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-TRAnining 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
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-aided 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 [17] 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 \( \mathcal{X} \) be the input space and \( \mathcal{Y} = \{0, 1\} \) be the output space. Under standard co-training setting, the input space is partitioned into two different views \( \mathcal{X}^1 \) and \( \mathcal{X}^2 \), i.e. \( \mathcal{X} = \mathcal{X}^1 \times \mathcal{X}^2 \). For any example \( x \in \mathcal{X} \), we use \( x^1 \) and \( x^2 \) to denote its two portions under the first view \( \mathcal{X}^1 \) and the second view \( \mathcal{X}^2 \) respectively, i.e. \( x = (x^1, x^2) \). 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 \( v_i = (x^1_i, x^2_i) \in \mathcal{X} \), \( u_j = (x^1_j, x^2_j) \in \mathcal{X} \) and \( y_i \in \mathcal{Y} \).

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 \( \mathcal{X}^1 \), i.e. \( \mathcal{L}_1 = \{(v^1_i, y_i)| i = 1, 2, \ldots, L\} \) and \( \mathcal{U}_1 = \{u^1_j| j = 1, 2, \ldots, U\} \). The corresponding sets \( \mathcal{L}_2 \) and \( \mathcal{U}_2 \) with respect to view \( \mathcal{X}^2 \) 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, Mullenbach 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 $X^i$ based on $L_i$ ($i = 1, 2$);
- **Repeat**
  - Apply classifier $f_i$ to predict labels of unlabeled examples in $U_i$ ($i = 1, 2$);
  - 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 $Z = \{(z_p, y_p) | p = 1, 2, \ldots, Z\}$, 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 $(z_p, y_p) \in Z$ corresponds to a vertex in the graph $G_Z$. There will be an edge $\overline{pq}$ connecting the two vertices of $z_p$ and $z_q$ if either $z_p$ is among the $k$-nearest neighbors of $z_q$ 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 $\overline{pq}$ 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 $G_Z$’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 $G_Z$. 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 $G_Z$ 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 $D(Y) = \{Pr(y = 1), Pr(y = 0)\}$. 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 $Z$.

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

$$J_p = \sum_{z_q \in C_p} w_{pq} I_{pq}$$ (1)

Here, $C_p$, corresponds to the set of examples which are connected with $z_p$ in $G_Z$. 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 $y_p$ is different to $y_q$. 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 $\sigma^2_{p|H_0}$:

$$\mu_{p|H_0} = (1 - Pr(y = y_p)) \sum_{z_q \in C_p} w_{pq}$$ (2)

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

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

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 \( J_p^* \), the higher the confidence of \( y_p \) being the correct label of \( z_p \). Therefore, based on the left unilateral p-value of \( J_p^* \) with respect to \( N(0,1) \), we can calculate the labeling confidence of \( (z_p, y_p) \) as follows:

\[
CF(z_p, y_p) = 1 - \Phi(J_p^*)
\]

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

Note that \( CF(z, y) \) 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 \( CF(z, y) \) 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 \( \mathcal{X}_1 \), 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 talk to 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 \( \mathcal{Z} \) be the instance space with probability distribution function \( \mathcal{D} \), namely \( \int_{z \in \mathcal{Z}} \mathcal{D}(z) dz = 1 \). 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 \( \mathcal{Z} \) to the output space \( \{0, 1\} \). Let \( \mathcal{H}_0 \in \mathcal{H} \) be the target (ground-truth) hypothesis to be learned, and \( \rho = \{(z_p, y_p) | 1 \leq p \leq m\} \) be a sequence of \( m \) labeled instances with random classification noise. Here, each \( z_p \) is independently drawn from \( \mathcal{Z} \) with respect to distribution \( \mathcal{D} \). Each label \( y_p \) is assumed to subject to a classification noise process with noise rate \( \eta \), i.e. \( y_p \) takes the correct label \( H_0(z_p) \) with probability \( 1 - \eta \) while the wrong label \( 1 - H_0(z_p) \) with probability \( \eta \).

