SemiBoost: Boosting for Semi-supervised Learning

Pavan K. Mallapragada PAVANM@CSE.MSU.EDU
Rong Jin RONGJIN@CSE.MSU.EDU
Anil K. Jain JAIN@CSE.MSU.EDU
Yi Liu LIUYI3@CSE.MSU.EDU
Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824.

Abstract

Semi-supervised learning has attracted a significant amount of attention in machine learning. Most previous studies have focused on designing special algorithms to effectively exploit the unlabeled data. Our goal is to improve the classification accuracy of *any* given supervised learning algorithm by using the available unlabeled examples. This problem is particularly important when we need to train a hand-crafted, supervised learning algorithm with a limited number of labeled examples and a multitude of unlabeled examples. We present a *boosting* framework for semi-supervised learning, termed as **SemiBoost**. Our empirical study on 21 different datasets demonstrates that the proposed framework is effective for improving the performance of several supervised learning algorithms given a large number of unlabeled examples. We also show that our algorithm, SemiBoost often outperforms state-of-the-art semi-supervised learning algorithms.

1. Introduction

Semi-supervised learning has received significant interest in machine learning. The key idea of semi-supervised learning is to exploit both labeled and unlabeled data for learning a classification model. A number of algorithms have been proposed for semi-supervised learning, such as graphical models (Nigam et al., 2000), graph-based approaches (Zhou et al., 2005), Gaussian process (Lawrence & Jordan, 2005), kernel learning (Zhu et al., 2005), transductive Support Vector Machine (Joachims, 1999), and manifold regularization (Belkin et al., 2004). One of the key assumptions in many semi-supervised learning algorithms is the *clustering assumption* (Chapelle et al., 2006), which requires the decision boundary to pass through the region where the density of the unlabeled data is low. This assumption allows the unlabeled data to regularize the decision boundary, which in turn influences the choice of classification models.

Most semi-supervised learning algorithms design specialized learning algorithms to effectively utilize both labeled and unlabeled data. However, it is often the case that a user already has a favorite supervised learning algorithm, and would like to improve its performance by utilizing the available unlabeled data. To solve this problem, we need a general framework that can be applied to improve the performance of *any* given supervised learning algorithm by utilizing the unlabeled examples. We refer to this problem as **Semi-supervised Improvement** to differentiate our work from the standard semi-supervised learning problems.

To address the semi-supervised improvement, we propose a boosting framework, termed **SemiBoost**, for improving a given supervised learning algorithm with unlabeled data. Similar to most boosting algorithms (Freund & Schapire, 1996), SemiBoost improves the classification accuracy iteratively. At each iteration, a number of unlabeled examples will be selected and used to train a new classification model using the given supervised learning algorithm. The trained classification models from each iteration are combined to form a final classification model. The key difficulties in designing SemiBoost are: (1) how to sample the unlabeled examples for training a new classification model at each iteration?, and (2) what class labels should be assigned to the selected unlabeled examples? It is important to note that unlike supervised boosting algorithms where we select labeled examples that are difficult to classify, SemiBoost needs to select

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

unlabeled examples.

One way to address the above questions is to exploit the clustering assumption and the large margin criterion. One can improve the classification margin by selecting the examples with the highest classification confidence, and assign them with the class labels that are predicted by the current classifier. This strategy was adopted by in MarginBoost (d'Alche Buc et al., 2002). However, a problem with this strategy is that the introduction of examples with predicted class labels may only help to increase the classification margin, without actually providing any novel information to the classifier.

We propose to use the pairwise similarity measurements to guide the selection of unlabeled examples at each iteration, as well as for assigning class labels to them. For each unlabeled example $\mathbf{x}_i$, we compute

- Consistency p_i between $\mathbf{x}_i$ and examples in its local neighborhood labeled as $+1$ if the class label assigned to $\mathbf{x}_i$ is $+1$, and
- Consistency q_i between $\mathbf{x}_i$ and examples in its local neighborhood labeled as -1 if the assigned class label to $\mathbf{x}_i$ is -1.

These consistencies are computed based on pairwise similarities and are used to decide which examples should be selected and which class label should be assigned to each selected example.

2. Related Work

A large number of approaches have been developed for semi-supervised learning which can be grouped into following three categories.

2.1. Graph Based Approaches

Graph-based approaches represent both the labeled and the unlabeled examples by a connected graph, in which each example is represented by a vertex, and pairs of vertices are connected by an edge if the corresponding examples have large similaritiy. The optimal class labels for the unlabeled examples are found by minimizing their inconsistency with both the supervised class labels and the graph structure. The well known approaches in this category include harmonic function based approach (Zhu et al., 2003), spectral graph transducer (SGT), Gaussian process based approach (Lawrence & Jordan, 2005), manifold regularization (Belkin et al., 2004) and label propagation approach (Bengio et al., 2006). The proposed framework is closely related to the graph-based approaches in the sense that both utilize the example similarities for semi-supervised learning. However, unlike most graph-based approaches that are non-parametric and do not build specific classification models, we create a specific classification model by learning from both the labeled and the unlabeled examples. This is particularly important for semi-supervised improvement, whose goal is to improve a given supervised learning algorithm with massive amounts of unlabeled data.

