Tri-Training: Exploiting Unlabeled Data Using Three Classifiers

Zhi-Hua Zhou, *Member, IEEE* and Ming Li

***Abstract*— In many practical data mining applications such as web page classification, unlabeled training examples are readily available but labeled ones are fairly expensive to obtain. Therefore, semi-supervised learning algorithms such as *co-training* have attracted much attention. In this paper, a new co-training style semi-supervised learning algorithm named *tri-training* is proposed. This algorithm generates three classifiers from the original labeled example set. These classifiers are then refined using unlabeled examples in the tri-training process. In detail, in each round of tri-training, an unlabeled example is labeled for a classifier if the other two classifiers agree on the labeling, under certain conditions. Since tri-training neither requires the instance space be described with *sufficient and redundant views* nor does it put any constraints on the supervised learning algorithm, its applicability is broader than that of previous co-training style algorithms. Experiments on UCI data sets and application to the web page classification task indicate that tri-training can effectively exploit unlabeled data to enhance the learning performance.**

***Index Terms*— Data Mining, Machine Learning, Learning from Unlabeled Data, Semi-supervised Learning, Co-training, Tri-training, Web Page Classification**

I. Introduction

In many practical data mining applications such as web page classification, unlabeled training examples are readily available but labeled ones are fairly expensive to obtain because they require human effort. Therefore, semi-supervised learning that exploits unlabeled examples in addition to labeled ones has become a hot topic.

Many current semi-supervised learning algorithms use a generative model for the classifier and employ Expectation-Maximization (EM) [11] to model the label estimation or parameter estimation process. For example, mixture of Gaussians [26], mixture of experts [17], and naive Bayes [20] have been respectively used as the generative model, while EM is used to combine labeled and unlabeled data for classification. There are also many other algorithms such as using transductive inference for support vector machines to optimize performance on a specific test set [16], constructing a graph on the examples such that the minimum cut on the graph yields an optimal labeling of the unlabeled examples according to certain optimization functions [4], etc.

A prominent achievement in this area is the *co-training* paradigm proposed by Blum and Mitchell [5], which trains two classifiers separately on two different views, i.e. two independent sets of attributes, and uses the predictions of each classifier on unlabeled examples to augment the training set of the other. Such an idea of utilizing the natural redundancy in the attributes has been employed in some other works. For example, Yarowsky [28] performed word sense disambiguation by constructing a sense classifier using the local context of the word and a classifier based on the senses of other occurrences of that word in the same document; Riloff and Jones [23] classified a noun phrase for geographic locations by considering both the noun phrase itself and the linguistic context in which the noun phrase appears; Collins and Singer [8] performed named entity classification using both the spelling of the entity itself and the context in which the entity occurs. It is noteworthy that the co-training paradigm has already been used in many domains such as statistical parsing and noun phrase identification [15] [21] [24] [27].

The standard co-training algorithm [5] requires two *sufficient and redundant views*, that is, the attributes be naturally partitioned into two sets, each of which is sufficient for learning and conditionally independent to the other given the class label. Dasgupta et al. [10] have shown that when the requirement is met, the co-trained classifiers could make fewer generalization errors by maximizing their agreement over the unlabeled data. Unfortunately, such a requirement can hardly be met in most scenarios. Goldman and Zhou [14] proposed an algorithm which does not exploit attribute partition. However, it requires using two different supervised learning algorithms that partition the instance space into a set of equivalence classes, and employing time-consuming cross validation technique to determine how to label the unlabeled examples and how to produce the final hypothesis.

In this paper, a new co-training style algorithm named *tri-training* is proposed. Tri-training does not require sufficient and redundant views, nor does it require the use of different supervised learning algorithms whose hypothesis partitions the instance space into a set of equivalence classes. Therefore it can be easily applied to common data mining scenarios. In contrast to previous algorithms that utilize two classifiers, tri-training uses three classifiers. This setting tackles the problem of determining how to label the unlabeled examples and how to produce the final hypothesis, which contributes much to the efficiency of the algorithm. Moreover, better generalization ability can be achieved through combining these three

Manuscript received xxx xx, 200x; revised xxx xx, 200x. This work was supported by the the National Science Fund for Distinguished Young Scholars of China under the Grant No. 60325207, the Foundation for the Author of National Excellent Doctoral Dissertation of China under the Grant No. 200343, the Jiangsu Science Foundation Key Project under the Grant No. BK2004001, and the National Science Foundation of China under the Grant No. 60496320.

The authors are with the National Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China (e-mail: {zhouzh,lim}@lamda.nju.edu.cn).

classifiers. Experiments on UCI data sets [3] and application to the web page classification task show that tri-training can effectively exploit unlabeled data, and the generalization ability of its final hypothesis is quite good, sometimes even outperforms that of the ensemble of three classifiers being provided with labels of all the unlabeled examples.

The rest of this paper is organized as follows. Section 2 presents the tri-training algorithm. Section 3 reports on the experiments on UCI data sets. Section 4 describes the application to the task of web page classification. Finally, Section 5 concludes and raises several issues for future work.

II. Tri-training

Let L denote the labeled example set with size $|L|$ and U denote the unlabeled example set with size $|U|$. In previous co-training style algorithms, two classifiers are initially trained from L, each of which is then re-trained with the help of unlabeled examples that are labeled by the latest version of the other classifier. In order to determine which example in U should be labeled and which classifier should be biased in prediction, the confidence of the labeling of each classifier must be explicitly measured. Sometimes such a measuring process is quite time-consuming [14].

Assume that besides these two classifiers, i.e. h_1 and h_2, a classifier h_3 is initially trained from L. Then, for any classifier, an unlabeled example can be labeled for it as long as the other two classifiers agree on the labeling of this example, while the confidence of the labeling of the classifiers are not needed to be explicitly measured. For instance, if h_2 and h_3 agree on the labeling of an example x in U, then x can be labeled for h_1. It is obvious that in such a scheme if the prediction of h_2 and h_3 on x is correct, then h_1 will receive a valid new example for further training; otherwise h_1 will get an example with noisy label. However, even in the worse case, the increase in the classification noise rate can be compensated if the amount of newly labeled examples is sufficient, under certain conditions, as shown below.

Inspired by Goldman and Zhou [14], the finding of Angluin and Laird [1] is used in the following analysis. That is, if a sequence σ of m samples is drawn, where the sample size m satisfies Eq. 1:

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

where ϵ is the hypothesis worst-case classification error rate, η (< 0.5) is an upper bound on the classification noise rate, N is the number of hypothesis, and δ is the confidence, then a hypothesis H_i that minimizes disagreement with σ will have the PAC property:

$$\Pr\left[d(H_i, H^*) \geq \epsilon\right] \leq \delta \tag{2}$$

where $d(,)$ is sum over the probability of elements from the symmetric difference between the two hypothesis sets H_i and H^* (the ground-truth). Let $c = 2\mu \ln(\frac{2N}{\delta})$ where μ makes Eq. 1 hold equality, then Eq. 1 becomes Eq. 3:

$$m = \frac{c}{\epsilon^2 (1-2\eta)^2} \tag{3}$$

To simplify the computation, it is helpful to compute the quotient of the constant c divided by the square of the error:

$$u = \frac{c}{\epsilon^2} = m(1-2\eta)^2 \tag{4}$$

In each round of tri-training, the classifiers h_2 and h_3 choose some examples in U to label for h_1. Since the classifiers are refined in the tri-training process, the amount as well as the concrete unlabeled examples chosen to label may be different in different rounds. Let L^t and L^{t-1} denote the set of examples that are labeled for h_1 in the t-th round and the $(t-1)$-th round, respectively. Then the training set for h_1 in the t-th round and $(t-1)$-th round are respectively $L \cup L^t$ and $L \cup L^{t-1}$, whose sample size m^t and m^{t-1} are $|L \cup L^t|$ and $|L \cup L^{t-1}|$, respectively. Note that the unlabeled examples labeled in the $(t-1)$-th round, i.e. L^{t-1}, won't be put into the original labeled example set, i.e. L. Instead, in the t-th round all the examples in L^{t-1} will be regarded as unlabeled and put into U again.

Let η_L denote the classification noise rate of L, that is, the number of examples in L that are mislabeled is $\eta_L|L|$. Let $\check{e}_1^t$ denote the upper bound of the classification error rate of $h_2 \& h_3$ in the t-th round, i.e. the error rate of the hypothesis derived from the combination of h_2 and h_3. Assuming there are z number of examples on which the classification made by h_2 agrees with that made by h_3, and among these examples both h_2 and h_3 make correct classification on z' examples, then $\check{e}_1^t$ can be estimated as $\frac{(z-z')}{z}$. Thus, the number of examples in L^t that are mislabeled is $\check{e}_1^t|L^t|$. Therefore the classification noise rate in the t-th round is:

$$\eta^t = \frac{\eta_L |L| + \check{e}_1^t |L^t|}{|L \cup L^t|} \tag{5}$$

Then, according to Eq. 4, u^t can be computed as:

$$u^t = m^t (1-2\eta^t)^2 = |L \cup L^t| \left(1 - 2\frac{\eta_L |L| + \check{e}_1^t |L^t|}{|L \cup L^t|}\right)^2 \tag{6}$$

Similarly, u^{t-1} can be computed as:

$$\begin{aligned} u^{t-1} &= m^{t-1}(1-2\eta^{t-1})^2 \\ &= |L \cup L^{t-1}| \left(1 - 2\frac{\eta_L |L| + \check{e}_1^{t-1} |L^{t-1}|}{|L \cup L^{t-1}|}\right)^2 \end{aligned} \tag{7}$$