Let \( dis(H_i, H_0) = \Pr_{z \sim \mathcal{D}}(z | H_i(z) \neq H_0(z)) \) denote the error rate of \( H_i \) with respect to \( H_0 \). Furthermore, let \( H_* \in \mathcal{H} \) be the hypothesis which has minimum disagreement with the sequence \( \rho \), i.e. \( H_* = \arg \min_{H \in \mathcal{H}} \sum_{p=1}^{m} [H(z_p) \neq y_p] \). Then, given the tolerance parameter \( \epsilon \), the confidence parameter \( \delta \), and the upper bound on the noise rate \( \eta \), the following theorem states the PAC property of learning from noisy examples:

**Theorem 1** (Angluin & Laird [1], 1988)

Given a sequence \( \rho \) of \( m \) independently drawn labeled instances, when the sample size \( m \) satisfies:

\[
m \geq \frac{2}{\epsilon^2(1-2\eta^b)^2} \ln \left( \frac{2N}{\delta} \right)
\]

Then, the hypothesis \( H_* \) which minimizes the disagreement with \( \rho \) will have the PAC property:

\[
Pr[dis(H_*, H_0) \geq \epsilon] \leq \delta.
\]

Here \( Pr[\cdot] \) is evaluated over all the possible sequences of \( \rho \) with length \( m \).

In other words, under specific level of noise rate (i.e. \( \eta \)), Eq.(5) specifies how many noisy labeled instances (i.e. \( m \)) are needed to learn a classifier with expected low error rate \( \epsilon \) 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 \( \mathcal{X}_1 \) and view \( \mathcal{X}_2 \) respectively, let \( f^*(\mathcal{S}) = \{(s,f(s)) | s \in \mathcal{S}\} \) 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 \( f_1^*(\mathcal{U}_1) \) and \( f_2^*(\mathcal{U}_2) \) will be explicitly estimated by conducting data editing on \( \mathcal{L}_1 \cup f_1^*(\mathcal{U}_1) \) and \( \mathcal{L}_2 \cup f_2^*(\mathcal{U}_2) \) 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 \( \mathcal{L}_2 \cup f_1^*(\mathcal{U}_1') \).

\[1\text{For any predicate } \pi, [\pi] = 1 \text{ if } \pi \text{ holds. Otherwise, } [\pi] = 0.\]
The set of labeled examples \( f_1^\triangle (U'_1) \) is defined in similar ways.

As \( f_1^\triangle (U'_1) \) usually contains noisy labels communicated from \( f_1 \), the task of updating \( f_2 \) based on \( L_2 \cup f_1^\triangle (U'_1) \) can be treated as the process of learning from examples with classification errors. Resorting to Eq.(5), by fixing \( N, \delta \) and letting \( c = 2 \ln(2N/\delta) \), the least accommodable hypothesis classification error \( \epsilon \) given \( m \) and \( \eta^b \) will be:

\[
\epsilon = \sqrt{\frac{c}{m(1 - 2\eta^b)^2}}
\]  

(8)

When learning from \( L_2 \cup f_1^\triangle (U'_1) \), the sample size \( m \) as shown in Eq.(8) becomes:

\[
m_{U'_1} = |L_2 \cup f_1^\triangle (U'_1)| = |L_2| + |U'_1|
\]  

(9)

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

\[
\eta^b_{U'_1} = \frac{1}{m_{U'_1}} \sum_{u^i \in U'_1} \left( 1 - \text{CF}_{L_2 \cup f_1^\triangle (U'_1)} (u^i, f_1(u^i)) \right)
\]  

(10)

Here, \( \text{CF}_{L_2 \cup f_1^\triangle (U'_1)} (u^i, f_1(u^i)) \) corresponds to the labeling confidence of a newly labeled example \((u^i, f_1(u^i)) \in L_1 \cup f_1^\triangle (U'_1) \) as defined in Eq.(4). 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 \( L_2 \) is assumed to be noise-free.