2.2. Clustering Assumption and the Maximum Margin Criterion

These approaches utilize the unlabeled data to regularize the decision boundary. In particular, the decision boundary that passes through the region with low density of unlabeled examples is preferred to the one that is heavily populated with unlabeled examples. Approaches in this category include transductive support vector machine (TSVM) (Joachims, 1999), null category approach (Lawrence & Jordan, 2005), and MarginBoost algorithm (d'Alche Buc et al., 2002).

2.3. Kernel Learning Methods

Kernel learning methods compute the optimal kernel matrices using both the labeled and the unlabeled examples. Cluster kernel (Chapelle et al., 2002), kernel alignment method (Cristianini et al., 2001), Semi-definite Programming approach (Lanckriet et al., 2004), graph kernel approach (Zhu et al., 2005) take this approach.

3. Semi-supervised Boosting

We first describe the semi-supervised improvement problem, and then present the SemiBoost algorithm

3.1. Semi-supervised Improvement

Let $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ denote the entire dataset, including both the labeled and the unlabeled examples. Suppose that the first n_l examples are labeled given by $\mathbf{y}_l = (y_1^l, y_2^l, \ldots, y_{n_l}^l)$ where each class label y_i^l is either $+1$ or -1. We denote by $\mathbf{y}_u = (y_1^u, y_2^u, \ldots, y_{n_u}^u)$ the class labels (assigned) of unlabeled examples, where $n_u = n - n_l$. Let the labels for the entire dataset be denoted as $\mathbf{y} = [\mathbf{y}_l; \mathbf{y}_u]$. Let $S = [S_{i,j}]_{n \times n}$ denote the symmetric similarity matrix, where $S_{i,j} \geq 0$ represents the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. Let $\mathcal{A}$ denote the given supervised learning algorithm. The goal of semi-supervised improvement is to improve the performance of $\mathcal{A}$ using the unlabeled examples and the pairwise similarity S.

3.2. SemiBoost

To improve the given learning algorithm $\mathcal{A}$, we follow the idea of boosting by running the algorithm $\mathcal{A}$ iteratively. A new classification model will be learned at each iteration using the algorithm $\mathcal{A}$, and the learned classification models in different iterations will be linearly combined to form the final classification model.

3.2.1. Objective Function

Our objective function $F(\mathbf{y}, S)$ is a combination of two terms, one measuring the inconsistency between labeled and unlabeled examples $F_l(\mathbf{y}, S)$, and the other measuring the inconsistency among the unlabeled examples $F_u(\mathbf{y}_u, S)$.

Inspired by the harmonic function approach, we define $F_u(\mathbf{y}, S)$, the inconsistency between class labels $\mathbf{y}$ and the similarity measurement S, as

$$F_u(\mathbf{y}_u, S) = \sum_{i,j=1}^{n_u} S_{i,j} \cosh(y_i^u - y_j^u) \quad (1)$$

where $\cosh(y_i - y_j) = (\exp(-y_i + y_j) + \exp(y_i - y_j))/2$ is the hyperbolic cosine function. Note that $\cosh(x)$ is a convex function with its minimum at $x = 0$. Eq (1) can be expanded as $F_u(\mathbf{y}_u, S) = \frac{1}{2}\sum S_{j,i} \exp(y_j^u - y_i^u) + \frac{1}{2}\sum S_{i,j} \exp(y_i^u - y_j^u)$, and due to the symmetry of S, we have $F_u(\mathbf{y}_u, S) = \sum_{ij} S_{i,j} \exp(y_i^u - y_j^u)$.

We then define the inconsistency between labeled and unlabeled examples $F_l(\mathbf{y}, S)$ as

$$F_l(\mathbf{y}, S) = \sum_{i=1}^{n_l}\sum_{j=1}^{n_u} S_{i,j} \exp(-2y_i^l y_j^u). \quad (2)$$

Combining Eqs (1) and (2) leads to the objective function,

$$F(\mathbf{y}, S) = F_l(\mathbf{y}, S) + CF_u(\mathbf{y_u}, S). \quad (3)$$

The constant C is introduced to weight the importance between the labeled and the unlabeled data. Given the objective function in (3), the optimal class label $\mathbf{y}_u$ is found by minimizing F. This is a convex optimization problem, and therefore can be solved effectively by numerical methods. However, since our goal is to improve the given learning algorithm $\mathcal{A}$ by the unlabeled data and the similarity matrix S, we present a boosting algorithm that can efficiently minimize the objective function F.

