported in [16] and [46] are not included here for comparison, as when the number of labeled examples is *quite few* (e.g. $|\mathcal{L}|$=4 for the *course* data), both of them would fail to function due to the embedded cross-validation [16] or bootstrap sampling [46] procedures on $\mathcal{L}$.

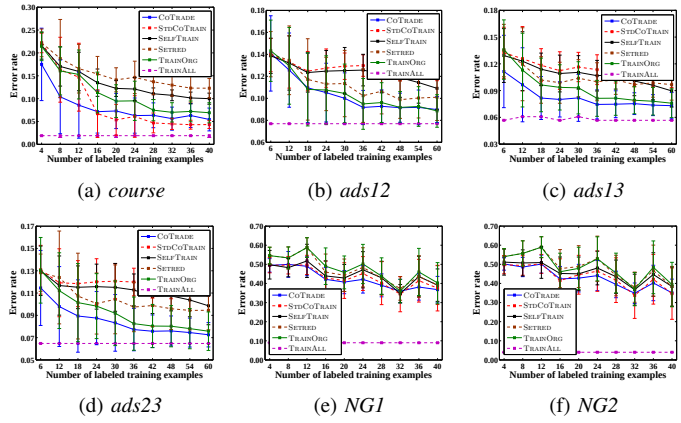
[6]For NAÏVE BAYES, the class prior probabilities are calculated based on *frequency counting*, and the class conditional probabilities are estimated by frequency counting for binary features while by fitting normal distributions for numerical features; For CART, the *Gini's diversity index* is used as the splitting criterion for classification tree building; For LIBSVM, kernel type is radial basis function for *course* and *ads* data sets while linear function for *newsgroup* data sets.



(a) *course* (b) *ads12* (c) *ads13*

(d) *ads23* (e) *NG1* (f) *NG2*

Fig. 3. Classification error rate of each comparing algorithm changes as the number of labeled training examples increases, where CART is utilized as the classifier inducer.



(a) *course* (b) *ads12* (c) *ads13*

(d) *ads23* (e) *NG1* (f) *NG2*

Fig. 4. Classification error rate of each comparing algorithm changes as the number of labeled training examples increases, where LIBSVM is utilized as the classifier inducer.

the *ads* data, and 1**p**1**n** examples for the *newsgroup* data. These algorithms terminate when no more examples are available for labeling or the number of training rounds reaches 50.

Furthermore, we have also included two *baseline* algorithms named TRAINORG and TRAINALL for reference purpose. TRAINORG trains classifiers on only the *initial* labeled training examples while TRAINALL trains classifiers on labeled examples *together with* unlabeled ones assuming that their ground-truth labels are available. Conceptually, TRAINORG and TRAINALL would serve as the *lower* and *upper* bound respectively for performance comparison. For any comparing algorithm, classifiers finally learned on two different views are combined to make predictions using the same method as in [4], i.e. choosing one of the two classifiers' outputs with higher posteriori estimate.

C. Experimental Results

For each comparing algorithm equipped with any classifier inducer, 100 independent runs are performed under every configuration of $\mathcal{L}$. In each run, a number of training examples are randomly chosen to constitute the desired labeled set $\mathcal{L}$ and the rest training examples are used to constitute the unlabeled set $\mathcal{U}$. For COTRADE (as shown in Table I), when the training examples ($\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{U}_1$, $\mathcal{U}_2$) and classifier inducer (*Learner*) are fixed, the parameter remained to be specified is k, i.e. number of nearest neighbors used in graph construction. Fig. 1 gives the performance of COTRADE on the *course* data set with different base learners, where k gradually varies from 1 to 15. Each point in the plot gives the average classification error rate of COTRADE out of 100 independent runs.

As shown in Figs. 1(a) to 1(c), in most cases, the performance of COTRADE slightly improves in the initial increasing phase of k ($k \leq 3$), and tends to *level up* (i.e. do not significantly change) in subsequent increasing phase of k ($k \geq 5$). Therefore, in the rest of this paper, all reported experimental results of COTRADE are obtained with $k = 10$.

Fig. 2 to Fig. 4 illustrate how each comparing algorithm performs with different classifier inducers, as the number of labeled training examples in $\mathcal{L}$ increases. Each point in the plot gives the average classification error rate of the comparing algorithm out of 100 independent runs.

As shown in Fig. 2 to Fig. 4, in most cases, few comparing algorithms (except TRAINALL which serves as the performance upper bound) consistently outperforms COTRADE. Concretely, on the *course* data set (plot (a) of each figure), COTRADE is consistently superior to STDCOTRAIN, SELFTRAIN, SETRED and TRAINORG when either NAÏVE BAYES or CART is incorporated as the classifier inducer. With LIBSVM, COTRADE consistently outperforms SELFTRAIN, SETRED and TRAINORG and is only inferior to STDCOTRAIN as the number labeled training examples exceeds 16.

On the *ads12* data set (plot (b) of each figure), COTRADE is less distinguishable from SETRED and TRAINORG with all classifier inducers. It is consistently superior to SELFTRAIN with LIBSVM and consistently superior to STDCOTRAIN with CART and LIBSVM. Note that the two views associated with this data set, i.e. *image url* and *base url*, may be strongly *correlated* due to the co-occurrence of domain names. For instance, for a tiger image, the *image url* and *base url* would probably correspond to "`http://`**`www.base-domain.com`**`/images/tiger.jpg`" and "`http://`**`www.base-domain.com`**`/index.html`". The high correlation between two views may weaken the benefits of labeling information exchange brought by co-training style algorithms, as the two classifiers trained on different views would be quite similar. This may be the reason that COTRADE does not evidently differ from some comparing algorithms on *ads12*.