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

\[
\epsilon_{U'_1} = \sqrt{c \cdot \sqrt{L + |U'_1|}} - 2 \sum_{u^i \in U'_1} \left( 1 - \text{CF}_{L_2 \cup f_1^\triangle (U'_1)} (u^i, f_1(u^i)) \right)
\]  

(11)

By keeping class distribution in \( f_2^\triangle (U'_1) \) the same as that in \( L_1 \), we can generate a series of candidate unlabeled data sets \( \Xi_1 = \{ U'_1^\xi | \xi \in \mathbb{N} \} \) to constitute supplementary labeled examples \( f_2^\triangle (U'_1^\xi) \) for \( f_2 \). Let \( \gamma \) be the ratio of the number of negative examples to the number of positive examples in \( L_1 \). Without loss of generality, we can assume that \( \gamma \) is greater than 1. Then, \( U'_1^\xi \) is formed by choosing \( \xi \) examples in \( U_1' \) with highest labeling confidence of being positive and \( \gamma \cdot \xi \) examples in \( U_1' \) with highest labeling confidence of being negative, if exist. COTRADE identifies optimal choice \( U'_1^* \in \Xi_1 \) for labeling information exchange which would yield smallest expected classification error \( \epsilon_1 \):

\[
U'_1^* = \arg \min_{U'_1^\xi \in \Xi_1} \epsilon_{U'_1^\xi}, \quad \epsilon_1 = \epsilon_{U'_1^*}
\]  

(12)

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

Thereafter, if \( \epsilon_1 \) is smaller than its previous value \( \epsilon'_1 \) determined in preceding round, \( f_2 \) will be updated based on \( L_2 \) together with the identified optimal choice, i.e., \( L_2 \cup f_1^\triangle (U'_1^*) \). 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 \( \epsilon_1 \) before COTRADE launches its co-training iteration is set to be \( 1/\sqrt{L} \) (i.e., \( \epsilon_0 \)). 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].
C. Iterative Procedure

To sum up, Table I presents the pseudo-code of the proposed algorithm. As for input parameters, $L_1$, $L_2$, $U_1$, and $U_2$ 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 $L_1$ and $L_2$ (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 undesirable 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) $NG_1$: 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) $NG_2$: 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 1 if the the example contains the corresponding word and set to 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. $NG_1$ and $NG_2$, 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

---


\(^3\)Data available at http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes/mini_newsgroup.tar.gz

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

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 augments 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 techniques5.

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 Naive BAYES, nonmetric-style method of decision trees (CART implementation [8]) and kernel-style method (LIBSVM implementation [10]) are utilized. Note that besides Naive 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 1p3n examples for the course data, 1p5n examples for ads and 1p15n examples for ads12.

For Naive 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 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.
the ads data, and pin 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 \textsc{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 (\textit{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 \textsc{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 \textsc{CoTrade} out of 100 independent runs.

As shown in Figs. 1(a) to 1(c), in most cases, the performance of \textsc{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 \textsc{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 \textsc{TrainAll} which serves as the performance upper bound) consistently outperforms \textsc{CoTrade}. Concretely, on the course data set (plot (a) of each figure), \textsc{CoTrade} is consistently superior to \textsc{StdCoTrain}, \textsc{SelfTrain}, \textsc{Setred} and \textsc{TrainOrg} when either \textsc{Naive Bayes} or \textsc{CART} is incorporated as the classifier inducer. With \textsc{Libsvm}, \textsc{CoTrade} consistently outperforms \textsc{SelfTrain}, \textsc{Setred} and \textsc{TrainOrg} and is only inferior to \textsc{StdCoTrain} as the number labeled training examples exceeds 16.

On the ads12 data set (plot (b) of each figure), \textsc{CoTrade} is less distinguishable from \textsc{Setred} and \textsc{TrainOrg} with all classifier inducers. It is consistently superior to \textsc{SelfTrain} with \textsc{Libsvm} and consistently superior to \textsc{StdCoTrain} with \textsc{CART} and \textsc{Libsvm}. Note that the two views associated with this data set, i.e. \textit{image url} and \textit{base url}, may be strongly correlated due to the co-occurrence of domain names. For instance, for a tiger image, the \textit{image url} and \textit{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 \textsc{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), \textsc{CoTrade} consistently outperforms \textsc{StdCoTrain}, \textsc{SelfTrain}, \textsc{Setred} and \textsc{TrainOrg} with all classifier inducers; On the NG1 and NG2 data sets (plots (e) and (f) of each figure), \textsc{CoTrade} is inferior to \textsc{StdCoTrain} with \textsc{Naive Bayes}, superior to \textsc{StdCoTrain} with \textsc{CART}, and nearly indistinguishable to \textsc{StdCoTrain} with \textsc{Libsvm}. \textsc{CoTrade} is also less distinguishable to \textsc{SelfTrain}, \textsc{Setred} and \textsc{TrainOrg} with either \textsc{Naive Bayes} or \textsc{CART}, while is slightly superior to them with \textsc{Libsvm}. Reasons
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 such as Naïve Bayes (Figure 2), the classification noise introduced in each round may not significantly 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., $5\%$).