As shown in Eq. 4, since u is in proportion to $1/\epsilon^2$, it can be derived that if $u^t > u^{t-1}$ then $\epsilon^t < \epsilon^{t-1}$, which implies that h_1 can be improved through utilizing L^t in its training. This condition can be expressed as Eq. 8 by comparing Eqs. 6 and 7:

$$\begin{aligned} &|L \cup L^t| \left(1 - 2\frac{\eta_L |L| + \check{e}_1^t |L^t|}{|L \cup L^t|}\right)^2 > \\ &|L \cup L^{t-1}| \left(1 - 2\frac{\eta_L |L| + \check{e}_1^{t-1} |L^{t-1}|}{|L \cup L^{t-1}|}\right)^2 \end{aligned} \tag{8}$$

Considering that η_L can be very small and assuming $0 \leq \check{e}_1^t, \check{e}_1^{t-1} < 0.5$, then the first term on the left hand of Eq. 8 is

bigger than its correspondence on the right hand if $|L^{t-1}| < |L^t|$, while the second term on the left hand is bigger than its correspondence on the right hand if $\check{e}_1^t|L^t| < \check{e}_1^{t-1}|L^{t-1}|$. These restrictions can be summarized into the condition shown in Eq. 9, which is used in tri-training to determine when an unlabeled example could be labeled for a classifier.

$$0 < \frac{\check{e}_1^t}{\check{e}_1^{t-1}} < \frac{|L^{t-1}|}{|L^t|} < 1 \tag{9}$$

Note that when $\check{e}_1^t < \check{e}_1^{t-1}$ and $|L^{t-1}| < |L^t|$, $\check{e}_1^t|L^t|$ may not be less than $\check{e}_1^{t-1}|L^{t-1}|$ due to the fact that $|L^t|$ may be far bigger than $|L^{t-1}|$. When this happens, in some cases L^t could be randomly subsampled such that $\check{e}_1^t|L^t| < \check{e}_1^{t-1}|L^{t-1}|$. Given $\check{e}_1^t$, $\check{e}_1^{t-1}$, and $|L^{t-1}|$, let integer s denote the size of L^t after subsampling, then if Eq. 10 holds, $\check{e}_1^t|L^t| < \check{e}_1^{t-1}|L^{t-1}|$ is obviously satisfied.

$$s = \left\lceil \frac{\check{e}_1^{t-1}|L^{t-1}|}{\check{e}_1^t} - 1 \right\rceil \tag{10}$$

where L^{t-1} should satisfy Eq. 11 such that the size of L^t after subsampling, i.e. s, is still bigger than $|L^{t-1}|$.

$$|L^{t-1}| > \frac{\check{e}_1^t}{\check{e}_1^{t-1} - \check{e}_1^t} \tag{11}$$

The pseudo-code of tri-training is presented in Table I. The function $MeasureError(h_j \& h_k)$ attempts to estimate the classification error rate of the hypothesis derived from the combination of h_j and h_k. Since it is difficult to estimate the classification error on the unlabeled examples, here only the original labeled examples are used, heuristically based on the assumption that the unlabeled examples hold the same distribution as that held by the labeled ones. In detail, the classification error of the hypothesis is approximated through dividing the number of labeled examples on which both h_j and h_k make incorrect classification by the number of labeled examples on which the classification made by h_j is the same as that made by h_k. The function $Subsample(L^t, s)$ randomly removes $|L^t| - s$ number of examples from L^t where s is computed according to Eq. 10.

It is noteworthy that the initial classifiers in tri-training should be diverse because if all the classifiers are identical, then for any of these classifiers, the unlabeled examples labeled by the other two classifiers will be the same as these labeled by the classifier for itself. Thus, tri-training degenerates to *self-training* [19] with a single classifier. In the standard co-training algorithm, the use of sufficient and redundant views enables the classifiers be different. In fact, previous research has shown that even when there is no natural attribute partitions, if there are sufficient redundancy among the attributes then a fairly reasonable attribute partition will enable co-training exhibit advantages [19]. While in the extended co-training algorithm which does not require sufficient and redundant views, the diversity among the classifiers is achieved through using different supervised learning algorithms [14]. Since the tri-training algorithm does not assume sufficient and redundant views and different supervised learning algorithms, the diversity of the classifiers have to be sought from other channels. Indeed, here the diversity is obtained through manipulating the original labeled example set. In detail, the initial classifiers are trained from data sets generated via bootstrap sampling [13] from the original labeled example set. These classifiers are then refined in the tri-training process, and the final hypothesis is produced via *majority voting*. The generation of the initial classifiers looks like training an ensemble from the labeled example set with a popular ensemble learning [12] algorithm, that is, Bagging [6].

Tri-training can be regarded as a new extension to the co-training algorithms [5] [14]. As mentioned before, Blum and Mitchell's algorithm requires the instance space be described by two sufficient and redundant views, which can hardly be satisfied in common data mining scenarios. Since tri-training does not rely on different views, its applicability is broader. Goldman and Zhou's algorithm does not rely on different views either. However, their algorithm requires two different supervised learning algorithms that partition the instance space into a set of equivalence classes. Moreover, their algorithm frequently uses 10-fold cross validation on the original labeled example set to determine how to label the unlabeled examples and how to produce the final hypothesis. If the original labeled

TABLE I

PSEUDO-CODE DESCRIBING THE TRI-TRAINING ALGORITHM

tri-training(L, U, $Learn$)

Input: L: Original labeled example set
U: Unlabeled example set
$Learn$: Learning algorithm

for $i \in \{1..3\}$ **do**
  $S_i \leftarrow BootstrapSample(L)$
  $h_i \leftarrow Learn(S_i)$
  $e_i^{'} \leftarrow .5$; $l_i^{'} \leftarrow 0$
end of for

repeat until none of h_i $(i \in \{1..3\})$ changes
  for $i \in \{1..3\}$ **do**
    $L_i \leftarrow \emptyset$; $update_i \leftarrow FALSE$
    $e_i \leftarrow MeasureError(h_j \& h_k)$ $(j, k \neq i)$
    if $(e_i < e_i^{'})$ % otherwise Eq. 9 is violated
    then for every $x \in U$ **do**
      if $h_j(x) = h_k(x)$ $(j, k \neq i)$
      then $L_i \leftarrow L_i \cup \{(x, h_j(x))\}$
    end of for
    if $(l_i^{'} = 0)$ % h_i has not been updated before
    then $l_i^{'} \leftarrow \left\lfloor \frac{e_i}{e_i^{'} - e_i} + 1 \right\rfloor$ % refer Eq. 11
    if $(l_i^{'} < |L_i|)$ % otherwise Eq. 9 is violated
    then if $(e_i|L_i| < e_i^{'} l_i^{'})$ % otherwise Eq. 9 is violated
      then $update_i \leftarrow TRUE$
      else if $l_i^{'} > \frac{e_i}{e_i^{'} - e_i}$ % refer Eq. 11
        then $L_i \leftarrow Subsample(L_i, \left\lceil \frac{e_i^{'} l_i^{'}}{e_i} - 1 \right\rceil)$
        % refer Eq. 10
        $update_i \leftarrow TRUE$
  end of for
  for $i \in \{1..3\}$ **do**
    if $update_i = TRUE$
    then $h_i \leftarrow Learn(L \cup L_i)$; $e_i^{'} \leftarrow e_i$; $l_i^{'} \leftarrow |L_i|$
  end of for
end of repeat

Output: $h(x) \leftarrow \arg\max_{y \in label} \sum_{i:\, h_i(x) = y} 1$

example set is rather small, cross validation will exhibit high variance and is not helpful for model selection. Also, the frequently used cross validation makes the learning process time-consuming. Since tri-training does not put any constraint on the supervised learning algorithm nor does it employ time-consuming cross validation process, both its applicability and efficiency are better.

III. Experiments on UCI Data Sets

Twelve UCI data sets [3] are used in the experiments. Information on these data sets are tabulated in Table II, where *pos/neg* presents the percentage of the number of positive examples against that of negative examples. Note that these data sets do not hold sufficient and redundant views.

TABLE II
Experimental data sets

data set	attribute	size	class	pos/neg
australian	14	690	2	55.5%/44.5%
bupa	6	345	2	42.0%/58.0%
colic	22	368	2	63.0%/37.0%
diabetes	8	768	2	65.1%/34.9%
german	20	1,000	2	70.0%/30.0%
hypothyroid	25	3,163	2	4.8%/95.2%
ionosphere	34	351	2	35.9%/64.1%
kr-vs-kp	36	3,196	2	52.2%/47.8%
sick	29	3,772	2	6.1%/93.9%
tic-tac-toe	9	958	2	65.3%/34.7%
vote	16	435	2	61.4%/38.6%
wdbc	30	569	2	37.3%/62.7%

For each data set, about 25% data are kept as test examples while the rest are used as the pool of training examples, i.e. $L \cup U$. In each pool, L and U are partitioned under different *unlabel rates* including 80%, 60%, 40%, and 20%. For instance, assuming a pool contains 1,000 examples, when the unlabel rate is 80%, 200 examples are put into L with their labels while the remaining 800 examples are put into U without their labels. Here the pos/neg ratio of L, U, and the test set are similar to that of the original data set. Note that on some big data sets, such as *hypothyroid*, *kr-vs-kp*, and *sick*, the number of labeled examples might be sufficient to train a good classifier even under 80% unlabel rate. However, even in these cases, semi-supervised learning algorithms such as tri-training can still be helpful, which will be shown in the experiments reported in this section.

J4.8 decision trees [29], BP neural networks, and Naive Bayes classifiers are used in the experiments. Under each unlabel rate, three independent runs with different random partition of L and U are performed. The averaged results are summarized in Tables III to VI, which present the classification error rates of the hypothesis at round 0, i.e. the combination of the three initial classifiers trained from L, the final hypothesis generated by tri-training, and the improvement of the latter over the former.