On the *ads13* and *ads23* data sets (plots (c) and (d) of each figure), COTRADE consistently outperforms STDCOTRAIN, SELFTRAIN, SETRED and TRAINORG with all classifier inducers; On the *NG1* and *NG2* data sets (plots (e) and (f) of each figure), COTRADE is inferior to STDCOTRAIN with NAÏVE BAYES, superior to STDCOTRAIN with CART, and nearly indistinguishable to STDCOTRAIN with LIBSVM. COTRADE is also less distinguishable to SELFTRAIN, SETRED and TRAINORG with either NAÏVE BAYES or CART, while is slightly superior to them with LIBSVM. Reasons

TABLE III
THE WIN/TIE/LOSS COUNTS FOR COTRADE AGAINST STDCOTRAIN, SELFTRAIN, SETRED AND TRAINORG UNDER DIFFERENT DATA SETS AND CLASSIFIER INDUCERS.

Data set	CoTrade **against**	Base learner (**win/tie/loss** [*min. p-value*, *max. p-value*, *ave. p-value*])		
		NAÏVE BAYES	CART	LIBSVM
course	STDCOTRAIN	**10/0/0** [*2e-59, 3e-5, 9e-6*]	**10/0/0** [*7e-40, 2e-26, 2e-27*]	**3/2/5** [*5e-14, 7e-1, 1e-1*]
	SELFTRAIN	**10/0/0** [*2e-64, 3e-34, 7e-35*]	**9/1/0** [*4e-19, 6e-2, 1e-2*]	**10/0/0** [*2e-37, 6e-7, 6e-8*]
	SETRED	**10/0/0** [*1e-25, 4e-11, 4e-12*]	**6/4/0** [*2e-7, 9e-2, 3e-2*]	**10/0/0** [*8e-19, 2e-3, 2e-4*]
	TRAINORG	**10/0/0** [*1e-64, 1e-25, 1e-26*]	**10/0/0** [*4e-19, 3e-2, 7e-3*]	**10/0/0** [*4e-19, 4e-5, 5e-6*]
ads12	STDCOTRAIN	**1/1/8** [*3e-46, 8e-2, 1e-2*]	**10/0/0** [*2e-38, 1e-5, 1e-6*]	**9/1/0** [*2e-48, 6e-1, 6e-2*]
	SELFTRAIN	**3/2/5** [*2e-47, 6e-1, 7e-2*]	**5/4/1** [*6e-6, 9e-1, 3e-1*]	**9/1/0** [*7e-34, 5e-1, 5e-2*]
	SETRED	**2/5/3** [*1e-25, 5e-1, 1e-1*]	**3/7/0** [*2e-3, 9e-1, 4e-1*]	**8/2/0** [*7e-34, 5e-1, 5e-2*]
	TRAINORG	**10/0/0** [*2e-49, 2e-4, 2e-5*]	**4/5/1** [*6e-6, 9e-1, 3e-1*]	**4/6/0** [*5e-3, 8e-1, 2e-1*]
ads13	STDCOTRAIN	**10/0/0** [*9e-66, 3e-5, 3e-6*]	**10/0/0** [*7e-56, 7e-25, 7e-26*]	**10/0/0** [*2e-40, 3e-8, 3e-9*]
	SELFTRAIN	**10/0/0** [*6e-80, 4e-54, 4e-55*]	**6/3/1** [*1e-21, 3e-1, 7e-2*]	**10/0/0** [*2e-32, 3e-5, 3e-6*]
	SETRED	**10/0/0** [*6e-27, 9e-12, 9e-13*]	**7/3/0** [*8e-10, 6e-1, 1e-1*]	**10/0/0** [*1e-12, 2e-3, 3e-4*]
	TRAINORG	**10/0/0** [*2e-83, 4e-32, 6e-33*]	**7/3/0** [*1e-21, 3e-1, 7e-2*]	**10/0/0** [*7e-14, 3e-4, 5e-5*]
ads23	STDCOTRAIN	**10/0/0** [*7e-50, 8e-24, 8e-25*]	**10/0/0** [*3e-55, 9e-17, 9e-18*]	**10/0/0** [*4e-55, 2e-6, 2e-7*]
	SELFTRAIN	**10/0/0** [*2e-102, 6e-75, 6e-76*]	**10/0/0** [*5e-21, 4e-2, 4e-3*]	**10/0/0** [*4e-33, 1e-5, 10e-6*]
	SETRED	**10/0/0** [*3e-37, 2e-15, 2e-16*]	**9/1/0** [*5e-9, 8e-2, 1e-2*]	**10/0/0** [*1e-10, 5e-4, 6e-5*]
	TRAINORG	**10/0/0** [*3e-92, 3e-41, 3e-42*]	**10/0/0** [*3e-21, 4e-3, 4e-4*]	**10/0/0** [*9e-12, 1e-2, 2e-3*]
NG1	STDCOTRAIN	**0/2/8** [*1e-36, 8e-1, 1e-1*]	**8/0/2** [*3e-6, 6e-3, 2e-3*]	**2/8/0** [*5e-5, 8e-1,3e-1*]
	SELFTRAIN	**5/5/0** [*4e-5, 8e-1, 2e-1*]	**1/7/2** [*1e-3, 8e-1,3e-1*]	**7/1/2** [*5e-9, 7e-1,7e-2*]
	SETRED	**6/1/3** [*4e-5, 7e-2, 1e-2*]	**2/6/2** [*2e-2, 9e-1,3e-1*]	**6/4/0** [*2e-24, 3e-1,3e-2*]
	TRAINORG	**6/1/3** [*4e-5, 7e-2, 1e-2*]	**2/8/0** [*2e-2, 9e-1,4e-1*]	**9/1/0** [*3e-27, 1e-1,1e-2*]
NG2	STDCOTRAIN	**1/1/8** [*6e-39, 7e-1, 7e-2*]	**8/0/2** [*2e-5, 1e-3, 3e-4*]	**1/9/0** [*8e-3, 8e-1, 3e-1*]
	SELFTRAIN	**4/6/0** [*1e-4, 9e-1, 2e-1*]	**0/8/2** [*3e-2, 5e-1, 2e-1*]	**8/2/0** [*6e-6, 4e-1, 6e-2*]
	SETRED	**4/4/2** [*5e-5, 2e-1, 7e-2*]	**0/8/2** [*4e-2, 4e-1, 2e-1*]	**8/2/0** [*1e-20, 1e-1,1e-2*]
	TRAINORG	**4/4/2** [*5e-5, 2e-1, 8e-2*]	**2/8/0** [*4e-2, 4e-1, 2e-1*]	**9/1/0** [*6e-23, 5e-1,5e-2*]