3.2.2. Algorithm

Let $h_t(\mathbf{x}) : \mathcal{X} \to \{-1, +1\}$ denote the classification model that is learned at the t-th iteration by the algorithm $\mathcal{A}$. Let $H(\mathbf{x}) : \mathcal{X} \to \mathbb{R}$ denote the classification model learned after the first T iterations. It is computed as a linear combination of the first T classification models, i.e.,

$$H(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$$

where α_t is the combination weight. At the $(T+1)$-st iteration, our goal is to find a new classifier $h(\mathbf{x})$ and the combination weight α that can efficiently minimize the objective function F. This leads to the following optimization problem:

$$\begin{aligned}\arg\min_{h(\mathbf{x}),\alpha} \quad & \sum_{i=1}^{n_l}\sum_{j=1}^{n_u} S_{i,j} \exp(-2y_i^l(H_j + \alpha h_j)) \\ & + C\sum_{i,j=1}^{n_u} S_{i,j} \exp(H_i - H_j)\exp(\alpha(h_i - h_j))\end{aligned}$$

where $H_i \equiv H(\mathbf{x}_i)$ and $h_i \equiv h(\mathbf{x}_i)$.

To simplify the computation, we construct the upper bound of the objective function. We first bound the quantity $\exp(\alpha(h_i - h_j))$ as

$$\exp(\alpha(h_i - h_j)) \le \frac{1}{2}\left(\exp(2\alpha h_i) + \exp(-2\alpha h_j)\right),$$

and then upper bound the objective function F by the following expression:

$$\begin{aligned}F \le & \sum_{i=1}^{n_l}\sum_{j=1}^{n_u} S_{i,j} \exp(-2y_i^l H_j)\exp(-2\alpha y_i^l h_j) \\ & + C\sum_{i,j=1}^{n_u} \frac{S_{i,j}}{2} \exp(H_i - H_j)\left(\exp(2\alpha h_i) + \exp(-2\alpha h_j)\right)\end{aligned}$$

We denote the above upper bound by $\overline{F}_1$, which can be rewritten as:

$$\overline{F}_1 = \sum_{i=1}^{n_u} \exp(-2\alpha h_i)p_i + \exp(2\alpha h_i)q_i \quad (4)$$

where

$$p_i = \sum_{j=1}^{n_l} S_{i,j}e^{-2H_i}\delta(y_j, 1) + \frac{C}{2}\sum_{j=1}^{n_u} S_{i,j}e^{H_j - H_i} \quad (5)$$

$$q_i = \sum_{j=1}^{n_l} S_{i,j}e^{2H_i}\delta(y_j, -1) + \frac{C}{2}\sum_{j=1}^{n_u} S_{i,j}e^{H_i - H_j} \quad (6)$$

and $\delta(x, y)$ is one when $x = y$ and zero otherwise. The quantities p_i and q_i can be interpreted as the confidence in classifying the example $\mathbf{x}_i$ into the positive class and the negative class, respectively.

- Compute the pairwise similarity $S_{i,j}$ between any two examples.
- Initialize $H(\mathbf{x}) = 0$
- For $t = 1, 2, \ldots, T$
  - Compute p_i and q_i for every example using Equations (5) and (6)
  - Compute the class label $z_i = \text{sign}(p_i - q_i)$ for each example
  - Sample example $\mathbf{x}_i$ by the weight $|p_i - q_i|$
  - Apply the algorithm $\mathcal{A}$ to train a binary classifier $h_t(\mathbf{x})$ using the sampled examples and their class labels z_i
  - Compute α_t using Equation (7)
  - Update the classification function as $H(\mathbf{x}) \leftarrow H(\mathbf{x}) + \alpha_t h_t(\mathbf{x})$

Figure 1. The SemiBoost algorithm

The upper bound in Eq (4) is not very useful for a boosting algorithm because it is difficult to compute the weights of examples directly from Eq (4). To further simplify the expression in Eq (4), we use the following inequality

$$\exp(\gamma x) \leq \exp(\gamma) + \exp(-\gamma) - 1 + \gamma x, \ \forall x \in [-1, +1],$$

which leads to an upper bound for $\overline{F}_1$:

$$\begin{aligned} \overline{F}_1 \quad \leq \quad & \sum_{i=1}^{n_u} (p_i + q_i)(\exp(2\alpha h_i) + \exp(-2\alpha h_i) - 1) \\ & - \sum_{i=1}^{n_u} 2\alpha h_i (p_i - q_i). \end{aligned}$$

We denote the upper bound in the above equation by $\overline{F}_2$. To minimize $\overline{F}_2$, the optimal class label z_i for the example $\mathbf{x}_i$ is $z_i = \text{sign}(p_i - q_i)$, and the weight for sampling example $\mathbf{x}_i$ is $|p_i - q_i|$. The optimal α that minimizes $\overline{F}_2$ is

$$\alpha = \frac{1}{4} \ln \frac{\sum_{i=1}^{n_u} p_i \delta(h_i, 1) + \sum_{i=1}^{n_u} q_i \delta(h_i, -1)}{\sum_{i=1}^{n_u} p_i \delta(h_i, -1) + \sum_{i=1}^{n_u} q_i \delta(h_i, 1).} \qquad (7)$$

Figure 1 summarizes the SemiBoost algorithm.