The performance of tri-training is compared with three semi-supervised learning algorithms, i.e. *co-training*, *self-training1*, and *self-training2*. The co-training algorithm is almost the same as the standard one [5] except that since the experimental data sets are without natural sufficient and redundant views, the original attribute sets are randomly partitioned into two subsets with similar sizes and then each subset is regarded as a view. The self-training1 algorithm uses the same three initial classifiers as these used by tri-training. In each round, instead of using the other two classifiers to label examples, each classifier labels unlabeled examples for itself while in predicting new examples, all the three classifiers are used. In other words, three single classifiers refined by self-training [19] is combined via majority voting. In contrast to tri-training, this algorithm does not utilize any co-training process while the voting scheme is used to improve generalization. The self-training2 algorithm also uses the same three initial classifiers as these used by tri-training. In each round, the unlabeled examples are labeled via majority voting the classifiers, and each classifier is refined by the same copy of the newly labeled data. Finally, the newly labeled examples and the original labeled examples are used together to train a single classifier which is used in prediction. It is worth noting that although self-training2 uses the same initial classifiers as these used in tri-training and self-training1, its initial hypotheses are different because it uses a single classifier instead of an ensemble in prediction. For fair comparison, the termination criteria used by these semi-supervised learning algorithms are similar to that used by tri-training. In Tables III to VI, the biggest improvements achieved by the semi-supervised learning algorithms have been boldfaced. Note that some values in the tables may look inconsistent due to truncation. For example, the initial and final performance of tri-training with J4.8 decision tree on *hypothyroid* appear identical in Table V but the improvement is not zero.

Tables III to VI show that tri-training can effectively improve the hypotheses with all the classifiers under all the unlabel rates. In fact, if the improvements are averaged across all the data sets, classifiers and unlabel rates, it can be found that the average improvement of tri-training is about 11.9%. It is impressive that with all the classifiers and under all the unlabel rates, tri-training has achieved the biggest average improvement. Moreover, Tables III to VI also show that if the algorithms are compared through counting the number of *winning data sets*, i.e. the number of data sets on which an algorithm has achieved the biggest improvement among the compared algorithms, tri-training is almost always the winner. In detail, under 80% unlabel rate, when J4.8 decision trees are used, tri-training has 11 wining data sets while the other algorithms have at most one winning data set; when Naive Bayes classifiers are used, tri-training and self-training1 have 7 winning data sets, respectively, while the remaining algorithms do not have winning data sets. Under 60% unlabel rate, when J4.8 decision trees and BP neural networks are used, tri-training has 8 and 9 winning data sets, respectively, while the other algorithms have at most 2 and 3 winning data sets, respectively; when Naive Bayes classifiers are used, tri-training has 10 winning data sets, while the other algorithms have at most 5 winning data sets. Under 40% unlabel rate, tri-training has 7, 6, and 9 winning data sets when J4.8 decision trees, BP neural networks, and Naive Bayes classifiers are used, respectively, while the other algorithms have at most 5, 4, and 6 winning data sets, respectively. Under 20% unlabel

TABLE III

THE CLASSIFICATION ERROR RATES OF THE INITIAL AND FINAL HYPOTHESES AND THE CORRESPONDING IMPROVEMENTS OF TRI-TRAINING, CO-TRAINING, SELF-TRAINING1, AND SELF-TRAINING2, UNDER 80% UNLABEL RATE

Data set	J4.8 decision tree											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.222	.193	**13.0%**	.263	.253	3.7%	.222	.202	8.7%	.198	.186	5.9%
bupa	.399	.368	**7.8%**	.448	.433	3.4%	.399	.380	4.9%	.394	.448	-13.7%
colic	.181	.163	**10.0%**	.206	.188	8.8%	.181	.174	4.0%	.188	.181	3.8%
diabetes	.316	.288	**8.8%**	.280	.266	5.0%	.316	.300	4.9%	.313	.299	4.4%
german	.351	.324	**7.6%**	.349	.337	3.4%	.351	.337	3.8%	.341	.341	0.0%
hypothyroid	.012	.011	**10.3%**	.018	.016	9.3%	.012	.012	0.0%	.008	.009	-10.0%
ionosphere	.155	.121	**22.0%**	.167	.144	13.6%	.155	.144	7.3%	.129	.137	-5.9%
kr-vs-kp	.035	.025	29.8%	.125	.123	1.7%	.035	.022	**36.9%**	.022	.023	-5.8%
sick	.024	.021	**11.9%**	.055	.051	7.1%	.024	.023	4.5%	.024	.028	-14.3%
tic-tac-toe	.292	.258	**11.4%**	.299	.295	1.4%	.292	.269	7.6%	.291	.281	3.3%
vote	.076	.055	**28.0%**	.083	.074	11.1%	.076	.070	8.0%	.052	.052	0.0%
wdbc	.094	.075	**20.0%**	.096	.082	14.6%	.094	.092	2.5%	.089	.094	-5.3%
ave.	.180	.159	**15.1%**	.199	.189	6.9%	.180	.169	7.8%	.171	.173	-3.1%

Data set	BP neural networks											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.200	.181	**9.6%**	.143	.135	5.4%	.200	.189	5.8%	.149	.151	-1.3%
bupa	.337	.295	**12.5%**	.399	.379	4.9%	.337	.326	3.4%	.348	.348	0.0%
colic	.254	.232	8.6%	.253	.239	5.7%	.254	.261	-2.9%	.283	.243	**14.1**%
diabetes	.286	.257	**10.3%**	.271	.262	3.2%	.286	.262	8.5%	.264	.278	-5.3%
german	.301	.287	4.9%	.283	.275	2.8%	.301	.284	**5.8%**	.315	.311	1.3%
hypothyroid	.029	.021	25.0%	.033	.030	10.0%	.029	.024	16.2%	.052	.036	**30.3%**
ionosphere	.201	.178	11.3%	.160	.141	**11.9%**	.201	.178	11.3%	.167	.175	-4.5%
kr-vs-kp	.034	.027	22.2%	.101	.098	2.9%	.034	.025	**27.2%**	.030	.030	-1.4%
sick	.036	.034	4.0%	.051	.046	**9.2%**	.036	.034	4.0%	.045	.045	-0.8%
tic-tac-toe	.071	.036	49.0%	.269	.256	4.7%	.071	.026	**62.7%**	.015	.015	0.0%
vote	.061	.050	17.5%	.076	.058	24.0%	.061	.043	**30.0%**	.043	.043	0.0%
wdbc	.042	.033	22.2%	.037	.028	**25.0%**	.042	.038	11.1%	.032	.030	7.1%
ave.	.154	.136	**16.4%**	.173	.162	9.1%	.154	.141	15.3%	.145	.142	3.3%

Data set	Naive Bayes											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.243	.224	**7.9%**	.238	.234	1.6%	.243	.224	**7.9%**	.236	.236	0.0%
bupa	.481	.442	8.1%	.459	.448	2.5%	.481	.438	**8.9%**	.490	.475	3.1%
colic	.217	.207	5.0%	.207	.203	1.8%	.217	.203	**6.7%**	.210	.221	-5.2%
diabetes	.267	.257	**3.9%**	.250	.245	2.1%	.267	.264	1.3%	.246	.241	2.1%
german	.285	.276	3.3%	.257	.253	1.6%	.285	.272	**4.7%**	.262	.267	-2.0%
hypothyroid	.024	.021	**8.9%**	.025	.024	3.4%	.024	.021	**8.9%**	.022	.022	0.0%
ionosphere	.155	.129	**17.1%**	.183	.175	4.2%	.155	.136	12.2%	.178	.186	-4.3%
kr-vs-kp	.142	.128	**10.0%**	.144	.143	0.6%	.142	.128	9.4%	.139	.138	0.9%
sick	.089	.084	5.6%	.043	.042	1.7%	.089	.083	**6.3%**	.079	.078	0.9%
tic-tac-toe	.343	.324	**5.7%**	.286	.280	2.0%	.343	.328	4.5%	.293	.294	-0.5%
vote	.113	.104	**8.1%**	.098	.098	0.0%	.113	.110	2.7%	.098	.101	-3.1%
wdbc	.054	.047	13.0%	.071	.066	6.7%	.054	.045	**17.4%**	.068	.073	-6.9%
ave.	.201	.187	**8.0%**	.188	.184	2.4%	.201	.188	7.6%	.193	.194	-1.3%

rate, when J4.8 decision trees and BP neural networks are used, tri-training has 8 and 9 winning data sets, respectively, while the remaining algorithms have at most 3 and 2 winning data sets, respectively; when Naive Bayes classifiers are used, tri-training has 7 winning data sets while the other algorithms have at most 6 winning data sets. Only when BP neural networks are used under 80% unlabel rate, tri-training has fewer winning data sets than self-training1.

The error rates of the compared algorithms are depicted in Figs. 1 to 3. Besides the semi-supervised learning algorithms, on each data set three single classifiers are trained from only the labeled training examples, i.e. L. The average error rate of the single classifiers is shown as a horizontal line in each figure, which is denoted by *single*. Moreover, three ensembles each comprising three classifiers are trained by Bagging from the pool of training examples, i.e. $(L \cup U)$ while labels of all the examples are provided. The average error rate of the ensembles is also shown as a horizontal line in each figure, which is denoted by *ens-all*. Note that in Figs. 1 to 3 the error rates have been averaged across all the experimental data sets, and since the semi-supervised learning algorithms may terminate in different rounds, the error rates at termination are used as the error rates of the rounds after termination.

Figs. 1 to 3 reveal that on all the subfigures, the final hypotheses generated by tri-training are better than the initial hypotheses, which confirms that tri-training can effectively exploit unlabeled examples to enhance the learning performance. When J4.8 decision trees are used, the hypotheses generated by tri-training are apparently better than these generated by the other semi-supervised learning algorithms in the same rounds, except that under 40% unlabel rate the hypotheses generated by tri-training and self-training1 are comparable.