on why COTRADE achieves less impressive performance on the newsgroup data sets are unclear here while worth further investigation.

From Fig. 2 to Fig. 4, in most cases, the gain of COTRADE over other comparing algorithms is more remarkable when there is relatively few examples in $\mathcal{L}$. This property of COTRADE is very attractive as when solving real-world semi-supervised learning problems, we will frequently encounter the difficulty of insufficient labeled training data. In addition, note that when STDCOTRAIN is implemented with *stable* learners[7] such as NAÏVE BAYES (Figure 2), the classification noise introduced in each round may not strongly affect its performance. However, when it is implemented with *unstable* learners such as CART (Figure 3), STDCOTRAIN would be severely impaired by the accumulated labeling noise and even be inferior to TRAINORG. On the other hand, the performance gaps between COTRADE and the other two semi-supervised learning algorithms, i.e. SELFTRAIN and SETRED, seem to be less sensitive to the choice of stable or unstable learners.

In addition to Figs. 2 to 4, we have also *quantitatively* examined the significance level of performance difference between COTRADE and other comparing algorithms. Note that given two comparing algorithms A and B, when the number of labeled training examples and classifier inducer are fixed, 100 independent runs are performed for each algorithm. Therefore, we choose to evaluate the significance level of the performance gap between two algorithms based on two-tailed pairwise t-tests. Concretely, the p-*value* returned by the two-tailed pairwise t-test can be used as a reasonable measure of *how much difference* between two algorithms' performance. The smaller the p-value is, the higher the level of performance difference is. Generally speaking, a *significant* difference is deemed to occur if the returned p-value is less than .05 (i.e. *5e-2*).

Table III reports the *win/tie/loss* counts based on statistical tests (TRAINALL is not included in the table as its performance surpasses all the other algorithms without any surprise.). For each data set and classifier inducer, a win (or loss) is counted (i.e. p<*5e-2*) when COTRADE is significantly better (or worse) than the comparing algorithm out of 100 runs, under a specific number of labeled training examples (i.e. $|\mathcal{L}|$). Otherwise, a tie is recorded. In addition, the *maximum*, *minimum*, and *average* p-values across different configurations of $|\mathcal{L}|$ are also summarized for reference purpose along with the win/tie/loss counts.

As shown in Table III, it is clear that COTRADE is superior or at least comparable to STDCOTRAIN and SELFTRAIN in most cases. Furthermore, either SETRED or TRAINORG seldom outperforms COTRADE. In summary, COTRADE is statistically superior to STDCOTRAIN, SELFTRAIN, SETRED and TRAINORG in around 68%, 71%, 67% and 76% cases, and is only inferior to them in around 18%, 7%, 7% and 3% cases.

D. Auxiliary Results

1) Supplementary Comparing Algorithms: In this subsection, the effectiveness of COTRADE is further evaluated against some other related learning approaches:

• CO-EM SVM [5]: CO-EM is one of the famous multi-view semi-supervised learning algorithms, which combines multi-view learning with the probabilistic EM procedure [30]. However, traditional CO-EM is confined to base learners which are capable of estimating posteriori probabilities such as

[7]Stable learner refers to the learning procedure where a small change in the training set will not result in large changes in its induced model [7].

TABLE IV
THE PREDICTIVE ERROR (MEAN± STD. DEVIATION) OF COTRADE AND OTHER COMPARING ALGORITHMS UNDER DIFFERENT NUMBER OF LABELED TRAINING EXAMPLES.

