Ensemble approaches to semi-supervised learning
Tanha, J.

Citation for published version (APA):
Tanha, J. (2013). Ensemble approaches to semi-supervised learning

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
This chapter presents a method that uses an ensemble of classifiers for co-training rather than feature subsets. The ensemble is used to estimate the probability of incorrect labeling and this is used with a theorem by Angluin and Laird [1] to derive a measure for deciding if adding a set of unlabeled data will reduce the error of a component classifier or not. Our method does not require a time consuming test for selecting a subset of unlabeled data. Experiments show that in most cases our method outperforms similar methods.

This chapter is an extension of the published paper by Tanha et al. [93], at the 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI) in 2011.

4.1 Introduction

Here we consider co-training [8] as one of the widely used semi-supervised learning methods. Co-training involves two, preferably independent, views of data both of which are individually sufficient for training a classifier. In co-training, the “views” are subsets of the features that describe the data. Each classifier predicts labels for the unlabeled data and a degree of confidence. Unlabeled instances that are labeled with high confidence by one classifier are used as training data for the other. This is repeated until none of classifiers changes. Algorithm 7 gives an overview of the co-training algorithm. Another approach to co-training is to use different learning algorithms instead of feature subsets as in [39]. This version uses statistical tests to decide which unlabeled data should be labeled with confidence.

In this chapter, we propose two improvements for co-training. We call the resulting method Ensemble-Co-Training. First we consider co-training by an ensemble of $N$ classifiers that are trained in parallel by different learning algorithms and second we derive a stop-criterion based on the training error rate, using a theorem by Angluin and Laird [1] that describes the effect of adding uncertain data.
Algorithm 7 Co-Training

Initialize: L, U;
L: Labeled data; U: Unlabeled data; N: Number of iterations;
X₁ and X₂ are two views of feature space;
Sample a set of unlabeled examples U′ from U;

\[ t \leftarrow 1 \]

for each iteration \( i < N \) do
  Train classifier \( f₁ \) regarding \( X₁ \) as feature space using labeled data L;
  Train classifier \( f₂ \) regarding \( X₂ \) as feature space using labeled data L;
  Allow \( f₁ \) to assign label to unlabeled examples in \( U' \);
  Allow \( f₂ \) to assign label to unlabeled examples in \( U' \);
  Select a set of high confidence predictions
  Add newly-labeled examples to L;
  Reproduce \( U' \) from U;

end for

Output: Generate final hypothesis;

Training an ensemble that votes on the labels for unlabeled data has the advantage that better estimates of confidence of predictions are obtained. This is useful because semi-supervised learning is used in settings where only a small amount of labeled data is available. The stop-criterion is useful because it prevents the extra iterations. Our proposed method uses Theorem 1, which has two parameters taken from PAC-learning (Probably Approximately Correct) [24], with an intuitive meaning: the probability and the size of a prediction error. We derive the algorithm from this theorem. We then perform a set of experiments on UCI dataset to show the effect of the proposed method. Our experiments show that Ensemble-Co-Training improves the classification performance and outperforms the other related methods.

The rest of this chapter is organized as follows. Section 4.2 outlines different kinds of co-training algorithms. In Section 4.3 we derive the Ensemble-Co-Training algorithm. Section 4.4 describes the experimental setup. In Section 4.5 we compare the results of Ensemble-Co-Training on the UCI datasets. Finally, in Section 4.6, we present our conclusion and discussion.

4.2 Semi-Supervised Co-Training

We distinguish two categories of co-training methods in terms of the number of classifiers and views of data, which are learning with multiple views, and learning with a single view and multiple classifiers.

The original co-training method uses multiple views of the data. As mentioned in the introduction, the co-training paradigm that was proposed by Blum
4.2. Semi-Supervised Co-Training

and Mitchell [8] works well in domains that naturally have multiple views of data. Nigam and Ghani [72] analyzed the effectiveness and applicability of co-training when there are no two natural views of data. They show that when independent and redundant views exist, co-training algorithms outperform other algorithms using unlabeled data, otherwise there is no difference. However in practice, many domains are not described by a large number of attributes that can naturally be split into two views. The result of randomly partitioned views is not always effective [72]. The authors of [72] proposed a new multi-view co-training algorithm, which is called co-EM. This algorithm combines multi-view learning with the probabilistic EM model. The Naive Bayes classifier is used to estimate class labels.

Another type of co-training algorithm uses a learning method with a single view and multiple classifiers. An example of this approach was developed by Goldman and Zhou [39]. They propose a method which trains two different classifiers with different learning algorithms. Their method uses a statistical test to select unlabeled data for labeling as selection metric. The rest of the co-training process in their method is the same as the original co-training. The same authors proposed Democratic Co-Learning [107]. In democratic co-learning, a set of different learning algorithms is used to train a set of classifiers separately on the labeled data set in self-training manner. They use a statistical method for selecting unlabeled data for labeling.