TABLE IV

THE CLASSIFICATION ERROR RATES OF THE INITIAL AND FINAL HYPOTHESES AND THE CORRESPONDING IMPROVEMENTS OF TRI-TRAINING, CO-TRAINING, SELF-TRAINING1, AND SELF-TRAINING2, UNDER 60% UNLABEL RATE

Data set	J4.8 decision tree											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.171	.162	**5.6%**	.223	.213	4.3%	.171	.166	3.4%	.168	.166	1.1%
bupa	.376	.341	**9.3%**	.402	.398	1.0%	.376	.376	0.0%	.371	.390	-5.2%
colic	.185	.163	**11.8%**	.210	.199	5.2%	.185	.167	9.8%	.210	.199	5.2%
diabetes	.286	.252	**12.1%**	.286	.267	6.7%	.286	.288	-0.6%	.267	.264	1.3%
german	.337	.316	**6.3%**	.318	.313	1.7%	.337	.333	1.2%	.306	.325	-6.1%
hypothyroid	.014	.011	21.2%	.034	.033	3.7%	.014	.010	**27.3%**	.014	.012	14.7%
ionosphere	.102	.087	14.8%	.134	.103	**22.9%**	.102	.091	11.1%	.121	.129	-6.3%
kr-vs-kp	.021	.018	**17.6%**	.078	.077	1.6%	.021	.020	7.8%	.013	.014	-6.5%
sick	.020	.016	**21.1%**	.045	.042	6.3%	.020	.018	10.5%	.021	.021	0.0%
tic-tac-toe	.176	.163	**7.9%**	.255	.252	1.1%	.176	.188	-6.3%	.225	.221	1.9%
vote	.058	.049	15.8%	.073	.055	**25.0%**	.058	.049	15.8%	.049	.040	18.8%
wdbc	.089	.077	13.2%	.089	.075	15.8%	.089	.073	**18.4%**	.094	.089	5.0%
ave.	.153	.138	**13.1%**	.179	.169	7.9%	.153	.148	8.2%	.155	.156	2.0%

Data set	BP neural networks											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.152	.135	**11.4%**	.155	.143	7.5%	.152	.152	0.0%	.155	.151	2.5%
bupa	.353	.302	**14.4%**	.386	.363	6.0%	.353	.306	13.3%	.340	.344	-1.1%
colic	.203	.192	5.4%	.242	.217	**10.4%**	.203	.192	5.4%	.225	.232	-3.2%
diabetes	.245	.233	**5.0%**	.250	.243	2.8%	.245	.238	2.8%	.243	.257	-5.7%
german	.296	.281	5.0%	.308	.288	**6.5%**	.296	.292	1.4%	.282	.297	-5.2%
hypothyroid	.027	.024	**10.9%**	.025	.024	5.1%	.027	.027	1.6%	.024	.027	-12.5%
ionosphere	.182	.159	12.5%	.148	.129	**12.8%**	.182	.174	4.2%	.155	.163	-4.9%
kr-vs-kp	.020	.015	**25.0%**	.111	.108	3.0%	.020	.016	18.8%	.015	.020	-37.1%
sick	.040	.033	**18.6%**	.044	.040	8.1%	.040	.033	17.7%	.034	.043	-28.1%
tic-tac-toe	.025	.021	**16.7%**	.258	.244	5.4%	.025	.025	0.0%	.015	.014	9.1%
vote	.046	.037	**20.0%**	.055	.046	16.7%	.046	.037	**20.0%**	.055	.070	-27.8%
wdbc	.040	.031	**23.5%**	.052	.045	13.6%	.040	.035	11.8%	.052	.052	0.0%
ave.	.136	.122	**14.0%**	.170	.158	8.2%	.136	.127	8.1%	.133	.139	-9.5%

Data set	Naive Bayes											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.262	.256	**2.2%**	.197	.193	2.0%	.262	.256	**2.2%**	.183	.187	-2.1%
bupa	.453	.434	4.3%	.444	.425	4.3%	.453	.430	**5.1%**	.460	.468	-1.7%
colic	.207	.196	**5.3%**	.206	.199	3.5%	.207	.196	**5.3%**	.221	.210	4.9%
diabetes	.257	.241	**6.1%**	.277	.274	1.2%	.257	.243	5.4%	.283	.288	-1.8%
german	.279	.269	**3.3%**	.253	.249	1.6%	.279	.276	1.0%	.257	.257	0.0%
hypothyroid	.027	.025	**6.3%**	.022	.021	3.8%	.027	.025	**6.3%**	.020	.019	4.3%
ionosphere	.235	.201	**14.5%**	.220	.220	0.0%	.235	.205	12.9%	.225	.217	3.4%
kr-vs-kp	.129	.121	**6.5%**	.147	.146	0.6%	.129	.122	5.8%	.143	.143	0.3%
sick	.086	.084	**2.5%**	.043	.043	0.8%	.086	.085	1.2%	.086	.087	-1.2%
tic-tac-toe	.311	.299	**4.0%**	.270	.267	1.0%	.311	.303	2.7%	.287	.287	0.0%
vote	.110	.104	**5.6%**	.123	.120	2.5%	.110	.104	**5.6%**	.119	.116	2.6%
wdbc	.040	.038	5.9%	.073	.068	**6.5%**	.040	.040	0.0%	.072	.070	3.2%
ave.	.200	.189	**5.5%**	.190	.185	2.3%	.200	.190	4.4%	.196	.196	1.0%

When BP neural networks are used, the hypotheses generated by tri-training are apparently better than these generated by co-training under all the unlabel rates, apparently better than these generated by self-training1 under 20% unlabel rate, and apparently better than these generated by self-training2 on all but 80% unlabel rate. When Naive Bayes classifiers are used, the hypotheses generated by tri-training are comparable to these generated by co-training and self-training1, while apparently better than these generated by self-training2 under all the unlabel rates.

For further studying the performance of the compared semi-supervised learning algorithms, the number of test examples misclassified by the algorithms are depicted in Figs. 4 to 6, which belongs to the one of the three runs of each algorithm that has the median performance. Note that here only a small number of figures are depicted since it may be too tedious to present all the figures (12 data sets $\times$ 3 classifiers $\times$ 4 unlabel rates = 144 figures). The figures presented are chosen according to the following two criteria. First, for each classifier and under each unlabel rate, the data sets where tri-training achieves median improvements can be chosen. For instance, according to Table VI, when BP neural networks are used under 20% unlabel rate, tri-training achieves its 6th and 7th biggest improvements on *sick* and *wdbc*, respectively, among all the twelve data sets. Therefore *sick* and *wdbc* can be chosen. Second, attempts are made to choose as more diverse data sets as possible. For instance, since *sick* has already been chosen when J4.8 decision trees are used under 80% unlabel rate, *wdbc* instead of *sick* is chosen for BP neural networks under 20% unlabel rate.

Figs. 4 to 6 reveal that on all the subfigures, the final hypotheses generated by tri-training are better than the initial

TABLE V

THE CLASSIFICATION ERROR RATES OF THE INITIAL AND FINAL HYPOTHESES AND THE CORRESPONDING IMPROVEMENTS OF TRI-TRAINING, CO-TRAINING, SELF-TRAINING1, AND SELF-TRAINING2, UNDER 40% UNLABEL RATE

Data set	J4.8 decision tree											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.162	.148	**8.3%**	.190	.180	5.1%	.162	.154	4.8%	.164	.160	2.4%
bupa	.391	.353	**9.9%**	.360	.348	3.2%	.391	.380	3.0%	.340	.375	-10.2%
colic	.196	.185	**5.6%**	.203	.196	3.6%	.196	.188	3.7%	.195	.188	3.7%
diabetes	.293	.273	7.1%	.279	.267	4.3%	.293	.267	**8.9%**	.278	.269	3.1%
german	.316	.293	**7.2%**	.318	.313	1.7%	.316	.301	4.6%	.290	.297	-2.3%
hypothyroid	.011	.011	**7.4%**	.015	.014	5.7%	.011	.011	**7.4%**	.010	.010	0.0%
ionosphere	.102	.087	14.8%	.126	.118	6.1%	.102	.080	**22.2%**	.102	.125	-22.2%
kr-vs-kp	.013	.010	20.0%	.073	.073	0.6%	.013	.008	**33.3%**	.016	.016	0.0%
sick	.018	.012	**30.0%**	.043	.041	4.2%	.018	.017	2.0%	.017	.016	4.3%
tic-tac-toe	.150	.133	**11.1%**	.222	.219	1.3%	.150	.143	4.6%	.212	.217	-2.6%
vote	.061	.049	20.0%	.083	.077	7.4%	.061	.040	**35.0%**	.067	.067	0.0%
wdbc	.080	.066	17.6%	.063	.047	**25.9%**	.080	.073	8.8%	.080	.080	0.0%
ave.	.149	.135	**13.3%**	.165	.158	5.8%	.149	.139	11.5%	.148	.152	-2.0%