Similar to most boosting algorithms, we can show that the proposed semi-supervised boosting algorithm reduces the original objective function F exponentially. This result is summarized in the following Theorem.

Theorem 1 *Let $\alpha_1, ..., \alpha_t$ be the combination weights that are computed by running the SemiBoost algorithm (Figure 1). Then, the objective function at $(t+1)$st iteration, i.e., F_{t+1}, is bounded as follows:*

$$F_{t+1} \leq \left[\sum_{i=1}^{n_u} \left(\sum_{j=1}^{n_l} S_{i,j} + C \sum_{j=1}^{n_u} S_{i,j} \right) \right] \exp\left(- \sum_{i=1}^{t} \gamma_i \right)$$

where $\gamma_i = \log(\cosh(\alpha_i))$.

Due to the space limitation, we omit the proof.

The theorem justifies the relaxations made in the above derivation. At each relaxation, the "touch-point" is maintained between the objective function and the bound. Conventional derivation of boosting algorithms follows the function gradient approach (Mason et al., 1999), which can also be viewed as a relaxation approach to approximate the original objective function by a linear function.

4. Results and Discussion

4.1. Datasets

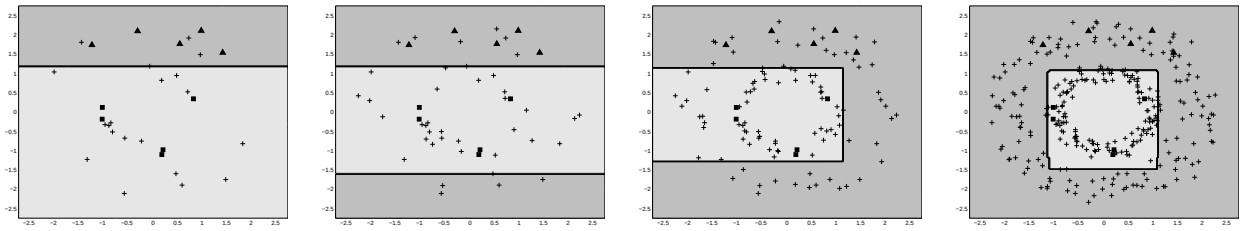
We use 8 datasets from UCI, 3 synthetic datasets, 5 datasets from Statlog (D.J. Newman & Merz, 1998), ethnicity dataset from (Jain & Lu, 2004), 3-newsgroups datasets from (Basu et al., 2004), and texture dataset (Jain & Farrokhina, 1991). A total of 21 datasets were used, which are summarized in the first three columns of Table 1 (shortened name of the dataset, number of samples, and dimensionality). Since the proposed algorithm is applicable for two-class problem, the multi-class datasets are converted into two class datasets by selecting the samples corresponding to the two most populated classes in the dataset.

We use the Radial Basis Function similarity inspired from its success in graph based approaches. For any two given samples $\mathbf{x}_i$ and $\mathbf{x}_j$, the similarity $S_{i,j}$ is computed as, $S_{i,j} = \exp(\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 / \sigma^2)$, where σ is the scale parameter controlling the spread of the radial basis function. It is well known that the choice of σ has a large impact on the performance of the algorithm (Zhu et al., 2003).

4.2. Experimental Setup

The experimental setup aims to study the following important aspects of the performance of the algorithm:

- Does SemiBoost improve the performance of any base classifier?
- How does SemiBoost algorithm compare with other semi-supervised algorithms?
- How does the SemiBoost algorithm behave with increase in the number of training samples?



(a) $n_u = 10$ (b) $n_u = 20$ (c) $n_u = 50$ (d) $n_u = 145$

Figure 2. Decision boundary obtained by SemiBoost with the increase in the number of unlabeled training samples on "two concentric rings" data, using J48 as the base classifier. There are 5 labeled training samples per class (■ and ▲). The number of unlabeled training samples (+) added per class is shown below the plot.

- How stable is SemiBoost with respect to the scale parameter σ and any imbalance in class priors?

We use % accuracy as the evaluation measure. All the experiments are run 10 times each, with different random selection of the training data. Mean and standard deviation of the % accuracy are reported. The proposed algorithm samples the data labeled from each iteration of boosting and builds a classifier $h_t(\mathbf{x})$. The number of such classifiers built will depend on the number of iterations T in boosting. T was set to 10 and we stop the boosting when weights α_t computed from Eq (7) become negative. We set the value of C in the objective function Eq (3) to be the ratio of number of labeled samples to the number of unlabeled samples $C = n_l/n_u$.