Zhou and Li [109] propose the Tri-Training method, which is a single view and multiple classifiers type of co-training algorithm. The proposed method uses a base learning algorithm to construct three classifiers from different sub-samples. The classifiers are first trained on data sets that are generated from the initial labeled data via bootstrap sampling [13]. Two classifiers then predict labels for the third classifier. Predictions are made via majority voting by the three final classifiers. Tri-Training does not need multiple views of data nor statistical testing. Tri-Training can be done with more than three classifiers, which gives a method called Co-Forest [60]. A problem with this approach in the context of semi-supervised learning is that there is only a small amount of labeled data and therefore it is not possible to select subsamples that vary enough and are of a sufficient size. Furthermore, ensemble learning methods need different classifiers [54] to be effective, but, as mentioned earlier, when there is only a small amount of labeled data then all initial training data that are generated by bagging methods will be roughly the same. Therefore, in this case the ensemble approach will not be effective.

There are many applications for the original co-training approach and co-EM, for example see [12]. In [88] an ensemble of decision trees is used for image classification. The basic idea of this algorithm is to start with a limited number of labeled images, and gradually propagate the labels to the unlabeled images. At each iteration a set of newly-labeled examples is added to the training set to improve the performance of the decision tree learning algorithm. The improved
decision trees are used to propagate labels to more images, and the process is repeated when it reaches the stopping condition. This algorithm directly uses the predictions of weak decision tree classifiers, which are not accurate enough. Adding mislabeled instances may lead to worse results.

In this chapter we propose a single view and multiple classifiers type of co-training algorithm. The main idea of our approach is to use multiple base classifiers with different learning algorithms instead of using same base learner on the different subsamples of original labeled data. In this approach, the algorithm does not require independent and redundant attributes, but instead it needs $N$ different hypotheses as in ensemble methods. For this, we employ $N$ different learning algorithms. This approach also tends to select a subset of high-confidence predictions at each iteration using different classifiers, which is more challenging in self-training. We use a notion from PAC-learning for controlling the error rate, selecting a subset of high-confidence predictions by ensemble, and deciding when to stop the training process.

Finally, our proposed method is related to “Disagreement Boosting” [58], which performs a form of boosting in which examples are re-weighted not by their predictability but by disagreement between multiple classifiers. Both methods use agreement between trained classifiers. In the boosting approach the learning algorithm constructs a hypothesis in each iteration and incorporates these in a linear combination where in our approach each learning algorithm contributes a hypothesis. These are then combined to make a prediction.

4.3 A Bound for Learning from noisy data

Two key issues in co-training are: (1) measuring the confidence in labels that are predicted for the unlabeled data, and (2) a criterion for stopping the training process [91]. Co-training aims at adding a subset of the high-confidence predictions, called newly-labeled examples. At some point labels will be noisy and cause the result of learning to become worse. This is a form of “overfitting”. Problems (1) and (2) can be solved in an empirical way, by using a holdout set of labeled data to assess the effect of adding newly-labeled data. However, since semi-supervised learning is used for learning tasks where labeled data is scarce, this is not a good solution. Instead, we propose an analytic solution for solving this problem. This can be summarized as follows. We use a theorem by Angluin and Laird [1] that relates the number of training data to the probability that a consistent hypothesis has an error larger than some threshold for a setting with training data and with a certain error in the labels. We use an ensemble of learners for co-training instead of two and the agreement among the predictions of labels for the unlabeled data to obtain an estimate of the labeling error rate. Using this, we can estimate the effect of learning on the error of the result of adding the newly-labeled data to the training set. This is used to decide which subset of high-confidence
predictions should be added to the initial labeled data in order to improve the classification performance. Finally, the training process will be stopped when the estimated error rate in the initial labeled data is expected to increase. Figure 4.1 shows the general overview of Ensemble-Co-Training. In this section we review the theorem that we use and show how it can be used to define a criterion for adding newly-labeled data. The entire algorithm is presented in section 4.3.2.

### 4.3.1 Criterion for error rate and number of unlabeled data

As addressed in the Introduction chapter, semi-supervised learning is the learning task, when there is a limited number of labeled data and a large amount of unlabeled data. Data points is divided into two parts: the points \( X_l = (x_1, x_2, \ldots, x_l) \), which are labeled by \( Y_l = \{+1, -1\} \), and the points \( X_u = (x_{l+1}, x_{l+2}, \ldots, x_{l+u}) \), the labels of which are not known. We consider \( l \ll u \), where \( l \) and \( u \) are the number of labeled data and unlabeled data respectively, which is more suitable for semi-supervised setting.