Data set	BP neural networks											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.148	.137	**7.8%**	.145	.135	6.7%	.148	.150	-1.3%	.134	.155	-15.9%
bupa	.318	.283	**11.0%**	.422	.406	3.7%	.318	.295	7.4%	.348	.383	-10.0%
colic	.192	.178	7.5%	.246	.239	2.9%	.192	.170	**11.3%**	.240	.225	6.1%
diabetes	.248	.238	**4.2%**	.288	.283	1.8%	.248	.245	1.4%	.249	.247	0.7%
german	.303	.285	5.7%	.287	.259	**9.8%**	.303	.301	0.4%	.307	.295	3.9%
hypothyroid	.023	.019	**16.4%**	.031	.029	6.8%	.023	.021	7.3%	.027	.023	14.1%
ionosphere	.140	.117	16.2%	.084	.065	**22.7%**	.140	.125	10.8%	.091	.095	-4.2%
kr-vs-kp	.016	.013	**21.1%**	.077	.069	10.8%	.016	.014	13.2%	.012	.011	6.9%
sick	.030	.028	5.9%	.043	.039	**8.4%**	.030	.031	-4.7%	.033	.054	-62.8%
tic-tac-toe	.022	.018	**18.7%**	.241	.224	6.9%	.022	.022	0.0%	.026	.022	15.8%
vote	.055	.052	5.6%	.067	.052	**22.7%**	.055	.052	5.6%	.049	.052	-6.2%
wdbc	.040	.031	23.5%	.033	.028	14.3%	.040	.028	**29.4%**	.031	.033	-7.7%
ave.	.128	.117	**12.0%**	.164	.152	9.8%	.128	.121	6.7%	.129	.133	-4.9%

Data set	Naive Bayes											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.254	.254	0.0%	.246	.242	1.6%	.254	.249	**2.3%**	.236	.236	0.0%
bupa	.450	.419	6.9%	.406	.375	7.6%	.450	.411	**8.6%**	.441	.464	-5.3%
colic	.210	.196	**6.9%**	.203	.203	0.0%	.210	.203	3.4%	.243	.243	0.0%
diabetes	.266	.247	**7.2%**	.243	.238	2.1%	.266	.247	**7.2%**	.248	.243	2.1%
german	.240	.231	**3.9%**	.277	.276	0.5%	.240	.233	2.8%	.269	.269	0.0%
hypothyroid	.021	.019	**11.8%**	.025	.024	3.4%	.021	.019	**11.8%**	.020	.021	-4.1%
ionosphere	.174	.163	**6.5%**	.194	.190	2.0%	.174	.167	4.3%	.194	.194	0.0%
kr-vs-kp	.146	.139	**5.1%**	.137	.136	0.6%	.146	.139	4.9%	.135	.135	-0.3%
sick	.082	.079	**3.9%**	.048	.048	0.0%	.082	.079	3.4%	.076	.076	0.0%
tic-tac-toe	.275	.271	1.5%	.277	.276	0.5%	.275	.269	**2.0%**	.309	.309	0.0%
vote	.089	.080	**10.3%**	.098	.098	0.0%	.089	.080	**10.3%**	.098	.098	0.0%
wdbc	.068	.061	**10.3%**	.061	.061	0.0%	.068	.063	6.9%	.061	.061	0.0%
ave.	.190	.180	**6.2%**	.185	.181	1.5%	.190	.180	5.7%	.194	.196	-0.6%

hypotheses. Comparing with co-training, the final hypotheses of tri-training are almost always better except on Fig. 5 (a) where the final hypothesis of co-training is slightly better. Comparing with self-training1, the final hypotheses of tri-training are better on most subfigures except on Fig. 4 (a), Fig. 6 (b), (c), and (d) where the final hypotheses of tri-training and self-training1 are comparable. Comparing with self-training2, the final hypotheses of tri-training are better on most subfigures except on Fig. 4 (d) where the final hypotheses of tri-training and self-training2 are comparable, and on Fig. 5 (a) and Fig. 6 (d) where the final hypotheses of self-training2 are better.

Figs. 4 to 6 also show that sometimes the performance of ens-all is worse than that of single, especially when Naive Bayes classifiers are used. This is not strange because previous research on ensemble learning has disclosed that Bagging does not always improve the performance and especially it does not work well with stable learners such as Naive Bayes classifiers [6]. However, Figs. 4 to 5 reveal that when ens-all is effective and although it has utilized more resource, i.e. being provided with labels of all the examples in L and U, sometimes the final hypotheses generated by tri-training can outperform that of ens-all, such as on Fig. 4 (b) and Fig. 5 (b) and (c). The above observations confirm that the tri-training process is effective in exploiting unlabeled examples.

IV. APPLICATION TO WEB PAGE CLASSIFICATION

The *web page classification* data set [1] consists of 1,051 web pages collected from web sites of Computer Science departments of four universities: Cornell University, University of

[1]This data set is available at http://www.cs.cmu.edu/afs/cs/project/theo-11/www/wwkb/

TABLE VI

THE CLASSIFICATION ERROR RATES OF THE INITIAL AND FINAL HYPOTHESES AND THE CORRESPONDING IMPROVEMENTS OF TRI-TRAINING, CO-TRAINING, SELF-TRAINING1, AND SELF-TRAINING2, UNDER 20% UNLABEL RATE

Data set	J4.8 decision tree											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.162	.148	**8.3%**	.217	.207	4.5%	.162	.154	4.8%	.124	.126	-1.6%
bupa	.376	.310	**17.5%**	.367	.352	4.2%	.376	.353	6.2%	.336	.332	1.1%
colic	.185	.170	7.8%	.192	.181	5.7%	.185	.178	3.9%	.207	.185	**10.5%**
diabetes	.297	.267	**9.9%**	.294	.290	1.2%	.297	.267	**9.9%**	.286	.288	-0.6%
german	.317	.296	**6.7%**	.327	.320	2.0%	.317	.305	3.8%	.314	.305	3.0%
hypothyroid	.011	.010	**14.8%**	.016	.015	5.3%	.011	.012	-7.4%	.010	.010	0.0%
ionosphere	.129	.102	**20.6%**	.122	.103	15.6%	.129	.117	8.8%	.129	.137	-5.9%
kr-vs-kp	.013	.010	22.6%	.095	.094	0.9%	.013	.009	**29.0%**	.009	.009	-4.8%
sick	.015	.013	**16.3%**	.041	.040	2.6%	.015	.013	14.0%	.012	.015	-23.5%
tic-tac-toe	.143	.125	**12.6%**	.252	.249	1.1%	.143	.158	-10.7%	.165	.166	-0.8%
vote	.052	.040	23.5%	.073	.064	12.5%	.052	.037	**29.4%**	.061	.067	-10.0%
wdbc	.066	.049	25.0%	.058	.042	**28.0%**	.066	.061	7.1%	.061	.054	11.5%
ave.	.147	.128	**15.5%**	.171	.163	7.0%	.147	.139	8.2%	.143	.141	-1.8%

Data set	BP neural networks											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.160	.135	**15.7%**	.134	.124	7.2%	.160	.137	14.5%	.137	.145	-5.6%
bupa	.345	.318	7.8%	.356	.348	2.2%	.345	.337	2.3%	.356	.313	**12.0%**
colic	.199	.181	**9.1%**	.185	.178	3.9%	.199	.217	-9.1%	.200	.185	7.3%
diabetes	.238	.220	**7.3%**	.256	.252	1.4%	.238	.236	0.7%	.265	.286	-7.8%
german	.288	.252	**12.5%**	.270	.257	4.9%	.288	.256	11.1%	.287	.299	-4.2%
hypothyroid	.020	.015	**25.0%**	.024	.022	7.0%	.020	.020	0.0%	.026	.024	6.7%
ionosphere	.117	.087	**25.8%**	.064	.049	23.5%	.117	.091	22.6%	.095	.095	0.0%
kr-vs-kp	.013	.009	26.7%	.072	.069	4.6%	.013	.009	30.0%	.010	.006	**42.3%**
sick	.034	.029	**16.5%**	.051	.049	4.1%	.034	.038	-10.3%	.038	.038	-0.9%
tic-tac-toe	.024	.019	**17.6%**	.209	.196	6.0%	.024	.022	5.9%	.018	.025	-38.5%
vote	.037	.028	25.0%	.064	.055	14.3%	.037	.024	**33.3%**	.055	.052	5.6%
wdbc	.045	.038	**15.8%**	.042	.037	11.1%	.045	.052	-15.8%	.051	.049	4.5%
ave.	.127	.111	**17.1%**	.144	.136	7.5%	.127	.120	7.1%	.128	.126	1.8%