The first experiment studies the improvement in the performance of three different base classifiers ($h_t(\mathbf{x})$) after applying SemiBoost: Decision Stump (DS), the J48 decision tree algorithm (J48), the Support vector machine with the sequential minimal optimization (SVM) algorithm. We compare the improvement on DecisionStump with SemiBoost's supervised counterpart Adaboost.M1 with Decision Stump as the base classifier. Software WEKA (Witten & Frank, 2005) was used to implement all the classifiers. All the algorithms are run with their default parameters (e.g. default C and a linear kernel was used for SVM algorithm) except for Adaboost.M1 where the resampling option was selected in WEKA instead of reweighting. We randomly selected 5 samples per class for use as labeled samples and the rest of them were used as unlabeled samples. The scale parameter σ was set to 3. These four classifiers cover a wide variety of supervised learning paradigms, and all of them have been shown to be successful in literature. We also compare the performance of the proposed algorithm to that of four state-of-the-art semi-supervised learning algorithms, Self Training (Rosenberg et al., 2005) with SVM (STSVM), Low Density Separation (LDS) (Chapelle & Zien, 2005), Spectral Graph Transducer (SGT) (Joachims, 2003) and Harmonic Function approach (Zhu et al., 2003). These four algorithms are selected as they are highly scalable to large datasets that we used in our experiments. SGT was compared to TSVM (Joachims, 1999) in (Joachims, 2003) and shown to perform better overall.

We study the change in performance of the algorithm with an increase in the number of unlabeled training samples, for a fixed number of labeled training samples. We split the dataset into two halves, the training set and the induction set. To train the SemiBoost, we take 5 labeled samples per class, and add unlabeled samples in inceasing steps of 20. The resulting classifier from this training is tested on the induction set.

To evaluate the stability of the algorithm with respect to scale parameter σ, we again use 5 labeled training samples per class, SVM as the base classifier, and $\sigma \in \{0.5, 1, 2, 3, 4\}$. We also study the stability of SemiBoost with respect to an imbalance in the class prior. Keeping the priors on the labeled samples equal, we choose a fraction ρ, $0 < \rho \leq 1$, of the unlabeled samples from class 2 and all the samples from class 1. For a fixed $\sigma = 3$, SVM classifier and 5 labeled training samples per class, the performance of SemiBoost is evaluated for different values of

4.3. Results

An illustration of the decision boundary achieved using SemiBoost is shown in Figure 2 on "two concentric rings" dataset. The experimental setup uses $\sigma = 3$, and J48 as the base classifier. The plots (a)-(d) in Figure 2 show the decision boundary achieved by addition of increasing number of unlabeled samples for a fixed set of labeled training samples. As the number of unlabeled samples increases, the decision boundary obtained approaches the desired decision boundary. ρ.

Table 1 compares the supervised and semi-supervised algorithms to the SemiBoost algorithm. The columns DS, J48 and SVM are the performances of the corre-

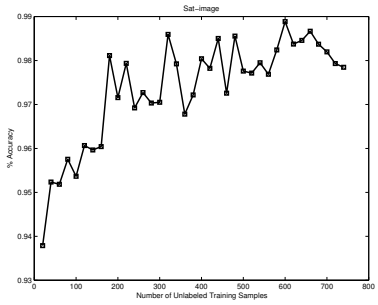


Figure 3. Performance of SemiBoost vs. number of unlabeled samples on UCI Sat-image dataset.

sponding base classifiers. The column Adaboost.M1 is the performance of the Adaboost.M1 algorithm using Decision Stump as its base classifier. The columns SB-X, where X is any one of the four base classifiers, show the performance obtained by semi-supervised improvement of the base classifier using SemiBoost. Results demonstrate that the supervised boosting algorithm Adaboost.M1 does not always improve the performance of the classifier, and in several cases it degrades the performance, especially when the base classifier is a strong classifier. However, SemiBoost significantly improves the performance of all the base classifiers. Sometimes there is a drop in the performance of AdaBoost due to overfitting, that can be alleviated by using several regularization approaches. We may argue here that broadly, the consistency requirement in unlabeled data is acting as a regularizer to the boosting process. Also, the classifier is more stable after semi-improvement as the variance in the performance drops using SemiBoost compared to the base classifiers. The best classification performance for each dataset is presented in the bold face in Table 1. SemiBoost-SVM performs best on most real datasets, and SemiBoost-J48 performs best on the synthetic datasets. The synthetic datasets were created to have a highly non-linear decision boundary, and a linear SVM is not able to separate them even with boosting.