Here in order to find criteria for selecting examples and for deciding when to stop, we need an estimator for the error rate. As we mentioned earlier, using a hold-out set for this is not attractive, because we have little labeled data. We
cannot expect that the effect gives substantial improvements on the labeled data either. This motivates a need for an estimate of the effect of labeling unlabeled data and adding them to the training data. Inspired by [39] and [109], we formulate a function that estimates the true classification error of a hypothesis from the size of the training set and the probability that a data point is mislabeled. This is based on a theorem, in the style of PAC-learning, by Angluin and Laird [1]. This theorem is as follows.

**Theorem 1.** If we draw a sequence $\sigma$ of $m$ data points where each data point has a probability $\eta$ of being mislabeled, and we compute the set of hypotheses that are consistent with $\sigma$ then if

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

holds, where $\epsilon$ is the classification error of the worst remaining candidate hypothesis on $\sigma$, $\eta (< 0.5)$ is an upper bound on the noise rate in the classifications of the data, $N$ is the number of hypotheses, and $\delta$ is a probability that expresses confidence, then for a hypothesis $H_i$ that minimizes disagreement with $\sigma$ holds that:

$$\Pr[d(H_i, H^*) \geq \epsilon] \leq \delta$$

where $d(\cdot)$ is the sum over the probabilities of differences between classifications of the data according to hypothesis $i$ and the actual data.

To construct an estimator for the error of a hypothesis, $\epsilon$, we rewrite the above inequality as follows. First, we set $\delta$ to a fixed value, which means that we assume that the probability of an error is equal for all hypotheses, and second assume that $N$ is (approximately) constant between two iterations.

Here, we introduce $c$ such that $c = 2\lambda \ln\left(\frac{2N}{\delta}\right)$, where $\lambda$ is chosen so that equation (4.1) holds. Substituting $c$ in (4.1) gives:

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

So, reformulating (4.3), then gives:

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

Based on this corollary, the error rate $\epsilon$ can be controlled. In particular the change in the (bound on the) error rate can be estimated and used to select the newly-labeled examples and to decide when to stop.

For this, we need to estimate $\eta$. This is done using a set of hypotheses that are constructed by different learning algorithms. Suppose that there are $k$ classifiers, denoted by $H^*$. All classifiers in $H^*$ except $h_j$, called $H_p$ and $p = 1, ..., k$ such that $p \neq j$, predict labels for the unlabeled data based on voting methods. Then the newly-labeled data is used for $h_j$, component classifier, in the next iteration.
of the training process. In more detail, let $L$, $U$, and $L_{i,j}$ denote the labeled data, unlabeled data, and the newly-labeled instances for the $j$th classifier in the $i$th iteration of training process respectively. Moreover, in the $i$th iteration, a component classifier $h_j$ has an initial labeled data with size $|L|$ and a number of newly-labeled data with size $|L_{i,j}|$, determined by the ensemble $H_p$. Assume that the error rate of $H_p$ on $L_{i,j}$ is $\hat{e}_{i,j}$. Then the number of instances that are mislabeled by $H_p$ in $L_{i,j}$ is estimated as $\hat{e}_{i,j}|L_{i,j}|$, where $\hat{e}_{i,j}$ is upper bound of classification error rate of the $H_p$. The same computation is also done for the initial labeled data such that $\eta_L|L|$, where $\eta_L$ denotes the classification noise rate of $L$. As mentioned above, the training set in the $i$th iteration of the training process is $L \cup L_{i,j}$ for a component classifier $h_j$. In this training set the number of noisy instances are instances in $L$ and instances in $L_{i,j}$. Therefore, the noise rate in this training set can be estimated by:

$$
\eta_{i,j} = \frac{\eta_L|L| + \hat{e}_{i,j}|L_{i,j}|}{|L| + |L_{i,j}|} \quad (4.5)
$$

As shown, the error rate in the $i$th iteration for a component classifier $h_j$ is estimated by (4.5). Since $c$ in (4.4) is a constant, for simplicity assume that $c = 1$, substituting (4.5) in (4.4), when training is in the $i$th iteration, gives:

$$
n_{i,j} \simeq \frac{1}{\hat{e}_{i,j}} = (|L| + |L_{i,j}|)(1 - 2\eta_{i,j})^2. \quad (4.6)
$$

From this we derive a criterion for whether adding data reduces the error of the result of learning or not.

**Theorem 2.** If the following inequality satisfies in the $i$th and $(i-1)$th iteration:

$$
\frac{\hat{e}_{i,j}}{\hat{e}_{i-1,j}} < \frac{|L_{i-1,j}|}{|L_{i,j}|} < 1 \quad (4.7)
$$

where $j = 1, 2, ..., k$, then in the $i$th and $(i-1)$th iteration, the worst-case error rate for a component classifier $h_j$ satisfies $\epsilon_{i,j} < \epsilon_{i-1,j}$.