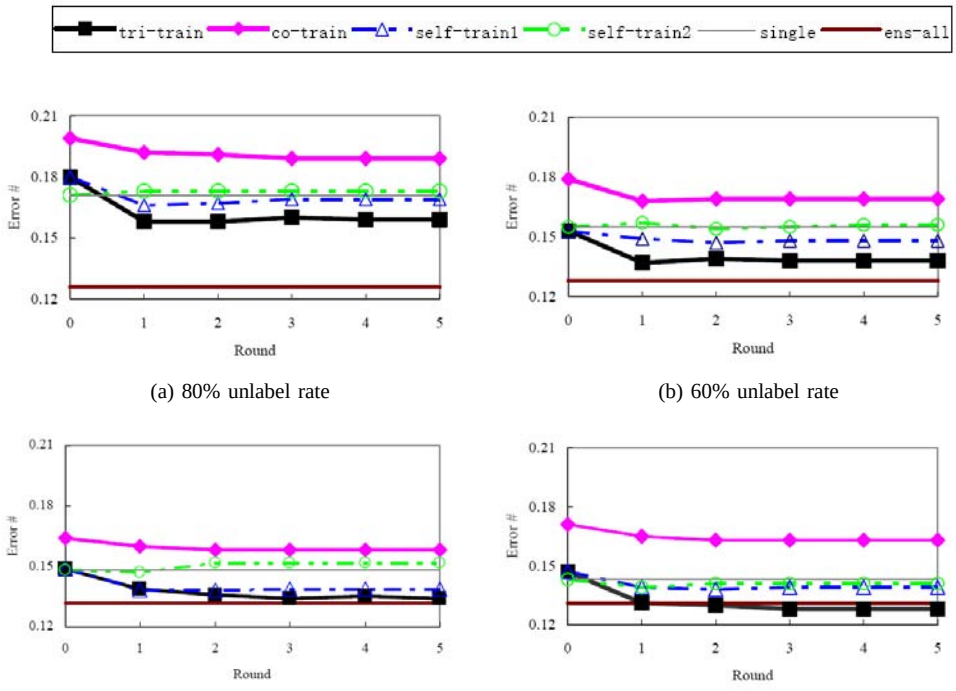
Data set	Naive Bayes											
	tri-training			co-training			self-training1			self-training2		
	initial	final	improv	initial	final	improv	initial	final	improv	initial	final	improv
australian	.218	.202	**7.1%**	.247	.245	0.8%	.218	.204	6.2%	.244	.242	0.8%
bupa	.473	.442	**6.6%**	.410	.394	3.8%	.473	.457	3.3%	.448	.475	-6.0%
colic	.250	.250	0.0%	.199	.192	**3.6%**	.250	.250	0.0%	.232	.232	0.0%
diabetes	.257	.245	4.7%	.251	.247	1.4%	.257	.243	**5.4%**	.261	.259	0.7%
german	.260	.247	5.1%	.268	.265	1.0%	.260	.245	**5.6%**	.242	.243	-0.6%
hypothyroid	.024	.021	**12.3%**	.021	.021	1.9%	.024	.021	**12.3%**	.021	.021	2.0%
ionosphere	.152	.136	**10.0%**	.156	.152	2.4%	.152	.136	**10.0%**	.171	.175	-2.2%
kr-vs-kp	.134	.127	5.3%	.132	.131	0.6%	.134	.127	**5.6%**	.125	.126	-0.7%
sick	.067	.064	**4.8%**	.046	.045	1.6%	.067	.064	3.7%	.064	.065	-1.1%
tic-tac-toe	.290	.281	3.3%	.281	.278	1.0%	.290	.279	**3.8%**	.309	.309	0.0%
vote	.095	.092	**3.2%**	.101	.098	3.0%	.095	.098	-3.2%	.095	.095	0.0%
wdbc	.056	.047	**16.7%**	.077	.077	0.0%	.056	.049	12.5%	.077	.077	0.0%
ave.	.190	.179	**6.6%**	.182	.179	1.8%	.190	.181	5.4%	.191	.193	-0.6%

Texas, University of Washington, and University of Wisconsin. These pages have been labeled into a number of categories, among which the category *course home page* is regarded as the target. That is, course home pages (22%) are the positive examples and all other pages are negative examples.

Here the experimental configuration is the same as that used in [5]. In each experiment, 263(25%) of the 1,051 web pages are first selected at random as test examples. The remaining data is used to generate a labeled training example set L containing 3 positive and 9 negative examples drawn at random. The remaining examples that are not drawn for L are used as the unlabeled pool U. Five runs with different training/test splits are performed, and the average result and standard deviation are recorded.

Note that this data set is with two sufficient and redundant views since a web page can be classified based on the words occurring on that page as well as these occurring in hyperlinks that point to that page [5]. Therefore, Blum and Mitchell [5] trained a page-based classifier and a hyperlink-based classifier for co-training. In addition, they defined a combined classifier to merge the outputs of these two classifiers.

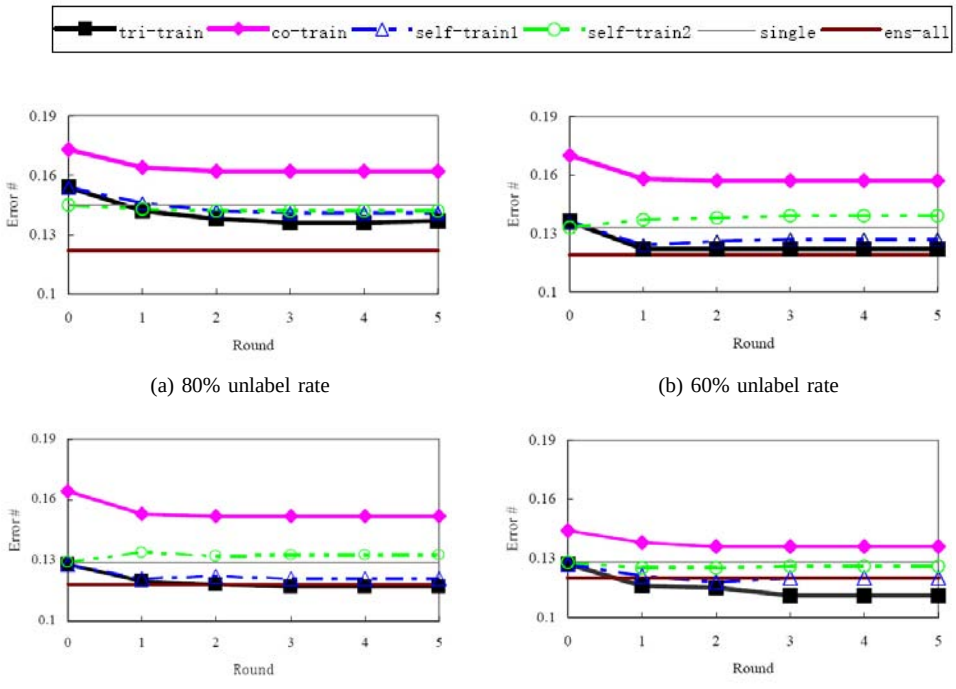
Here the algorithm described in [5] is re-implemented and the parameters are set to the same values as in [5]. J4.8 decision trees, BP neural networks, and Naive Bayes classifiers are used as alternatives to train the classifiers. The average predictive error rates and corresponding standard deviations are shown in Table VII, where *initial* denotes the performance achieved before semi-supervised learning, i.e. the performance obtained using only the labeled training examples, *final* denotes the performance achieved after semi-supervised learning, *improv* denotes the corresponding improvements, and *Hypothesis* denotes the performance of the hypotheses, i.e. the learned



(a) 80% unlabel rate (b) 60% unlabel rate

(c) 40% unlabel rate (d) 20% unlabel rate

Fig. 1. Error rates averaged across all the data sets when J4.8 decision trees are used

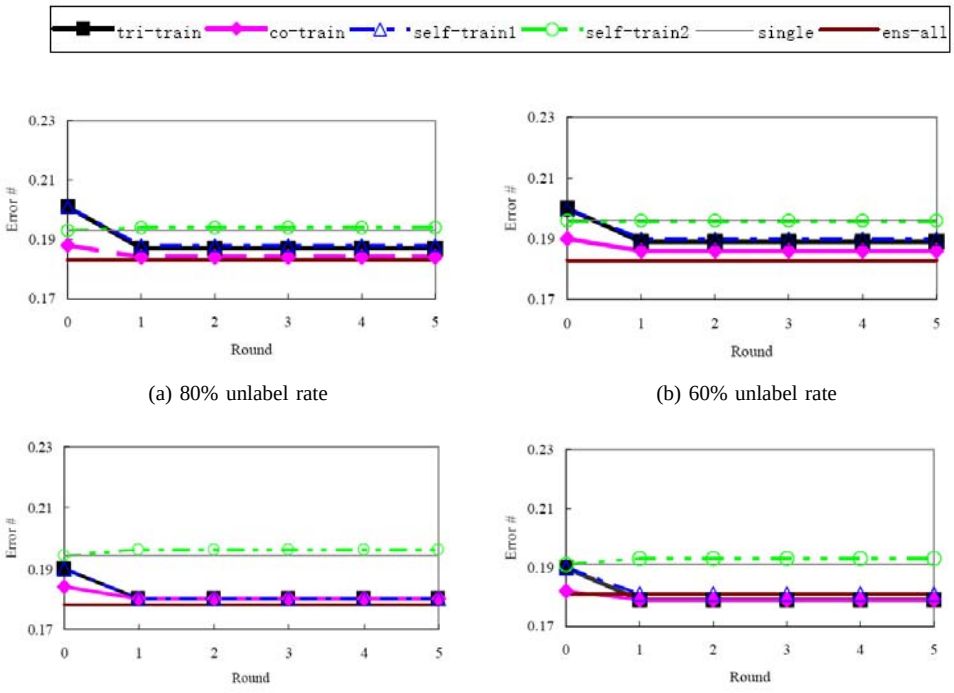


(a) 80% unlabel rate (b) 60% unlabel rate

(c) 40% unlabel rate (d) 20% unlabel rate

Fig. 2. Error rates averaged across all the data sets when BP neural networks are used

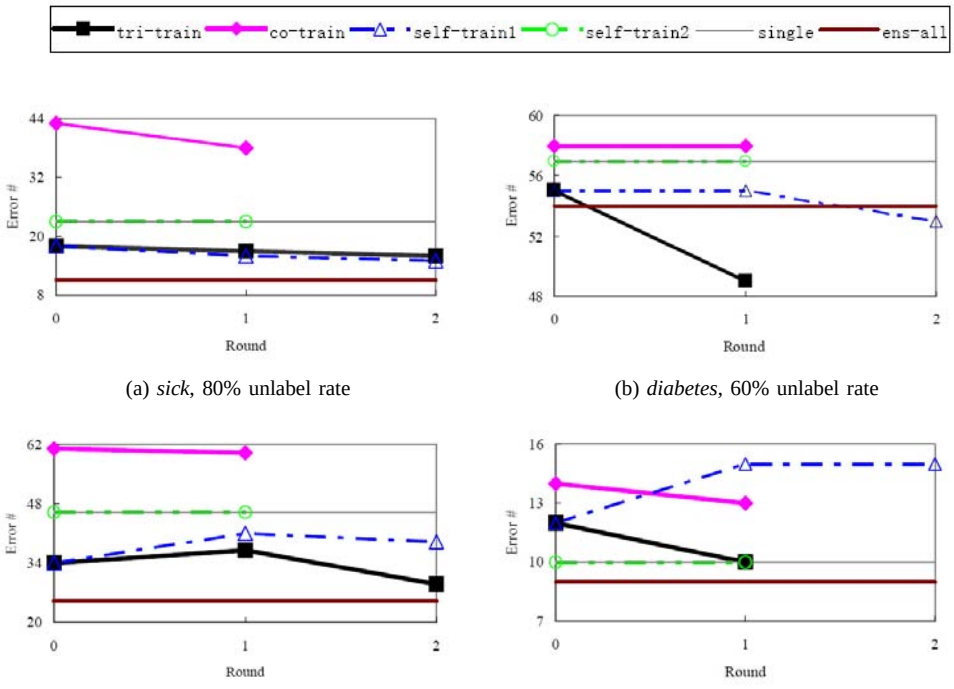
models. Note that after stemming and feature selection, 66 and 5 features are used to train the page-based and hyperlink-based classifiers, respectively. Table VII also presents the results obtained by tri-training, self-training1, and self-training2 on this data set. Note that since these algorithms do not require different views, the page-based and hyperlink-based features are put together in training the individual classifiers. In the table the best final hypothesis and the biggest improvement