Data set	Algorithm	Number of labeled training examples ($\|\mathcal{L}\| = 4\alpha$ for *course* and *newsgroup*; $\|\mathcal{L}\| = 6\alpha$ for *ads*)									
		$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$	$\alpha = 6$	$\alpha = 7$	$\alpha = 8$	$\alpha = 9$	$\alpha = 10$
course	CoTrade	.17±.08	.10±.08	.09±.07	.07±.05	.07±.04	.06±.04	.06±.03	**.06±.03**	.06±.03	**.05±.03**
	Co-EM Svm	**.08±.01**	**.07±.01**	.07±.02	.07±.01	.07±.02	.07±.02	.07±.02	.07±.02	.07±.01	.06±.02
	Co-MR	.20±.06	.19±.05	.19±.05	.21±.05	.21±.06	.21±.05	.21±.06	.22±.07	.21±.06	.20±.05
	Co-Graph-1NN	.24±.09	.22±.06	.22±.05	.21±.05	.21±.05	.20±.04	.20±.04	.19±.03	.19±.03	.18±.02
	Co-Graph-3NN	.22±.00	.22±.05	.21±.02	.21±.02	.21±.03	.21±.03	.20±.01	.20±.02	.20±.02	.20±.01
ads12	CoTrade	.14±.03	.13±.03	**.11±.03**	**.10±.03**	**.10±.02**	**.09±.01**	**.09±.01**	**.09±.01**	**.09±.01**	**.09±.01**
	Co-EM Svm	.19±.20	.13±.03	.12±.01	.12±.01	.12±.02	.11±.01	.11±.01	.11±.01	.11±.01	.11±.01
	Co-MR	.21±.07	.23±.07	.28±.03	.29±.02	.29±.02	.29±.02	.29±.02	.29±.02	.29±.02	.29±.01
	Co-Graph-1NN	.73±.04	.73±.04	.73±.04	.74±.03	.74±.03	.74±.03	.74±.03	.75±.03	.75±.02	.75±.03
	Co-Graph-3NN	.14±.00	.69±.04	.69±.04	.71±.04	.72±.04	.72±.04	.72±.04	.72±.03	.73±.03	.73±.04
ads13	CoTrade	.11±.04	.10±.04	.08±.02	.08±.02	.08±.02	**.07±.02**	.07±.02	.08±.01	**.07±.01**	**.07±.01**
	Co-EM Svm	**.08±.01**	**.08±.01**	.08±.01	.08±.01	.08±.01	.08±.01	.08±.01	.08±.01	.08±.01	.08±.01
	Co-MR	.24±.03	.26±.03	.28±.02	.28±.02	.28±.02	.28±.02	.28±.01	.27±.01	.27±.02	.27±.01
	Co-Graph-1NN	.75±.05	.75±.04	.75±.04	.75±.03	.75±.03	.75±.03	.75±.03	.76±.03	.75±.03	.75±.03
	Co-Graph-3NN	.14±.00	.71±.05	.73.±04	.73±.04	.74±.04	.74±.03	.74±.03	.74±.03	.75±.03	.75±.03
ads23	CoTrade	.11±.03	**.10±.04**	**.09±.03**	**.09±.02**	**.08±.03**	**.08±.02**	**.08±.01**	**.08±.01**	**.07±.01**	**.07±.01**
	Co-EM Svm	.12±.01	.12±.01	.11±.01	.11±.01	.11±.01	.11±.01	.11±.01	.11±.01	.11±.01	.11±.01
	Co-MR	.24±.05	.26±.04	.27±.02	.28±.02	.28±.02	.28±.02	.28±.02	.28±.02	.28±.02	.28±.02
	Co-Graph-1NN	.71±.04	.72±.04	.72±.03	.72±.03	.72±.03	.73±.03	.73±.03	.73±.03	.73±.03	.74±.03
	Co-Graph-3NN	.14±.00	.68±.03	.68±.04	.70±.04	.70±.04	.71±.03	.71±.04	.72±.03	.72±.03	.72±.03
NG1	CoTrade	.49±.04	.50±.07	.49±.05	.42±.06	.41±.06	.42±.07	.39±.06	**.36±.06**	.38±.07	.37±.07
	Co-EM Svm	**.43±.06**	**.41±.06**	**.41±.05**	**.39±.05**	.40±.05	**.40±.05**	.40±.05	.39±.04	.38±.04	.38±.04
	Co-MR	.47±.05	.45±.05	.43±.05	.43±.06	.42±.06	.42±.06	.43±.06	.41±.05	.41±.05	.42±.06
	Co-Graph-1NN	.48±.02	.47±.02	.46±.02	.46±.02	.45±.02	.45±.03	.44±.02	.43±.03	.43±.03	.43±.03
	Co-Graph-3NN	.49±.02	.48±.02	.47±.02	.47±.02	.47±.02	.46±.03	.46±.03	.45±.03	.45±.03	.45±.03
NG2	CoTrade	.50±.03	.49±.05	.50±.05	.42±.07	.43±.07	.44±.08	.39±.07	.35±.05	.40±.09	.35±.07
	Co-EM Svm	**.39±.08**	**.36±.06**	**.34±.06**	**.32±.05**	**.33±.06**	**.33±.06**	**.33±.06**	**.33±.06**	**.33±.06**	**.33±.05**
	Co-MR	.47±.04	.44±.04	.42±.04	.43±.05	.42±.06	.42±.05	.43±.06	.42±.05	.43±.06	.43±.06
	Co-Graph-1NN	.48±.01	.47±.02	.47±.02	.46±.02	.45±.02	.45±.02	.43±.03	.43±.03	.43±.02	.43±.03
	Co-Graph-3NN	.49±.02	.48±.02	.47±.02	.47±.02	.47±.03	.46±.03	.45±.03	.45±.03	.45±.03	.45±.03

NAÏVE BAYES. Brefeld and Scheffer [5] broke this restriction by incorporating support vector machines into the Co-EM framework. The proposed Co-EM Svm algorithm is found to be highly competitive to other semi-supervised SVM approaches, and achieves the state-of-the-art performance on the *course* data (less than 1% error rate). In this subsection, Co-EM Svm is re-implemented and compared with CoTrade. Specifically, linear support vector machines are used as the base learners and the number of Co-EM iterations is set to 15.

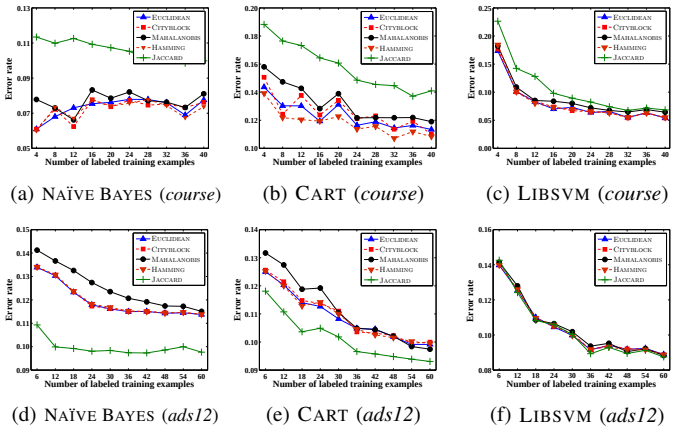
• Co-MR [36]: Recall that CoTrade estimates the labeling conference on unlabeled data based on the cut edge weight statistic, which is essentially to impose the *manifold assumption* on the constructed weighted graph. Interestingly, this is very similar in spirit to another family of algorithms which also combine the manifold smoothness assumptions with multiple views [35], [36]. Actually, the Co-MR approach [36] derives a *co-regularization kernel* by exploiting two RKHSs (Reproducing Kernel Hilbert Spaces) defined over the same input space $\mathbb{X}$, one on the "ambient representation" in $\mathbb{X}$ and another on the "intrinsic representation" in a neighborhood graph. In this subsection, Co-MR is re-implemented and compared with CoTrade. Specifically, each data set adopts an unified representation by merging the two views, and the regularization parameters γ_1, γ_2 varied on a grid of values (10^{-6}, 10^{-4}, 10^{-2}, 1, 10, 100) where the results from *optimal* configurations are reported.