**Proof:**

Given the inequalities in (4.7), then will have:

$$
|L_{i,j}| > |L_{i-1,j}| \quad \text{and} \quad \hat{e}_{i,j}|L_{i,j}| < \hat{e}_{i-1,j}|L_{i-1,j}| \quad (4.8)
$$

thus, it can be easily shown that:

$$
|L| + |L_{i,j}| > |L| + |L_{i-1,j}| \quad (4.9)
$$

and then

$$
\frac{\eta|L| + \hat{e}_{i,j}|L_{i,j}|}{|L| + |L_{i,j}|} < \frac{\eta|L| + \hat{e}_{i-1,j}|L_{i-1,j}|}{|L| + |L_{i-1,j}|} \quad (4.10)
$$
According to (4.5) and (4.10) will have:

\[
\eta_{i,j} = \frac{\eta |L| + \hat{\epsilon}_{i,j}|L_{i,j}|}{|L| + |L_{i,j}|} \quad \text{and} \quad \eta_{i-1,j} = \frac{\eta |L| + \hat{\epsilon}_{i-1,j}|L_{i-1,j}|}{|L| + |L_{i-1,j}|}
\]  

(4.11)

hence, \( \eta_{i,j} < \eta_{i-1,j} \) and then it can be easily written as:

\[
n_{i-1,j} = (|L| + |L_{i-1,j}|)(1 - 2\eta_{i-1,j})^2
\]

(4.12)

and

\[
n_{i,j} = (|L| + |L_{i,j}|)(1 - 2\eta_{i,j})^2
\]

(4.13)

So, according to \( \eta_{i,j} < \eta_{i-1,j} \) and (4.10) will have; \( n_{i,j} > n_{i-1,j} \), and since according to (4.4) \( n \propto \frac{1}{\epsilon^2} \), then \( \epsilon_{i,j} < \epsilon_{i-1,j} \).

Theorem 2 can be interpreted as saying that if the inequality (4.7) is satisfied, then the worst-case error rate of a component classifier \( h_j \) will be iteratively reduced, and the size of newly-labeled instances is iteratively increased in the training process.

As can be derived from (4.7), \( \hat{\epsilon}_{i,j} < \hat{\epsilon}_{i-1,j} \) and \( |L_{i,j}| > |L_{i-1,j}| \) should be satisfied at the same time. However, in some cases \( \hat{\epsilon}_{i,j}|L_{i,j}| < \hat{\epsilon}_{i-1,j}|L_{i-1,j}| \) may be violated because \( |L_{i,j}| \) might be much larger than \( |L_{i-1,j}| \). When this occurs, in order not to stop the training process, a subsample of \( |L_{i,j}| \) are randomly selected such that new \( |L_{i,j}| \) satisfies:

\[
|L_{i,j}| < \frac{\hat{\epsilon}_{i-1,j}|L_{i-1,j}|}{\hat{\epsilon}_{i,j}},
\]

(4.14)

The condition in inequality (4.7) is used for stopping the training process and controlling the number of newly-labeled data as well as the error rate in the Ensemble-Co-Training algorithm. In Section 4.3.2 we present the Ensemble-Co-Training algorithm based on the analysis that we did in this section. This condition is based on several assumptions. The theorem holds for a "candidate elimination" type of learning algorithm rather than a best hypothesis estimator that learns from examples that are drawn at random. In our application the examples are selected and a single hypothesis is constructed in each iteration. At the end the multiple hypotheses are combined in an ensemble. An overview of our proposed algorithm is presented in Figure 4.2.

4.3.2 Ensemble-Co-Training Algorithm

In Ensemble-Co-Training each component classifier \( h_j \) is first trained on the original labeled data. Then \( H_p \) is built using all classifiers except one. This ensemble is used to select a set of high-confidence predictions using voting methods. Next,
4.3. A Bound for Learning from noisy data

- Initialize: $L, U, H$
- $L$: Labeled data; $U$: Unlabeled data;
- $H$: Ensemble of classifiers;
- At each iteration $i$:
  1. for $j \in \{1, 2, \ldots, k\}$
     - Find $\hat{e}_{i,j}$ as error rate for component classifier $h_j$ based on disagreement among classifiers
     - Assign labels to the unlabeled examples based on the agreement among ensemble $H_p$
     - Sample high-confidence examples for component classifier $h_j$
     - Build the component classifier $h_j$ based on newly-labeled and original labeled examples
  2. Control the error rate for each component classifier based on inequality (4.7)
     - Update ensemble $H$
   - Generate final hypothesis

Figure 4.2: An outline of the Ensemble-Co-Training

$H_p$ estimates the error rate for each component classifier. After that, a subset of $U$ is selected by ensemble $H_p$ for a component classifier $h_j$. A pre-defined threshold is used for selecting high-confidence predictions. Data that have an improvement of error above the threshold are added to the labeled training data. Note that each classifier has its own training set through the training process. This avoids that classifiers converge too early and strengthens the effect of the ensemble. The data that is labeled for the classifier is not removed from the unlabeled data $U$ to allow it to be labeled for other classifiers as well. This training process is repeated until there are no more data that can be labeled such that they improve the performance of any classifier. An outline of the Ensemble-Co-Training algorithm is shown in Figure 4.2.