The last four columns in Table 1 correspond to the semi-supervised algorithms HF, SGT, LDS and STSVM. Although not very common, there are cases where the base classifiers which do not use the unlabeled samples, outperform these algorithms (e.g., SVM outperforms HF, SGT and ST-SVM on the datasets satimage, image-seg, vehicle and ion). This is because SGT and HF do not always utilize the best available classifier for the given data. On the other hand, SemiBoost improves any base classifier. The poor performance of the HF algorithm observed in some cases is due to its sensitivity to σ, and due to the small number of labeled training samples. Self-training is similar to SemiBoost in the sense that it aims to improve the performance of any available classifier using unlabeled samples. However, self-training solely relies on the classifier predictions to obtain training data from unlabeled samples. A small number of training samples may result in a classifier with high variance, and any error in the learning of base classifier gets magnified over the self-training iterations. For this reason, on most of the datasets, self-training performs worse than the base classifier. Similar observations were made in (Ando & Zhang, 2005). SemiBoost, on the other hand, gives more reliable predictions as it combines the similarity information effectively with classifier predictions.

LDS is another popular approach which is shown to outperform several semi-supervised learning algorithms (Huang & Kecman, 2005). We adopted the choice of $\rho = 4$ from (Chapelle & Zien, 2005), and observed that LDS performs better than SB-SVM only on the synthetic-datasets. In most of those cases, SemiBoost using decision tree (SB-J48) outperforms it. The synthetic datasets have well-defined low-density regions between the classes making them easy for the LDS algorithm.

The performance of SemiBoost with an increase in the number of unlabeled training samples is presented in Figure 3. Due to limited space, we show the results only on the UCI sat-image classification task. Results on the remaining datasets are very similar. Figure 3 shows that for a fixed labeled training set, as more number of unlabeled samples increases, the performance on the induction set increases. This shows that the unlabeled samples are indeed helping in improving the generalization of the classifier.

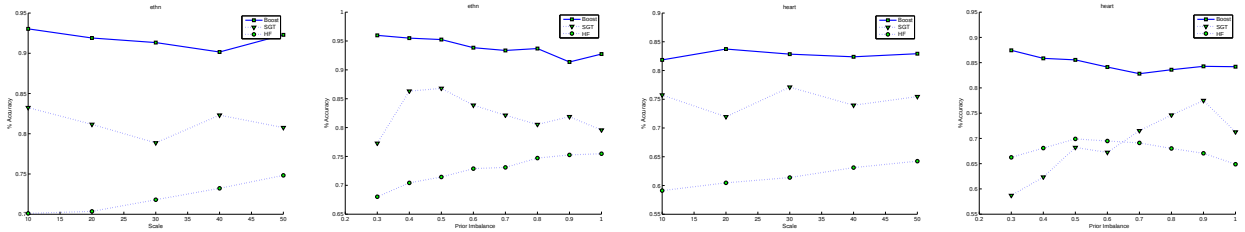
SemiBoost is relatively more stable to the scale parameter, and performs better than the HF and SGT at all scales (Figures 4(a) and 4(c)). This is a desirable property given the fact that it is a difficult problem to choose the right scale parameter. The plots in Figure 4(b) and Figure 4(d) show the sensitivity of SemiBoost to an imbalance in the prior. These figures show that SemiBoost is more stable compared to the baseline algorithms.

5. Conclusions and Future Work

We have proposed an algorithm for semi-supervised learning using a boosting framework. The strength of SemiBoost lies in its abilitiy to improve the performance of any given base classifier. Overall, the results demonstrate the feasibility of this approach and the superior performance of SemiBoost compared to the

Table 1. Performance of different algorithms with 5 training samples per class. The algorithms chosen as base classifiers for boosting are Decision Stump (DS), Decision Tree (J48) and Support Vector Machine (SVM). For each algorithm, the SB- prefixed column shows the improvement achieved by using the SemiBoost algorithm. Harmonic Function (HF), Spectral Graph Transducer (SGT), Low Density Separation (LDS) and Self-training SVM (STSVM) approaches were used for comparison. Adaboost was used with DecisionStump as the base classifier for comparison.