• Co-Graph: As shown in Subsection V-C, CoTrade achieves highly comparable performance over the comparing algorithms, especially when the number of labeled training examples is few. One may wonder that with few labeled training examples, the weighted k-nearest neighbor graph is really doing most of the work for CoTrade, while the base learner (i.e. $Learner$ in Table I) is just a way to get a convenient *out-of-sample* prediction function. To verify whether this is the case, a simple algorithm named Co-Graph is designed which makes prediction solely based on the graph structure. Concretely, for each test example x, a weighted graph is constructed over $\mathcal{L} \cup \mathcal{U} \cup \{x\}$, and the nearest *labeled neighbors* for x are identified. Here the distance between two examples is the graph distance, i.e. the sum of weights along the shortest path between them. Two implementations of Co-Graph are studied, i.e. to predict the label of the nearest labeled neighbor (Co-Graph-1NN) or the majority vote of 3 nearest labeled neighbors (Co-Graph-3NN). For each data set, results from the view with better performance are reported.

Considering that both Co-EM Svm and Co-MR are kernel-based approaches and Co-Graph doesn't involve any specific learning procedure, we choose to compare their performance with CoTrade implemented with LibSVM. Furthermore, due to the distinctions in data pre-processing and experimental setup, the performance of Co-EM Svm reported here would be a bit different from those reported in literature [5].

Table IV reports the error rate (mean±std. deviation) of each comparing algorithm under different number of labeled training examples. When the data set and the number of labeled training examples are fixed, the performance of one algorithm is shown in boldface if it significantly outperforms all the other algorithms (two-tailed pairwise t-test at .05 significance level).

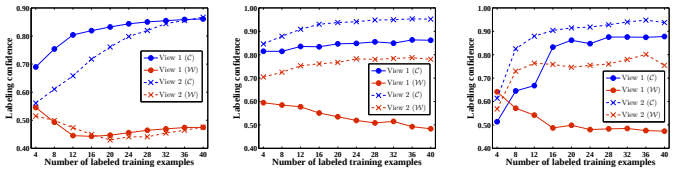
As shown in Table IV, CoTrade achieves close perfor-



(a) NAÏVE BAYES (*course*) (b) CART (*course*) (c) LIBSVM (*course*)

(d) NAÏVE BAYES (*ads12*) (e) CART (*ads12*) (f) LIBSVM (*ads12*)

Fig. 5. Classification error rate of COTRADE with different distance measures (on the *course* and *ads12* data sets) changes as the number of labeled training examples increases.



(a) NAÏVE BAYES (b) CART (c) LIBSVM

Fig. 6. Estimated confidence of COTRADE (on the *course* data set) for unlabeled examples which are correctly predicted (drawn in blue and denoted as $\mathcal{C}$) and wrongly predicted (drawn in brown and denoted as $\mathcal{W}$).

mance to CO-EM SVM while outperforms CO-MR in most cases. In addition, COTRADE is also superior to CO-GRAPH-1NN and CO-GRAPH-3NN in almost all cases, and the two implementations of CO-GRAPH totally fail on the three *ads* data sets. In summary, COTRADE is statistically superior to CO-EM SVM, CO-MR, CO-GRAPH-1NN and CO-GRAPH-3NN in around 40%, 82%, 88% and 88% cases, and is only inferior to them in around 32%, 12%, 10% and 8% cases.

2) Graph Distance Measure: In Subsection V-C, EUCLIDEAN distance is employed to construct COTRADE's k-nearest neighborhood graph. In this subsection, we further investigate how COTRADE performs with other forms of distance measures. Given two d-dimensional feature vectors $\boldsymbol{a} = (a_1, a_2, \ldots, a_d)^{\mathrm{T}}$ and $\boldsymbol{b} = (b_1, b_2, \ldots, b_d)^{\mathrm{T}}$, the following four distance measures are considered here:

- CITYBLOCK distance $d_{\mathrm{C}}(\boldsymbol{a}, \boldsymbol{b}) = \sum_{j=1}^{d} |a_j - b_j|$, which is actually the *first order* Minkowski distance and also known as MANHATTAN distance;
- MAHALANOBIS distance $d_{\mathrm{M}}(\boldsymbol{a}, \boldsymbol{b}) = \sqrt{(\boldsymbol{a}-\boldsymbol{b})^{\mathrm{T}} \cdot \mathbf{S}^{-1} \cdot (\boldsymbol{a}-\boldsymbol{b})}$, here "$\mathbf{S}$" is the *covariance matrix* of the feature vectors;
- HAMMING distance $d_{\mathrm{H}}(\boldsymbol{a}, \boldsymbol{b}) = \frac{(\mathbf{1}-\boldsymbol{a})^{\mathrm{T}}\boldsymbol{b} + \boldsymbol{a}^{\mathrm{T}}(\mathbf{1}-\boldsymbol{b})}{d}$, which measures the percentage of *binary features* that differ. Here, "$\mathbf{1}$" represents the d-dimensional vector with all ones.
- JACCARD distance $d_{\mathrm{J}}(\boldsymbol{a}, \boldsymbol{b}) = \frac{(\mathbf{1}-\boldsymbol{a})^{\mathrm{T}}\boldsymbol{b} + \boldsymbol{a}^{\mathrm{T}}(\mathbf{1}-\boldsymbol{b})}{\boldsymbol{a}^{\mathrm{T}}\mathbf{1} + \boldsymbol{b}^{\mathrm{T}}\mathbf{1} - \boldsymbol{a}^{\mathrm{T}}\boldsymbol{b}}$, which measures the percentage of *binary features* that differ out of all features that are *nonzero* in both vectors.