In the Ensemble-Co-Training algorithm instead of computing $\epsilon_{i,j}$, we use the disagreement among classifiers as error rate, called $\hat{e}_{i,j}$. Through the training process the algorithm attempts to decrease the disagreement among classifiers in order to improve the performance.

Note that inequality (4.7) sometimes could not be satisfied, because the size of $L_{i,j}$ is much larger than the size of $L_{i-1,j}$. In this case the training process cannot reduce the error rate due to stopping the training process earlier. This is because of the fact that the error rate is a worse case error rather than an expectation. To solve this problem, a subsample of $L_{i,j}$ is randomly selected that does satisfy inequality (4.7), i.e., $\hat{e}_{i,j}|L_{i,j}| < \hat{e}_{i-1,j}|L_{i-1,j}|$, then will have:

$$|L_{i,j}| = \left[ \frac{\hat{e}_{i-1,j}|L_{i-1,j}|}{\hat{e}_{i,j}} \right] - c'.$$  \hspace{1cm} (4.15)
In (4.14) the size of $L_{i-1,j}$ should be restricted by some criterion. Intuitively, it can be bounded by:

$$|L_{i-1,j}| = \left\lfloor \frac{\hat{e}_{i,j}}{e_{i-1,j} - \hat{e}_{i,j}} \right\rfloor + d'$$

(4.16)

To simplify (4.15) and (4.16) assume that $c' = d' = 1$.

Figure 4.3 presents the pseudo-code of the Ensemble-Co-Training algorithm in more detail. The Ensemble-Co-Training algorithm uses inequality (4.7), (4.15), and (4.16) as well as three main functions: EstimateError, LabelingUnlabeled, and finalHypothesis. The EstimateError($H_p, L$) function estimates the classification error rate of hypothesis derived from combination of $h_1, ..., h_k$ as the disagreement among classification models such that $k \neq j$ for the component classifier $h_j$ on the training data. Note that our assumption is that the training data and the newly-labeled data have the same data distribution. Since estimating the classification error rate on the unlabeled instances cannot be done easily, in Ensemble-Co-Training only the initial labeled data are used for estimating the error rate.

We use two different error rate estimations in Ensemble-Co-Training. The first approach is the disagreement among predictions by hypotheses produced by different learning algorithms. In addition to this we check if a set of new examples does not increase the error on the labeled data. The second approach is the use of the classification confidence to estimate the error rate.

The LabelingUnlabeled function labels a subset of high-confidence predictions by $H_p$ for the component classifier $h_j$. After the last iteration the resulting hypotheses are combined into an ensemble classifier. We experiment with two methods for combining hypotheses: “Average of Probabilities” and “Majority Voting” [52]. Suppose that $Y = (y_1, y_2, ..., y_m)$ are the class labels and there are $K$ classifiers. The “Average of Probabilities” voting method for prediction of the new example $x$ is computed as follows:

$$\arg \max_m \left( \frac{1}{K} \sum_{i=1}^K p_i(y_m|x) \right),$$

(4.17)

For the “Majority Voting” method, the maximum number of classifiers is considered as the main rule. It means that the majority of the classifiers should agree to assign label $y_m$ to the instance $x$.

Finally, the finalHypothesis function forms the final classification model based on voting methods. In the experiments, we compare the Ensemble-Co-Training algorithm without applying the criterion that we mentioned earlier, to show the impact of the criterion on the performance. In particular we use a version of Democratic Co-Learning [107], which is another form of self-training with multiple classifiers, without the conservative statistical test.
Ensemble-Co-Training(L, U, H, θ)

Input: L: Initial labeled data, U: Unlabeled instances,
H: The N Classifiers,
θ: The pre-defined confidence Threshold
{P_l}^{M}_{l=1}: Prior probability

Begin
for j ∈ {1,2,...,k} do
    h_j ← Learn(Classifier_j ,L)
    L_{0,j} ← 0
    \hat{e}_{0,j} ← 0.5 // Upper bound of error rate
end // for