Dataset	n	d	DS	AdaB	SB-DS	J48	SB-J48	SVM	SB-SVM	HF	SGT	LDS	ST-SVM
austra	690	15	60.8 (12.5)	61.1 (10.8)	79.0 (2.2)	57.6 (13.6)	81.3 (3.5)	71.2 (8.5)	**85.0** (2.3)	56.1 (0.7)	76.1 (8.0)	79.37 (6.29)	65.16 (15.19)
german	1000	24	56.2 (8.1)	53.6 (7.9)	66.2 (5.2)	56.6 (6.7)	66.3 (7.9)	59.1 (8.3)	**69.2** (7.9)	59.7 (0.5)	52.9 (4.4)	59.41 (8.67)	65.03 (5.81)
heart	270	9	72.7 (14.4)	60.1 (12.8)	82.0 (1.7)	67.2 (15.9)	**83.2** (2.2)	68.4 (6.7)	82.6 (1.3)	63.9 (2.9)	75.0 (5.4)	72.35 (11.22)	64.04 (9.39)
ion	351	10	65.1 (9.0)	69.3 (8.6)	81.1 (8.6)	64.5 (8.4)	**90.5** (3.4)	75.3 (4.6)	87.3 (3.0)	34.4 (0.9)	73.3 (7.8)	79.77 (7.50)	64.87 (2.98)
iris	100	4	93.0 (9.1)	84.2 (4.9)	**100.0** (0.0)	87.7 (10.9)	99.9 (0.4)	**100.0** (0.0)	**100.0** (0.0)	**100.0** (0.0)	**100.0** (0.0)	**100.00** (0.0)	**100.0** (0.0)
wdbc	569	14	74.2 (18.5)	58.3 (5.8)	88.8 (2.0)	85.1 (9.6)	92.0 (2.5)	75.5 (5.7)	**95.3** (0.8)	29.6 (2.4)	86.9 (3.5)	88.30 (9.79)	68.75 (10.22)
vehicle	435	16	59.6 (9.4)	66.3 (06.6)	76.9 (6.9)	62.6 (5.0)	91.9 (5.0)	80.6 (6.5)	**95.1** (1.5)	42.8 (3.5)	77.6 (7.1)	72.66 (14.97)	78.38 (10.62)
image	660	18	94.3 (7.3)	74.9 (9.2)	**100.0** (11.8)	90.6 (0.0)	99.5 (0.2)	**100.0** (0.1)	**100.0** (0.0)	98.3 (0.0)	95.1 (4.2)	**100.0** (0.00)	**100.00** (0.00)
derm	184	33	97.1 (1.8)	80.0 (12.1)	99.9 (0.4)	97.4 (3.2)	99.9 (0.4)	**100.0** (0.0)	**100.0** (0.0)	98.0 (0.4)	58.8 (9.6)	**100.0** (0.00)	**100.00** (0.00)
isolet	600	51	72.9 (15.9)	60.3 (4.2)	93.2 (0.5)	66.2 (16.3)	92.9 (1.3)	90.8 (3.7)	**99.3** (0.4)	55.1 (1.4)	98.4 (0.8)	85.90 (20.40)	90.98 (12.84)
mfeat	400	76	92.5 (2.3)	91.5 (2.4)	98.3 (0.5)	86.6 (14.9)	97.9 (1.7)	97.5 (2.9)	**100.0** (0.0)	97.4 (0.2)	**100.0** (0.0)	**100.0** (0.00)	99.23 (1.39)
optdig	1143	42	72.5 (13.4)	58.8 (6.5)	91.7 (1.5)	74.1 (14.7)	96.6 (0.9)	87.8 (2.3)	99.6 (0.1)	53.2 (2.2)	**99.7** (0.0)	94.70 (15.81)	88.94 (12.90)
sat	3041	36	79.3 (1.6)	86.1 (10.2)	89.7 (5.1)	82.0 (1.8)	95.8 (8.4)	99.2 (0.5)	**99.7** (0.1)	34.4 (1.9)	63.9 (5.9)	94.09 (10.72)	99.83 (0.06)
texture	2026	19	91.6 (9.7)	85.1 (7.9)	99.3 (0.2)	88.7 (9.6)	99.3 (0.6)	98.8 (1.0)	**100.0** (0.0)	89.3 (2.8)	**100.0** (0.0)	**100.0** (0.00)	99.73 (0.36)
ethn	2630	30	62.9 (9.2)	71.5 (3.7)	78.7 (3.0)	61.8 (9.3)	86.4 (2.4)	68.2 (6.9)	**92.7** (3.2)	75.5 (0.1)	79.6 (6.0)	79.69 (11.19)	64.61 (10.88)
hf-rings	500	2	77.4 (6.1)	79.5 (08.7)	83.5 (1.2)	76.7 (7.0)	96.3 (2.9)	85.5 (2.0)	87.1 (1.1)	82.6 (2.6)	82.3 (4.0)	**97.92** (6.58)	83.00 (11.82)
rings	300	2	63.8 (2.1)	71.8 (8.1)	76.0 (9.0)	61.1 (11.1)	**98.2** (0.7)	53.7 (7.7)	60.2 (5.1)	95.8 (1.8)	48.8 (2.0)	90.62 (10.78)	49.83 (0.71)
spiral	300	2	59.0 (5.0)	65.6 (7.6)	67.3 (11.8)	62.3 (6.7)	**92.7** (2.7)	54.4 (5.0)	58.8 (5.9)	73.7 (4.0)	49.0 (1.9)	85.93 (10.50)	50.28 (4.62)
same300	199	20	68.4 (12.3)	71.1 (9.1)	75.6 (13.5)	64.3 (11.4)	85.8 (6.9)	68.3 (6.5)	**87.2** (4.0)	61.6 (1.8)	79.4 (8.1)	71.96 (11.07)	59.84 (11.62)
sim300	195	20	57.7 (5.9)	61.1 (6.8)	68.1 (3.3)	58.9 (5.4)	**76.6** (4.1)	62.3 (6.7)	73.0 (7.5)	45.8 (1.8)	66.4 (9.2)	53.89 (4.05)	57.19 (7.89)
diff300	200	20	91.1 (5.3)	82.9 (11.4)	**95.8** (0.4)	81.0 (10.9)	95.4 (2.2)	72.4 (6.3)	89.7 (5.5)	53.7 (3.1)	93.6 (6.2)	82.42 (6.66)	66.68 (9.42)