Fig. 5 illustrates how COTRADE performs under various distance measures as the number of labeled training examples increases. Without loss of generality, results on the *course* and *ads12* data sets are reported. Each point in the plot gives the average classification error rate of the comparing measure out of 100 independent runs.

As shown in Fig. 5, EUCLIDEAN distance yields superior or at least comparable performance to MAHALANOBIS distance in most cases, and is almost indistinguishable to CITYBLOCK and HAMMING distances. Note that although JACCARD distance has behaved quite well on the *ads12* data set (second row), its performance is nearly the worst on the *course* data set (first row). These results show that although it seems hard to tell which distance measure could be the best choice, EUCLIDEAN distance is at least a relatively robust choice for neighborhood graph construction.

Here we choose to employ EUCLIDEAN distance as the distance measure mainly based on its simplicity and empirical evidences, while justifying this choice from theoretical point of view may provide more insightful explanations for the success of COTRADE. Generally, the problem of choosing the best distance measure for a specific learning task is very difficult, and a number of efforts have been made towards tackling this problem under the name of *distance metric learning* [42]. How to identify or learn the optimal distance measure for COTRADE and how does it affect the performance of the algorithm are worth further investigation.

VI. DISCUSSION

In this section, the underlying reasons for COTRADE's good performance is further explored. The exploration is accomplished in two different ways: one is to closely inspect the labeling confidence estimated by COTRADE on unlabeled examples (as defined in Eq.(4)), and the other is to conduct bias-variance (BV) decomposition [13] on the comparing algorithms.

Fig. 6 gives the estimated labeling confidence for unlabeled examples on COTRADE's *first round* of co-training on the *course* data. Similar conclusions can be drawn based on the results on other data sets, which are not reported here for brevity. When the number of labeled training examples and classifier inducer are fixed, each point in the plot corresponds to the average confidence value of newly labeled examples over 100 runs.

It is obvious from Fig. 6 that on either view of each data set, when the classifier inducer is fixed, COTRADE will give much larger confidence estimates to labels *correctly* predicted than those *wrongly* predicted. Two-tailed pairwise t-test at .05 significance level reveals that in nearly all cases (>99%), the labeling confidence estimated for correctly labeled examples are significantly larger than those estimated for wrongly labeled examples.

Note that as stated in Subsection IV-A, the use of labeling confidence should *not* be regarded as an estimate for the probability of an example being correctly or wrongly labeled. While on the other hand, recall that COTRADE conducts labeling

information exchange between each classifier in the order of *descending* confidence values, the apparent gaps between the labeling confidence of correct and incorrect predictions in the first round will definitely benefit the following training process of CoTrade.

In addition to the above discussion, we further investigate the properties of CoTrade by exploiting techniques of BV decomposition, which is a rather useful tool to understand the behavior of machine learning algorithms [13]. Roughly speaking, this technique decomposes the expected error of one learning algorithm (under fixed training set size) into three terms, i.e. the *intrinsic noise* corresponding to the expected loss of Bayesian optimal classifier, the (squared) *bias* measuring the degree of match between the algorithm's average output and the target, and the *variance* measuring the sensitivity of the learning algorithm w.r.t. different training sets. For a specific problem, the smaller the values of bias and variance, the better the performance of the learning algorithm.

We have conducted BV decomposition analysis between two algorithms, i.e. CoTrade and StdCoTrain, on the *course* and *ads* data. For brevity, results on the other comparing algorithms are not included here but won't affect the major conclusions of our analysis. As our algorithm makes binary predictions, in this paper, the popular BV decomposition approach proposed by Kohavi and Wolpert for zero-one loss function [19] is used.

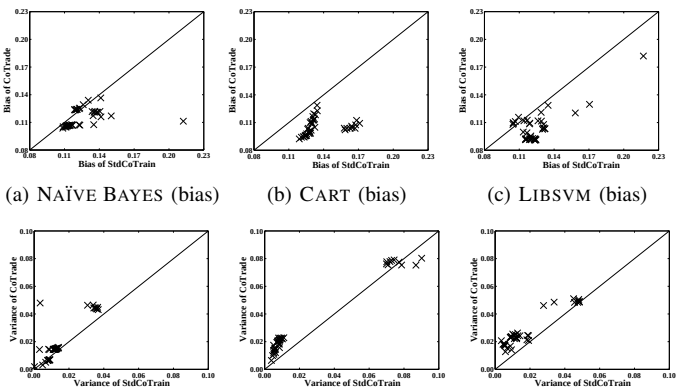
Concretely, let $\mathbb{X}$ be the instance space with probability distribution function $\mathbf{D}$. Furthermore, let $P_\theta(y|x)$ be the posteriori probability of example $x \in \mathbb{X}$ having label $y \in \{0, 1\}$ for the *target function* f, and $P(y|f, m, x)$ be the posteriori probability of example x being predicted with label y given the target function f and a training set with size m. Then, the expected error of a learning algorithm $\mathcal{A}$ can be decomposed as follows:

$$E(\mathcal{A}) = \int_{x\in\mathbb{X}} \left(\sigma_x^2 + \text{bias}_x^2 + \text{variance}_x\right) \cdot \mathbf{D}(x)\ \mathbf{d}x, \quad \text{where}$$

$$\begin{aligned} \sigma_x^2 &\equiv \frac{1}{2}\left(1 - \sum\nolimits_{y\in\{0,1\}} P_\theta(y|x)^2\right) \\ \text{bias}_x^2 &\equiv \frac{1}{2}\sum\nolimits_{y\in\{0,1\}} \left[P_\theta(y|x) - P(y|f,m,x)\right]^2 \\ \text{variance}_x &\equiv \frac{1}{2}\left(1 - \sum\nolimits_{y\in\{0,1\}} P(y|f,m,x)^2\right) \end{aligned} \quad (13)$$