i ← 0 // Number of iteration
repeat
    i ← i + 1
    for j ∈ {1,2,...,k}
        \hat{e}_{i,j} ← EstimateError(H_p, L)
        if(\hat{e}_{i,j} < \hat{e}_{i-1,j}) then
            for each x ∈ U do
                if(Confidence(x, H_p ) > θ)
                LabelingUnlabeled(x,H_p)
                L_{i,j} ← L_{i,j} ∪ \{(x,H_p)\}
            end // for
        end // if
        if( |L_{i-1,j}| = 0 ) then
            L_{i-1,j} ← \lceil \frac{\hat{e}_{i,j}}{L_{i,j}} + 1 \rceil
        end // if
        if( |L_{i-1,j}| < |L_{i,j}| and \hat{e}_{i,j}|L_{i,j}| < \hat{e}_{i-1,j}|L_{i-1,j}| ) then
            update_{i,j} ← TRUE
        else if( |L_{i-1,j}| > \frac{\hat{e}_{i,j}}{L_{i,j}} ) then
            for each class l, sample n_l ∝ P_l as
            L_{i,j} ← Subsample(L_{i,j},\lceil \frac{\hat{e}_{i-1,j}|L_{i-1,j}|}{\hat{e}_{i,j}} - 1 \rceil )
            update_{i,j} ← TRUE
        end // if
    end // for j
    for j ∈ {1,2,...,k}
        if( update_{i,j}= TRUE ) then
            h_j ← Learn(Classifier_j ,L ∪ L_{i,j})
            \hat{e}_{i-1,j} ← \hat{e}_{i,j}
            L_{i-1,j} ← |L_{i,j}|
        end // if
    end // for j
until ( none of h_i changes )
end // repeat

Output: Final H(x) ← argmax \sum_{i:h_i(x)=y} 1

End

Figure 4.3: Ensemble-Co-Training Algorithm
4.4 Experiments

In the experiments we compare Ensemble-Co-Training (ECT) with several other algorithms. One comparison is with using the base learner on the labeled data only for supervised learning and the second is with semi-supervised learning algorithms. The purpose is to evaluate if Ensemble-Co-Training exploits the information in the unlabeled data. We also compare Ensemble-Co-Training to the variations that use different voting strategies for LabelingUnlabeled and EstimateError functions. Furthermore, we include comparisons with the algorithms for semi-supervised learning, in particular self-training, Tri-Training, and Co-Forest.

In our experiments, we use the WEKA implementation of the classifiers with default parameter settings [42].

4.4.1 Datasets

We use UCI datasets for assessing the proposed algorithms. We selected these datasets, because they vary in the number of features and examples, and distribution of classes. Table 4.1 summarizes the specification of eight benchmark datasets from the UCI data repository [32] which are used in our experiments.

Table 4.1: Overview of Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Attributes</th>
<th>Size</th>
<th>Perc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>6</td>
<td>345</td>
<td>58</td>
</tr>
<tr>
<td>Colic</td>
<td>22</td>
<td>368</td>
<td>63</td>
</tr>
<tr>
<td>Diabetes</td>
<td>6</td>
<td>768</td>
<td>65</td>
</tr>
<tr>
<td>Heart</td>
<td>13</td>
<td>270</td>
<td>55</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>19</td>
<td>155</td>
<td>79</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>34</td>
<td>351</td>
<td>64</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>9</td>
<td>958</td>
<td>65</td>
</tr>
<tr>
<td>Vote</td>
<td>16</td>
<td>435</td>
<td>61</td>
</tr>
</tbody>
</table>

For each dataset, about 30 percent of the data are kept as test set, and the rest is used as training examples, which include a small amount of labeled and a large pool of unlabeled data. Training instances in each experiment are partitioned into 90 percent unlabeled data and 10 percent labeled data, keeping the class proportions in all sets similar to the original data set. In each experiment, eight independent runs are performed with different random partitions of $L$ and $U$. The average results are summarized in Tables 4.2 to 4.7.

4.4.2 Base Learner

In self-training and tri-training, we use C4.4grafted classifier as the base learner, which is a decision tree learner [103] with “grafting” and Laplacian correction,
adaptations that often improve performance in domains with sparse data. The Random Forest [15] approach is employed in the Co-Forest learning method. We describe Ensemble-Co-Training for any number of supervised learning algorithms. Here, we only consider four learners: C4.4grafted, Naive Bayes, Random forest, and J48 with Laplacian Correction algorithm. To make performance comparable in co-forest, self-training, and Ensemble-Co-Training we set the value of predefined threshold \(\theta\) at 0.75 for all classifiers.

### 4.5 Results

In the first experiment we compare the learning algorithms in the supervised setting. Table 4.2 shows the classification accuracies. As can be seen, in general ensemble methods achieve better classification accuracies than a single classifier, which is what we expected. The best performance is boldfaced for each dataset.