(a) ethnicity-Scale (b) ethnicity-ρ (c) heart-Scale (d) heart-ρ

Figure 4. Performance of SemiBoost with change in values of parameters scale and ρ.

state-of-the-art semi-supervised learning algorithms. The observed stability of SemiBoost suggests that it can be quite useful in practice. One of the main limitations of SemiBoost is that it is a two-class algorithm. We are exploring the multiclass extension by redefining the consistency measures to handle multiple classes.

Acknowledgements

The research was partially supported by ONR grant no. N000140710225 and NSF grant no. IIS-0643494.

References

Ando, R. K., & Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 1817–1853.

Basu, S., Bilenko, M., & Mooney, R. J. (2004). A probabilistic framework for semi-supervised clustering. *Proc. of KDD* (pp. 59–68).

Belkin, M., Niyogi, P., & Sindhwani, V. (2004). *Manifold regularization: a geometric framework for learning from examples* . Technical Report (TR-2004-06). University of Chicago, Dept of Computer Science.

Bengio, Y., Alleau, O. B., & Le Roux, N. (2006). Label propagation and quadratic criterion. In O. Chapelle, B. Schölkopf and A. Zien (Eds.), *Semi-supervised Learning*, 193–216. MIT Press.

Chapelle, O., Scholkopf, B., & Zien, A. (Eds.). (2006). *Semi-supervised Learning*. MIT Press, Cambridge, MA.

Chapelle, O., Weston, J., & Scholkopf, B. (2002). Cluster kernels for semi-supervised learning. *Advances in NIPS 15* (pp. 585–592).

Chapelle, O., & Zien, A. (2005). Semi-supervised classification by low density separation. *Proc. of 10th AISTATS* (pp. 57–64).

Cristianini, N., Shawe-Taylor, J., Elisseeff, A., & Kandola, J. S. (2001). On kernel-target alignment. *Advances in NIPS 14* (pp. 367–373).

d'Alche Buc, F., Grandvalet, Y., & Ambroise, C. (2002). Semi-supervised marginboost. *Advances in NIPS 14* (pp. 553–560).

D.J. Newman, S. Hettich, C. B., & Merz, C. (1998). UCI repository of machine learning databases.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proc. of 13th ICML* (pp. 148–156).

Huang, T. M., & Kecman, V. (2005). Performance comparisons of semi-supervised learning algorithms. *Proc. Workshop on Learning with Partially Classified Training Data, at the 22nd ICML* (pp. 45–49).

Jain, A., & Lu, X. (2004). Ethnicity identification from face images. *Proc. SPIE., Defense and Security Symposium* (pp. 114–123).

Jain, A. K., & Farrokhina, F. (1991). Unsupervised texture segmentation using gabor filters. *Pattern Recognition*, *24*, 1167–1186.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. *Proc. 16th ICML* (pp. 200–209).

Joachims, T. (2003). Transductive learning via spectral graph partitioning. *Proc. 20th ICML* (pp. 290–297).

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. I. (2004). Learning the kernel matrix with semidefinite programming. *JMLR*, *5*, 27–72.

Lawrence, N. D., & Jordan, M. I. (2005). Semi-supervised learning via gaussian processes. *Advances in NIPS 17* (pp. 753–760).

Mason, L., Baxter, J., Bartlett, P., & Frean, M. (1999). *Boosting algorithms as gradient descent in function space* (Technical Report). RSISE, Australian National University.

Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning*, *39*, 103–134.

Rosenberg, C., Hebert, M., & Schneiderman, H. (2005). Semi-supervised self-training of object detection models. *Proc. 7th WACV* (pp. 29–36).

Witten, I. H., & Frank, E. (2005). *Data mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco. 2nd edition.

Zhou, D., Huang, J., & Scholkopf, B. (2005). Learning from labeled and unlabeled data on a directed graph. *Proc. 22nd ICML* (pp. 1036–1043).

Zhu, X., Gharahmani, Z., & Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. *Proc. 20th ICML* (pp. 912–919).

Zhu, X., Kandola, J., Ghahramani, Z., & Lafferty, J. (2005). Nonparametric transforms of graph kernels for semi-supervised learning. *Advances in NIPS17* (pp. 1641–1648).