Here, σ_x^2 represents the intrinsic noise of f, and the remaining *bias* and *variance* terms, i.e. bias_x^2 and variance_x, are estimated via a *frequency-based* procedure [19]. Accordingly, Fig. 7 illustrates the scatter plots between CoTrade and StdCoTrain in terms of bias and variance. Each marker '$\times$' in the scatter plots is derived based on 100 runs.

As shown in the *first* row of Fig. 7, most points are under the diagonal indicating that CoTrade performs better than StdCoTrain in terms of *bias*. On the contrary, as shown in the *second* row of Fig. 7, most points are above the diagonal indicating that CoTrade performs worse than StdCoTrain in terms of *variance*. Therefore, we suppose that compared to other co-training style algorithms, CoTrade can *largely* reduce the algorithm's bias, at the cost of *slightly* increasing the variance by a smaller magnitude (about $1/5 \sim 1/2$). This would be one of the possible explanations for the success of our proposed approach.



(a) Naïve Bayes (bias) (b) CART (bias) (c) LIBSVM (bias)

(d) Naïve Bayes (variance) (e) CART (variance) (f) LIBSVM (variance)

Fig. 7. Scatter plots between CoTrade and StdCoTrain in terms of *bias* (first row) and *variance* (second row).

VII. Conclusion

For co-training style algorithms, one key factor for their success is how to choose predictions with authentic high confidence for labeling information communication. In this paper, based on particular data editing techniques, we propose the CoTrade algorithm which can explicitly and reliably estimate the labeling confidence of the classifiers' outputs. Experiments show that our algorithm can effectively exploit unlabeled data in training, especially when few labeled examples are available. Possible explanations for CoTrade's good performance are also discussed.

In the future, it is very important to conduct more insightful theoretical analyses on the effectiveness of CoTrade. Furthermore, designing other kinds of methods to effectively estimate labeling confidence is also worth studying.

Acknowledgment

The authors want to thank the anonymous reviewers for helpful comments.

References

[1] D. Angluin and P. Laird, "Learning from noisy examples," *Machine Learning*, vol. 2, no. 4, pp. 343–370, 1988.

[2] M. Anthony and N. Biggs, *Computational Learning Theory: An Introduction*. Cambridge: Cambridge University Press, 1992.

[3] M.-F. Balcan, A. Blum, and K. Yang, "Co-training and expansion: Towards bridging theory and practice," in *Advances in Neural Information Processing Systems 17*, L. Saul, Y. Weiss, and L. Bottou, Eds. Cambridge, MA: MIT Press, 2005, pp. 89–96.

[4] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the 11th Annual Conference on Computational Learning Theory*, Madison, WI, 1998, pp. 92–100.

[5] U. Brefeld and T. Scheffer, "Co-EM support vector learning," in *Proceedings of the 21st International Conference on Machine Learning*, Banff, Canada, 2004, pp. 121–128.

[6] ——, "Semi-supervised learning for structured output variables," in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006, pp. 145–152.

[7] L. Breiman, "Heuristics of instability and stabilization in model selection," *Annals of Statistics*, vol. 24, no. 6, pp. 2350–2383, 1996.

[8] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984.

[9] C. E. Brodley and M. A. Friedl, "Identifying and eliminating mislabeled training instances," *Journal of Artificial Intelligence Research*, vol. 11, pp. 131–167, 1999.

[10] C.-C. Chang and C.-J. Lin, *LIBSVM: A library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm.

[11] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006.

[12] S. Dasgupta, M. Littman, and D. McAllester, "PAC generalization bounds for co-training," in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, pp. 375–382.

[13] P. Domingos, "A unified bias-variance decomposition and its applications," in *Proceedings of the 17th International Conference on Machine Learning*, Standford, CA, 2000, pp. 231–238.

[14] J. Du, C. X. Ling, and Z.-H. Zhou, "When does co-training work in real data?" *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 5, pp. 788–799, 2011.

[15] B. Efron and R. Tibshirani, *An Introduction to the Boostrap*. New York: Chapman & Hall, 1993.

[16] S. Goldman and Y. Zhou, "Enhancing supervised learning with unlabeled data," in *Proceedings of the 17th International Conference on Machine Learning*, Standord, CA, 2000, pp. 327–334.

[17] Y. Jiang and Z.-H. Zhou, "Editing training data for *k*NN classifiers with neural network ensemble," in *Lecture Notes in Computer Science 3173*, F. Yin, J. Wang, and C. Guo, Eds. Berlin: Springer, 2004, pp. 356–361.

[18] T. Joachims, "A probabilistic analysis of the rocchio algorithm with tfidf for text categorization," in *Proceedings of the 14th International Conference on Machine Learning*, Nashville, TN, 1997, pp. 143–151.

[19] R. Kohavi and D. Wolpert, "Bias plus variance decomposition for zero-one loss functions," in *Proceedings of the 13th International Conference on Machine Learning*, Bari, Italy, 1996, pp. 275–283.