Table 4.2: Average Classification Accuracy of Supervised learning with Grafted decision tree (DT), Tri-training, Co-Forest, and Ensemble-Co-Training (ECT)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Grafted DT</th>
<th>Tri-training</th>
<th>Co-Forest</th>
<th>ECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>58.63</td>
<td>56.96</td>
<td>55.97</td>
<td>58.01</td>
</tr>
<tr>
<td>Colic</td>
<td>80.74</td>
<td>71.30</td>
<td>73.98</td>
<td>81.02</td>
</tr>
<tr>
<td>Diabetes</td>
<td>65.35</td>
<td>67.63</td>
<td>66.49</td>
<td>66.40</td>
</tr>
<tr>
<td>Heart Statlog</td>
<td>73.54</td>
<td>78.41</td>
<td>72.73</td>
<td>76.14</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>72.57</td>
<td>80.29</td>
<td>75.71</td>
<td>78.00</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>79.13</td>
<td>77.76</td>
<td>82.48</td>
<td>83.11</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>65.00</td>
<td>65.60</td>
<td>67.78</td>
<td>67.66</td>
</tr>
<tr>
<td>Vote</td>
<td>94.41</td>
<td>87.53</td>
<td>90.79</td>
<td>94.64</td>
</tr>
</tbody>
</table>

In the second experiment we compare the performance of Ensemble-Co-Training, Self-Training, Tri-Training, and Co-Forest in the semi-supervised setting. Table 4.3 shows the classification accuracies of the methods. In four out of eight datasets Ensemble-Co-Training achieves the highest classification accuracies and in the other sets the differences are small.

Here, we wish to find an optimal strategy for labeling the unlabeled examples as well as combining hypotheses using different voting strategies, which are “Averaging Probability” and “Majority voting”. The classification accuracies of these methods are shown in Table 4.4. Using the “Majority voting” method for estimating the error rate improves the classification accuracies of five data sets out of eight in our experiments.

Table 4.5 gives the classification accuracies of using different voting methods for the LabelingUnlabeled function in Ensemble-Co-Training, which are again “Averaging Probability” and “Majority Voting” methods. The best classification
Chapter 4. Ensemble Co-Training

Table 4.3: Average Classification Accuracy of Self-training with Grafted decision tree (DT), Tri-training, Co-Forest, and Ensemble-Co-Training (ECT)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Grafted DT</th>
<th>Tri-training</th>
<th>Co-Forest</th>
<th>ECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>57.01</td>
<td>56.97</td>
<td>56.64</td>
<td>58.58</td>
</tr>
<tr>
<td>Colic</td>
<td>78.91</td>
<td>72.69</td>
<td>76.48</td>
<td>81.25</td>
</tr>
<tr>
<td>Diabetes</td>
<td>65.88</td>
<td>67.72</td>
<td>67.94</td>
<td>66.45</td>
</tr>
<tr>
<td>Heart Statlog</td>
<td>71.78</td>
<td>78.25</td>
<td>74.84</td>
<td>77.44</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>73.20</td>
<td><strong>82.00</strong></td>
<td>81.43</td>
<td>80.29</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>79.76</td>
<td>79.38</td>
<td><strong>86.58</strong></td>
<td>83.48</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>67.35</td>
<td>65.83</td>
<td><strong>68.49</strong></td>
<td>67.96</td>
</tr>
<tr>
<td>Vote</td>
<td>93.94</td>
<td>87.06</td>
<td>92.42</td>
<td><strong>94.99</strong></td>
</tr>
</tbody>
</table>

Table 4.4: Average Classification Accuracy of Ensemble-Co-Training (ECT) using Averaging Probability and Majority voting in the EstimateError function for estimating the error rate

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ECT Averaging Probability</th>
<th>ECT Majority voting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>58.58</td>
<td>59.59</td>
</tr>
<tr>
<td>Colic</td>
<td>81.25</td>
<td>81.39</td>
</tr>
<tr>
<td>Diabetes</td>
<td>66.45</td>
<td>67.02</td>
</tr>
<tr>
<td>Heart Statlog</td>
<td><strong>77.44</strong></td>
<td>76.30</td>
</tr>
<tr>
<td>Hepatitis</td>
<td><strong>80.29</strong></td>
<td>80.00</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>83.48</td>
<td><strong>83.98</strong></td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>67.96</td>
<td><strong>68.02</strong></td>
</tr>
<tr>
<td>Vote</td>
<td>94.99</td>
<td>94.99</td>
</tr>
</tbody>
</table>

accuracies among the different settings in Ensemble-Co-Training are boldfaced in Table 4.5. Using “Majority voting” for the LabelingUnlabeled function in Ensemble-Co-Training achieves the best results.

In the previous experiment (second) we used the default settings for the learning algorithms. In the next experiment we optimize the parameters. This is not possible in practical settings because the amount of data that is needed is normally not available. Yet it puts the non-optimized results in perspective. In particular compare the best current setting of Ensemble-Co-Training and Co-Forest which are almost always the two best methods. Table 4.6 shows the result of this comparison. On average the classification accuracy of Ensemble-Co-Training is 1.99% higher than the Co-Forest algorithm.