(a) 80% unlabel rate (b) 60% unlabel rate

(c) 40% unlabel rate (d) 20% unlabel rate

Fig. 3. Error rates averaged across all the data sets when Naive Bayes classifiers are used



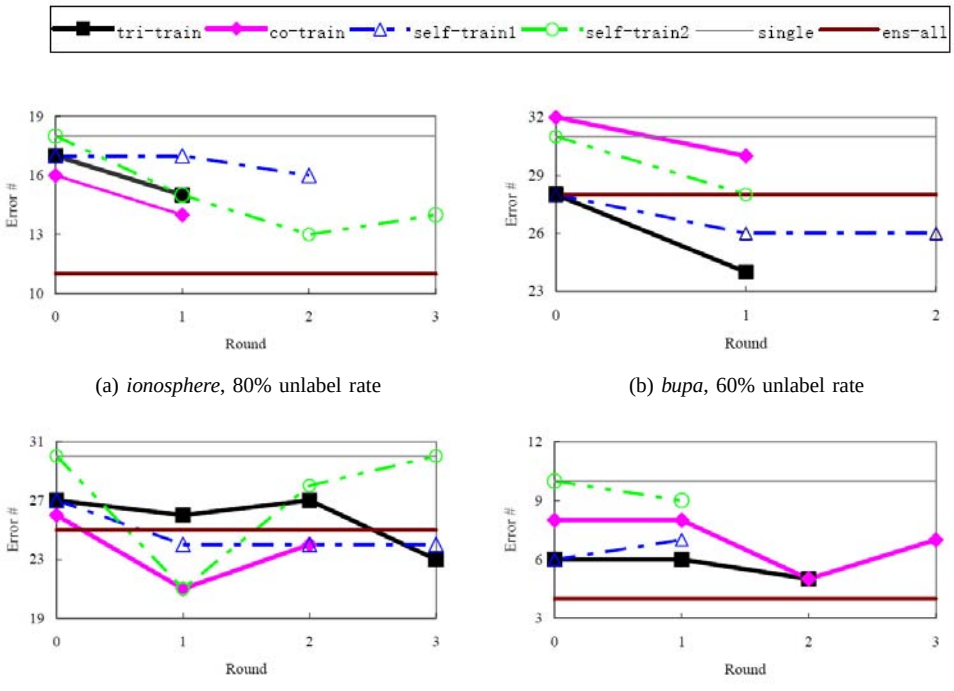
(a) *sick*, 80% unlabel rate (b) *diabetes*, 60% unlabel rate

(c) *tic-tac-toe*, 40% unlabel rate (d) *hypothyroid*, 20% unlabel rate

Fig. 4. Results on data sets where tri-training achieves median improvement (J4.8 decision trees are used)

with each base classifier have been boldfaced.

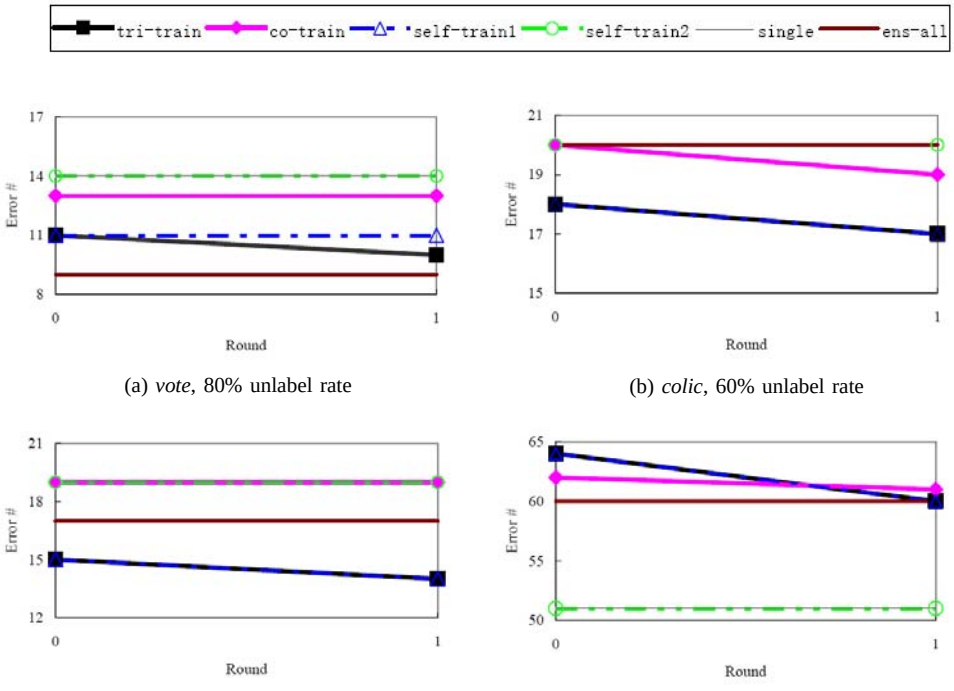
Table VII shows that on the web page classification task, tri-training can effectively utilize unlabeled data to enhance the learning performance. Actually, when Naive Bayes classifiers are used, the improvement of the final hypothesis generated by tri-training is about 13.4%, while when J4.8 decision trees and BP neural networks are used, the improvements are more impressive, i.e. 39.0% and 25.3%, respectively.



(a) *ionosphere*, 80% unlabel rate (b) *bupa*, 60% unlabel rate

(c) *australian*, 40% unlabel rate (d) *wdbc*, 20% unlabel rate

Fig. 5. Results on data sets where tri-training achieves median improvement (BP neural networks are used)



(a) *vote*, 80% unlabel rate (b) *colic*, 60% unlabel rate

(c) *ionosphere*, 40% unlabel rate (d) *german*, 20% unlabel rate

Fig. 6. Results on data sets where tri-training achieves median improvement (Naive Bayes classifiers are used)

Table VII also shows that the improvement of tri-training is bigger than that of the co-training algorithm and the self-training algorithms when J4.8 decision trees and BP neural networks are used. While when Naive Bayes classifiers are used, the improvement of tri-training is smaller than that of co-training and the self-training algorithms. In particular, when J4.8 decision trees and BP neural networks are used, the improvements brought by tri-training are much bigger than that

TABLE VII

THE PERFORMANCES OF TRI-TRAINING, CO-TRAINING, SELF-TRAINING1, AND SELF-TRAINING2 ON THE WEB PAGE CLASSIFICATION PROBLEM

Component learner	J4.8 decision tree			BP neural network			Naive Bayes classifier		
or hypothesis	initial	final	improv	initial	final	improv	initial	final	improv
tri-training									
Component 1	.246 ± .052	.141 ± .023	42.7%	.143 ± .031	.106 ± .019	25.9%	.106 ± .022	.100 ± .030	5.7%
Component 2	.211 ± .089	.141 ± .023	33.2%	.134 ± .049	.107 ± .021	20.1%	.102 ± .024	.095 ± .030	6.9%
Component 3	.215 ± .075	.141 ± .023	34.4%	.186 ± .033	.115 ± .031	38.2%	.142 ± .019	.100 ± .028	29.6%
Hypothesis	.231 ± .068	**.141 ± .023**	**39.0%**	.146 ± .033	**.109 ± .022**	**25.3%**	.112 ± .027	.097 ± .028	13.4%
co-training									
Page-based	.152 ± .032	.172 ± .027	-13.2%	.130 ± .052	.154 ± .030	-18.5%	.113 ± .026	.100 ± .019	11.5%
Hyperlink-based	.159 ± .014	.137 ± .035	13.8%	.160 ± .035	.116 ± .010	27.5%	.157 ± .040	.144 ± .022	8.3%
Hypothesis	.151 ± .030	.144 ± .012	4.6%	.126 ± .028	.116 ± .031	7.9%	.115 ± .019	**.078 ± .017**	**32.2%**
self-training1									
Component 1	.246 ± .052	.212 ± .101	13.8%	.143 ± .031	.110 ± .029	23.1%	.106 ± .022	.090 ± .016	15.1%
Component 2	.211 ± .089	.165 ± .009	21.8%	.134 ± .049	.103 ± .019	23.1%	.102 ± .024	.088 ± .021	13.7%
Component 3	.215 ± .075	.176 ± .027	18.1%	.186 ± .033	.113 ± .013	39.2%	.142 ± .019	.103 ± .028	27.5%
Hypothesis	.231 ± .068	.160 ± .013	30.7%	.146 ± .033	.110 ± .021	24.7%	.112 ± .027	.094 ± .022	16.1%
self-training2									
Component 1	.246 ± .052	.165 ± .009	32.9%	.143 ± .031	.122 ± .029	25.9%	.106 ± .022	.099 ± .020	6.6%
Component 2	.211 ± .089	.165 ± .009	21.8%	.134 ± .049	.122 ± .029	20.1%	.102 ± .024	.099 ± .020	2.9%
Component 3	.215 ± .075	.165 ± .009	23.3%	.186 ± .033	.122 ± .029	38.2%	.142 ± .019	.099 ± .020	30.3%
Hypothesis	.165 ± .009	.165 ± .009	0.0%	.114 ± .025	.122 ± .029	-7.0%	.116 ± .019	.099 ± .020	14.7%

brought by co-training; while when Naive Bayes classifiers are used, the improvement of tri-training is much smaller than that of co-training. However, Table VII shows that the component Naive Bayes classifiers refined by tri-training are better than these refined by co-training. This may imply that although the majority voting scheme used by tri-training is effective in combining the component J4.8 decision trees and BP neural networks, it may be far less effective than the combination scheme used by the co-training algorithm in combining the component Naive Bayes classifiers.