[20] J. Koplowitz and T. A. Brown, "On the relation of performance to editing in nearest neighbor rules," *Pattern Recognition*, vol. 13, no. 3, pp. 251–255, 1981.

[21] N. Kushmerick, "Learning to remove internet advertisements," in *Proceedings of the 3rd International Conference on Autonomous Agents*, Seattle, WA, 1999, pp. 175–181.

[22] P. Laird, "Learning from good data and bad," Ph.D. dissertation, Department of Computer Science, Yale University, 1987.

[23] M. Li, H. Li, and Z.-H. Zhou, "Semi-supervised document retrieval," *Information Processing and Management*, vol. 45, no. 3, pp. 341–355, 2009.

[24] M. Li and Z.-H. Zhou, "SETRED: Self-training with editing," in *Lecture Notes in Artificial Intelligence 3518*, T. B. Ho, D. Cheung, and H. Liu, Eds. Berlin: Springer, 2005, pp. 611–621.

[25] ——, "Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 37, no. 6, pp. 1088–1098, 2007.

[26] D. Mavroeidis, K. Chaidos, S. Pirillos, D. Christopoulos, and M. Vazirgiannis, "Using tri-training and support vector machines for addressing the ecml-pkdd 2006 discovery challenge," in *Proceedings of ECML-PKDD 2006 Discovery Challenge Workshop*, Berlin, Germany, 2006, pp. 39–47.

[27] F. Muhlenbach, S. Lallich, and D. A. Zighed, "Identifying and handling mislabeled instances," *Journal of Intelligent Information Systems*, vol. 22, no. 1, pp. 89–109, 2004.

[28] I. Muslea, S. Minton, and C. A. Knoblock, "Active + semi-supervised learning = robust multi-view learning," in *Proceedings of the 19th International Conference on Machine Learning*, Sydney, Australia, 2002, pp. 435–442.

[29] ——, "Active learning with multiple views," *Journal of Artificial Intelligence Research*, vol. 27, pp. 203–233, 2006.

[30] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *Proceedings of the 9th ACM International Conference on Information and Knowledge Management*, McLean, VA, 2000, pp. 86–93.

[31] D. Pierce and C. Cardie, "Limitations of co-training for natural language learning from large data sets," in *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, Pittsburgh, PA, 2001, pp. 1–9.

[32] J. S. Sánchez, R. Barandela, A. I. Marqués, R. Alejo, and J. Badenas, "Analysis of new techniques to obtain quality training sets," *Pattern Recognition Letters*, vol. 24, no. 7, pp. 1015–1022, 2003.

[33] A. Sarkar, "Applying co-training methods to statistical parsing," in *Proceedings of the 2nd Annual Meeting of the North American Chapter of the Association of Computational Linguistics*, Pittsburgh, PA, 2001, pp. 95–102.

[34] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[35] V. Sindhwani, P. Niyogi, and M. Belkin, "A co-regularization approach to semi-supervised learning with multiple views," in *Proceedings of the ICML'05 Workshop on Learning with Multiple Views*, Bonn, Germany, 2005.

[36] V. Sindhwani and D. S. Rosenberg, "An RKHS for multi-view learning and manifold co-regularization," in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008, pp. 976–983.

[37] M. Steedman, M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim, "Bootstrapping statistical parsers from small data sets," in *Proceedings of 11th Conference of the European Chapter of the Association for Computational Linguistics*, Budapest, Hungary, 2003, pp. 331–338.

[38] W. Wang and Z.-H. Zhou, "Analyzing co-training style algorithms," in *Lecture Notes in Artificial Intelligence 4701*, J. N. Kok, J. Koronacki, R. L. de Mantaras, S. Matwin, D. Mladenic, and A. Skowron, Eds. Berlin: Springer, 2007, pp. 454–465.

[39] ——, "On multi-view active learning and the combination with semi-supervised learning," in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008, pp. 1152–1159.

[40] ——, "A new analysis of co-training," in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010, pp. 1135–1142.

[41] D. R. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Transactions on Systmes, Man and Cybernetics*, vol. 2, pp. 408–421, 1972.

[42] L. Yang and R. Jin, "Distance metric learning: A comprehensive survey," Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, Tech. Rep., 2006.

[43] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proceedings of the 14th International Conference on Machine Learning*, Nashville, TN, 1997, pp. 412–420.

[44] Y. Zhou and S. Goldman, "Democratic co-learning," in *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, Boca Raton, FL, 2004, pp. 594–602.

[45] Z.-H. Zhou, K.-J. Chen, and H.-B. Dai, "Enhancing relevance feedback in image retrieval using unlabeled data," *ACM Transactions on Information Systems*, vol. 24, no. 2, pp. 219–244, 2006.

[46] Z.-H. Zhou and M. Li, "Tri-training: Exploiting unlabeled data using three classifiers," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 11, pp. 1529–1541, 2005.

[47] ——, "Semi-supervised regression with co-training style algorithms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 11, pp. 1479–1493, 2007.

[48] ——, "Semi-supervised learning by disagreement," *Knowledge and Information Systems*, vol. 24, no. 3, pp. 415–439, 2010.

[49] Z.-H. Zhou, D.-C. Zhan, and Q. Yang, "Semi-supervised learning with very few training examples," in *Proceedings of the 22nd National Conference on Artificial Intelligence*, Vancouver, Canada, 2007, pp. 675–680.

[50] X. Zhu, "Semi-supervised learning literature survey," Department of Computer Science, University of Wisconsin at Madison, Madison, WI, Tech. Rep. 1530, 2006.

[51] D. A. Zighed, S. Lallich, and F. Muhlenbach, "Separability index in supervised learning," in *Lecture Notes in Artificial Intelligence 2431*, T. Elomaa, H. Mannila, and H. Toivonen, Eds. Berlin: Springer, 2002, pp. 475–487.