Here, we compare the results of using a version of Democratic co-learning, without the conservative statistical test, and Ensemble-Co-Training without the criterion that we discussed. The aim of this comparison is to show the advantages of using the criterion. As can be seen in Table 4.7 there is still some improvement
4.5. Results

Table 4.5: Average Classification Accuracy of Ensemble-Co-Training (ECT) using Average voting and Majority voting in the Labeling Unlabeled function for assignment labels

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ECT Averaging Probability</th>
<th>ECT Majority Voting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>58.58</td>
<td>59.94</td>
</tr>
<tr>
<td>Colic</td>
<td>81.25</td>
<td>82.50</td>
</tr>
<tr>
<td>Diabetes</td>
<td>66.45</td>
<td>67.11</td>
</tr>
<tr>
<td>Heart Statlog</td>
<td>77.44</td>
<td>78.73</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>80.29</td>
<td>81.75</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>83.48</td>
<td>86.71</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>67.96</td>
<td>69.40</td>
</tr>
<tr>
<td>Vote</td>
<td>94.99</td>
<td>94.59</td>
</tr>
</tbody>
</table>

Table 4.6: Average Classification Accuracy and Standard Deviation of Co-Forest and Ensemble-Co-training with the best setting

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Co-Forest</th>
<th>ECT Majority Voting</th>
<th>Improvement%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>56.64(±1.89)</td>
<td>59.94(±2.11)</td>
<td>3.30</td>
</tr>
<tr>
<td>Colic</td>
<td>76.48(±3.42)</td>
<td>82.50(±0.84)</td>
<td>6.02</td>
</tr>
<tr>
<td>Diabetes</td>
<td>67.94(±1.65)</td>
<td>67.11(±1.11)</td>
<td>-0.83</td>
</tr>
<tr>
<td>Heart Statlog</td>
<td>74.84(±3.11)</td>
<td>78.73(±0.48)</td>
<td>3.90</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>81.43(±1.5)</td>
<td>81.75(±2.11)</td>
<td>0.32</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>86.58(±2.21)</td>
<td>86.71(±1.31)</td>
<td>0.12</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>68.49(±1.11)</td>
<td>69.40(±1.24)</td>
<td>0.91</td>
</tr>
<tr>
<td>Vote</td>
<td>92.42(±1.63)</td>
<td>94.59(±0.84)</td>
<td>2.17</td>
</tr>
</tbody>
</table>

in the results, but it is less than using Ensemble-Co-Training with the criterion. The third column (“ECT Majority Voting”) of Table 4.6 shows the results of using the criterion and the second column (“ECT”) of Table 4.7 gives the results without employing the criterion. The presented method not only improves the accuracy, but also it reduces the complexity of the training process. Another observation in this experiment is that the Ensemble-Co-Training without the criterion also works better than the Democratic co-learning, because it benefits from the co-training approach for the training process.

Finally, we compare the performance of all methods that we discussed in this chapter. Figure 4.4 shows the results of the comparisons. We observe that Ensemble-Co-Training with optimized setting gives the best results in all datasets in our experiments.
Table 4.7: Classification Accuracy of Ensemble-Co-training without using the Criterion and Democratic Co-learning

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ECT</th>
<th>Democratic Co-Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bupa</td>
<td>58.74</td>
<td>57.05</td>
</tr>
<tr>
<td>Colic</td>
<td>82.36</td>
<td>80.60</td>
</tr>
<tr>
<td>Diabetes</td>
<td>66.27</td>
<td>66.05</td>
</tr>
<tr>
<td>Heart</td>
<td>76.70</td>
<td>76.30</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>80.37</td>
<td>79.60</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>82.99</td>
<td>84.02</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>67.52</td>
<td>67.40</td>
</tr>
<tr>
<td>Vote</td>
<td>94.79</td>
<td>94.48</td>
</tr>
</tbody>
</table>

Figure 4.4: The Performance Comparisons of all methods

4.6 Conclusion and Discussion

We propose a method that uses an ensemble of classifiers for co-training rather than feature subsets. The ensemble is used to estimate the probability of incorrect labeling and this is used with a theorem by Angluin and Laird [1] to derive a measure for deciding if the adding of a set of unlabeled data will reduce the error of a component classifier or not. Our method does not require a time consuming test for selecting a subset of unlabeled data. Experiments show that in most cases our method outperforms similar methods.

Balcan and Blum [2] present a general analysis of semi-supervised learning
4.6. Conclusion and Discussion

with discriminative classifiers (that do not try to model the distribution of the data). They point out that an assumption is required on the relation between the distribution of the data and of the classes. Without such an assumption semi-supervised learning is not possible. Our method is implicitly based on the assumption that the learning algorithms that construct the individual hypotheses bring the relevant learning biases to bear on the data and that their agreement is a good measure for the predictability of the data. The experiments show that the learning algorithms that were included together give good results in a range of application domains that vary substantially in many dimensions. For the future work, it would be interesting to make a further comparison with Disagreement Boosting methods.