Actually, through observing Table VII it can be found that when J4.8 decision trees and BP neural networks are used, the page-based classifiers degenerate in the co-training process, but the final hypotheses of co-training are still not bad. This suggests that the combination scheme used by the co-training algorithm may play an important role. Moreover, the co-training algorithm evidently utilizes the advantages offered by the two sufficient and redundant views, because even when one component classifier has degenerated, the improvement of the other component classifier can still enable the improving of the final hypothesis. This confirms the claim raised by Blum and Mitchell [5], that is, when there exist sufficient and redundant views, appropriately utilizing them will benefit the learning performance.

V. CONCLUSION

In this paper, the tri-training algorithm is proposed. Through employing three classifiers, tri-training is facilitated with good efficiency and generalization ability because it could gracefully choose examples to label and use multiple classifiers to compose the final hypothesis. Moreover, its applicability is wide because it neither requires sufficient and redundant views nor does it put any constraint on the employed supervised learning algorithm. Experiments on UCI data sets and application to web page classification indicate that although the algorithm is simple, it could exploit unlabeled examples effectively.

Note that the performance of semi-supervised learning algorithms are usually not stable because the unlabeled examples may often be wrongly labeled during the learning process [4] [20]. A promising solution to this problem may be using *data editing* mechanisms, such as the one described in [18], to help identify the wrongly labeled examples. Incorporating data editing mechanisms into tri-training and other semi-supervised learning algorithms is an interesting issue to be investigated in future work.

Ensemble learning techniques [12], in particular, Boosting, have already been introduced into semi-supervised learning [2] [9]. It is evident that the working style of tri-training exhibits a new way to exploit ensemble techniques in this area. However, in its current form, such an exploitation is very limited because there are only three classifiers. Although previous research has shown that using three classifiers to make an ensemble could already improve the generalization ability [22], better performance can be anticipated with more classifiers, which is another interesting future issue.

Besides semi-supervised learning, unlabeled examples can be exploited by active learning [7], where the labels of some selected unlabeled examples are asked from the user. The employment of ensemble techniques in tri-training enables the introduction of a classic active learning method, i.e. *query-by-committee* [25]. Roughly, the most disagreed unlabeled example by the classifiers can be selected to query. Designing effective algorithm to combine tri-training with query-by-committee is an issue well-worth studying.

Moreover, in the present implementation of tri-training, the classifiers are re-trained in each round. If the base learners are incremental algorithms, it might be feasible for the classifiers

to learn only the newly labeled examples, which could help improve the efficiency. This is also an interesting issue to be explored in the future.

ACKNOWLEDGEMENT

The authors want to thank the anonymous reviewers for their constructive comments, especially for the suggestions on the comparison with self-training algorithms.

REFERENCES

[1] D. Angluin and P. Laird, "Learning from noisy examples," Machine Learning, vol.2, no.4, pp.343–370, 1988.
[2] K.P. Bennett, A. Demiriz, and R. Maclin, "Exploiting unlabeled data in ensemble methods," in Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Canada, pp.289–296, 2002.
[3] C. Blake, E. Keogh, and C.J. Merz, "UCI repository of machine learning databases" [http://www.ics.uci.edu/∼mlearn/MLRepository.html], Department of Information and Computer Science, University of California, Irvine, CA, 1998.
[4] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," in Proceedings of the 18th International Conference on Machine Learning, Williamston, MA, pp.19–26, 2001.
[5] 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, pp.92–100, 1998.
[6] L. Breiman, "Bagging predictors," Machine Learning, vol.24, no.2, pp.123–140, 1996.
[7] D. Cohn, L. Atlas, and R. Ladner, "Improved generalization with active learning," Machine Learning, vol.15, no.2, pp.201–221, 1994.
[8] M. Collins and Y. Singer, "Unsupervised models for named entity classifications," in Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, College Park, MD, pp.100–110, 1999.
[9] F. D'Alché-Buc, Y. Grandvalet, and C. Ambroise, "Semi-supervised marginboost," in Advances in Neural Information Processing Systems 14, T.G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, pp.553–560, 2002.
[10] 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, pp.375–382, 2002.
[11] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," Journal of the Royal Statistical Society, Series B, vol.39, no.1, pp.1–38, 1977.
[12] T.G. Dietterich, "Ensemble methods in machine learning," in Lecture Notes in Computer Science 1867, J. Kittler and F. Roli, Eds. Berlin: Springer, pp.1–15, 2000.
[13] B. Efron and R. Tibshirani, An Introduction to the Bootstrap, New York: Chapman & Hall, 1993.
[14] S. Goldman and Y. Zhou, "Enhancing supervised learning with unlabeled data," in Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, pp.327–334, 2000.
[15] R. Hwa, M. Osborne, A. Sarkar, and M. Steedman, "Corrected co-training for statistical parsers," in Working Notes of the ICML'03 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, Washington, DC, 2003.
[16] T. Joachims, "Transductive inference for text classification using support vector machines," in Proceedings of the 16th International Conference on Machine Learning, Bled, Slovenia, pp.200–209, 1999.
[17] D.J. Miller and H.S. Uyar, "A mixture of experts classifier with learning based on both labelled and unlabelled data," in Advances in Neural Information Processing Systems 9, M. Mozer, M.I. Jordan, and T. Petsche, Eds. Cambridge, MA: MIT Press, pp.571–577, 1997.
[18] F. Muhlenbach, S. Lallich, and D.A. Zighed, "Identifying and handling mislabelled instances," Journal of Intelligent Information Systems, vol.22, no.1, pp.89–109, 2004.
[19] 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, pp.86–93, 2000.
[20] K. Nigam, A.K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," Machine Learning, vol.39, no.2–3, pp.103–134, 2000.
[21] 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, pp.1–9, 2001.
[22] J.R. Quinlan, "MiniBoosting decision trees," [http://www.cse.unsw.edu.au/∼quinlan/miniboost.ps], 1998.
[23] E. Riloff and R. Jones, "Learning dictionaries for information extraction by multi-level bootstrapping," in Proceedings of the 16th National Conference on Artificial Intelligence, Orlando, FL, pp.474–479, 1999.
[24] A. Sarkar, "Applying co-training methods to statistical parsing," in Proceedings of the 2nd Annual Meeting of the North American Chapter of the Association for Computational Linguistics, Pittsburgh, PA, pp.95–102, 2001.
[25] H. Seung, M. Opper, and H. Sompolinsky, "Query by committee," in Proceedings of the 5th Annual ACM Conference on Computational Learning Theory, Pittsburgh, PA, pp.287–294, 1992.
[26] B. Shahshahani and D. Landgrebe, "The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon," IEEE Transactions on Geoscience and Remote Sensing, vol.32, no.5, pp.1087–1095, 1994.
[27] 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 the 11th Conference on the European Chapter of the Association for Computational Linguistics, Budapest, Hungary, pp.331–338, 2003.
[28] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, Cambridge, MA, pp.189–196, 1995.
[29] I.H. Witten and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, San Francisco, CA: Morgan Kaufmann, 2000.

Zhi-Hua Zhou (S'00-M'01) received the BSc, MSc and PhD degrees in computer science from Nanjing University, China, in 1996, 1998 and 2000, respectively, all with the highest honor. He joined the Department of Computer Science & Technology of Nanjing University as a lecturer in 2001, and is a professor and head of the LAMDA group at present. His research interests are in artificial intelligence, machine learning, data mining, pattern recognition, information retrieval, neural computing, and evolutionary computing. In these areas he has published over 40 technical papers in refereed international journals or conference proceedings. He has won the Microsoft Fellowship Award (1999), the National Excellent Doctoral Dissertation Award of China (2003), and the Award of National Science Fund for Distinguished Young Scholars of China (2003). He is an associate editor of *Knowledge and Information Systems*, and on the editorial boards of *Artificial Intelligence in Medicine*, *International Journal of Data Warehousing and Mining*, *Journal of Computer Science & Technology*, and *Journal of Software*. He served as program committee member for various international conferences and chaired many native conferences. He is a senior member of China Computer Federation (CCF) and the vice chair of CCF Artificial Intelligence & Pattern Recognition Society, a councilor of Chinese Association of Artificial Intelligence (CAAI), the vice chair and chief secretary of CAAI Machine Learning Society, and a member of IEEE and IEEE Computer Society.

Ming Li received his BSc degree in computer science from Nanjing University, China, in 2003. He got some awards such as peoples' scholarship for outstanding undergraduate. Currently he is a PhD candidate at the Department of Computer Science & Technology of Nanjing University. He is a member of the LAMDA Group. His research interests are in machine learning and data mining, especially in learning with unlabeled examples.