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., $p<5\%$) 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

<table>
<thead>
<tr>
<th>Data set</th>
<th>CoTRADE against</th>
<th>Base learner (win/tie/loss [min. $p$-value, max. $p$-value, ave. $p$-value])</th>
</tr>
</thead>
<tbody>
<tr>
<td>course</td>
<td>StdCoTRAIN</td>
<td>Naïve Bayes: 0/0/0 [5e-59, 5e-5, 9e-6] CART: 0/0/0 [5e-59, 5e-5, 9e-6] LIBSVM: 0/0/0 [5e-59, 5e-5, 9e-6]</td>
</tr>
<tr>
<td></td>
<td>SelfTRAIN</td>
<td>0/0/0 [5e-59, 5e-5, 9e-6] 9/1/0 [4e-19, 9e-2, 1e-2] 0/0/0 [5e-59, 5e-5, 9e-6]</td>
</tr>
<tr>
<td></td>
<td>SetRED</td>
<td>0/0/0 [5e-59, 5e-5, 9e-6] 9/1/0 [4e-19, 9e-2, 1e-2] 0/0/0 [5e-59, 5e-5, 9e-6]</td>
</tr>
<tr>
<td></td>
<td>TrainORG</td>
<td>0/0/0 [5e-59, 5e-5, 9e-6] 9/1/0 [4e-19, 9e-2, 1e-2] 0/0/0 [5e-59, 5e-5, 9e-6]</td>
</tr>
</tbody>
</table>

---

5Stable 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].
NÁIVE 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 X, one on the “representation ability” in X and another on the “intrinsinc 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 \( \gamma_1, \gamma_2 \) varied on a grid of values \( (10^{-4}, 10^{-4}, 10^{-2}, 1, 10, 100) \) 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 \( \mathcal{U} \cup \{x\} \), 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 \( \pm \) 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-
rance 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 \( a = (a_1, a_2, \ldots, a_d)^T \) and \( b = (b_1, b_2, \ldots, b_d)^T \), the following four distance measures are considered here:

- **Cityblock** distance \( d_C(a, b) = \sum_{j=1}^{d} |a_j - b_j| \), which is actually the first order Minkowski distance and also known as Manhattan distance;
- **Mahalanobis** distance \( d_M(a, b) = \sqrt{(a - b)^T S^{-1} (a - b)} \), here “S” is the covariance matrix of the feature vectors;
- **Hamming** distance \( d_H(a, b) = \frac{(1-a)^T b + a^T (1-b)}{d} \), which measures the percentage of binary features that differ. Here, “1” represents the d-dimensional vector with all ones.
- **Jaccard** distance \( d_J(a, b) = \frac{(1-a)^T b + a^T (1-b)}{a^T 1 + b^T 1 - a^T b} \), 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 $\mathcal{X}$ be the instance space with probability distribution function $\mathbf{D}$. Furthermore, let $P_{\theta}(y|x)$ be the posteriori probability of example $x \in \mathcal{X}$ having label $y \in \{0, 1\}$ for the target function $f$, and $P(y|f, m, x)$ be the posterior 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:

$$E(\mathcal{A}) = \int_{x \in \mathcal{X}} (\sigma_x^2 + \text{bias}_x^2 + \text{variance}_x) \cdot \mathbf{D}(x) \, dx,$$

where

$$\sigma_x^2 = \frac{1}{2} \left( 1 - \sum_{y \in \{0, 1\}} P_{\theta}(y|x)^2 \right),$$

$$\text{bias}_x^2 = \frac{1}{2} \sum_{y \in \{0, 1\}} [P_{\theta}(y|x) - P(y|f, m, x)]^2,$$

$$\text{variance}_x = \frac{1}{2} \left( 1 - \sum_{y \in \{0, 1\}} P(y|f, m, x)^2 \right).$$

Here, $\sigma_x^2$ represents the intrinsic noise of $f$, and the remaining bias and variance terms, i.e. $\text{bias}_x^2$ and $\text{variance}_x$, 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.

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
