An AdaBoost algorithm for multiclass semi-supervised learning

Tanha, J.; van Someren, M.; Afsarmanesh, H.

DOI
10.1109/ICDM.2012.119

Publication date
2012

Document Version
Submitted manuscript

Published in
12th IEEE International Conference on Data Mining: ICDM 2012, 10-13 December 2012, Brussels, Belgium: proceedings

An AdaBoost Algorithm for Multiclass Semi-Supervised Learning

Jafar Tanha, Maarten van Someren, Hamideh Afsarmanesh
Informatics Institute, University of Amsterdam
Science Park 904, 1098 XH Amsterdam, Netherlands
Email: {*j.tanha, m.w.vanSomeren, h.afsarmanesh*}*@uva.nl*

***Abstract*—We present an algorithm for multiclass Semi-Supervised learning which is learning from a limited amount of labeled data and plenty of unlabeled data. Existing semi-supervised algorithms use approaches such as one-versus-all to convert the multiclass problem to several binary classification problems which is not optimal. We propose a multiclass semi-supervised boosting algorithm that solves multiclass classification problems directly. The algorithm is based on a novel multiclass loss function consisting of the margin cost on labeled data and two regularization terms on labeled and unlabeled data. Experimental results on a number of UCI datasets show that the proposed algorithm performs better than the state-of-the-art boosting algorithms for multiclass semi-supervised learning.**

***Keywords*-Semi-Supervised Learning; boosting; multiclass classification**

I. Introduction

In many applications of Machine Learning, for example in classification of images, texts, and multimedia, it is difficult to obtain values of the target variable, while unlabeled examples are plentiful. Semi-supervised learning algorithms use not only the labeled data but also the unlabeled data to build a classifier. One approach to semi-supervised learning is to extend a boosting algorithm, which is one of the most successful meta-classifiers for supervised learning [4]. Boosting uses a base learner to construct a series of weak classifiers. Each classifier is assigned a weight and when a termination criterion is reached, then the final hypothesis is constructed as a weighted combination of the classifiers from the boosting. In supervised learning, boosting uses the prediction error to determine the weights of training data and the weights for the classifiers.

Recently, boosting has been extended to semi-supervised learning. Examples are ASSEMBLE [1] and MarginBoost [3]. They use the pseudo-margin for unlabeled data in order to improving the classification performance using unlabeled examples. ASSEMBLE and MarginBoost rely only on the classifier predictions to assign the "pseudo-labels" to the unlabeled data, which is not always optimal [7]. An advantage of this approach is that it can easily be applied to multiclass classification, see [9]. A different idea is to not estimate the pseudo-margin from the predictions of the base learner. Instead, beside the margin on the labeled data, also the "consistency" is maximized. Consistency is a form of smoothness. Class labels are consistent if they are equal for data that are similar. For this the calculation of the weights for data and for classifiers in the boosting algorithm must be modified. This is done in SemiBoost [7] and RegBoost [2]. Experiments show that this idea is more effective than the approach based on pseudo-margin or confidence, see [2] and [7]. However, the solutions for the weights of data and classifiers in the boosting algorithm only exist for binary classification problems. For a multiclass semi-supervised learning problem, the only available algorithms, beside supervised learning of the labeled data only, are now those based on pseudo-margin or the use of a binary method that minimizes error on the labeled data and consistency over the unlabeled data. This must then be used in a one-versus-all or similar meta-algorithm to handle multiclass classification problems. This approach can have various problems, such as imbalanced class frequencies, increased complexity, no guarantee to obtain an optimal joint classifier, and different scales for the outputs of generated binary classifiers which complicates combining them, see [6] and [8]. Recently, in [11] a new boosting method is used for semi-supervised multiclass learning which uses similarity between predictions and data. It maps labels to n-dimensional space but this mapping may not lead to train a classifier that minimizes the margin cost properly.

In this paper we present a boosting algorithm for multiclass semi-supervised learning which minimizes both the empirical error on the labeled data and the inconsistency over labeled and unlabeled data, named Multi-SemiAdaBoost. This generalizes the SemiBoost and RegBoost algorithms from binary to multiclass classification using a coding scheme. Our proposed method uses the margin on the labeled data, the similarity among labeled and unlabeled data, and the similarity among unlabeled data in the loss function. We give a formal definition of this loss function and derive functions for the weights of classifiers and unlabeled data by minimizing an upper bound on the objective function. We then compare the performance of the algorithm to the state-of-the-art algorithms. The results show that our algorithm gives the best results.

This paper is organized as follows: section II formalizes the setting and the loss function, section III derives the weights for the boosting algorithm, sections IV and V present the experiments and the results and section VI draws the main conclusions.

II. Multiclass Semi-Supervised Learning

In this section, we define the multiclass setting and a loss function for multiclass semi-supervised learning.

A. Multiclass setting

In multiclass semi-supervised learning for the labeled points X_l= $(x_1, x_2, ..., x_l)$ labels $\{1, ..., K\}$ are provided, and for the unlabeled points $X_u = (x_{l+1}, x_{l+2}, ..., x_{l+u})$, the labels are not known. In the multiclass setting, let $Y_i \in R^k$ define a K-dimensional vector with all entries equal to $-\frac{1}{K-1}$ except a 1 in position i. Each class label $i \in \{1, ..., K\}$ is then mapped into a vector Y_i. Each entry $y_{i,j}$ is thus defined as follows:

$$y_{i,j} = \begin{cases} 1 & \text{if } i = j \\ \frac{-1}{K-1} & \text{if } i \neq j \end{cases} \tag{1}$$

where K is the number of classes and Y is the set of K-dimensional vectors such that for each $Y_i \in Y$, $\sum_{j=1}^{K} y_{i,j} = 0$. We use the above notation to formulate the loss function for multiclass semi-supervised learning. This coding was used by Zhu et. al [12] for supervised Multiclass AdaBoost algorithm, named SAMME.

For our algorithm we also need a (symmetric) similarity matrix $S = [S_{i,j}]_{n \times n}$, where $S_{i,j} = S_{j,i}$ is the similarity between the points x_i and x_j. $S^{lu} = [S_{i,j}]_{n_l \times n_u}$ denotes the similarity matrix of the labeled and unlabeled data and $S^{uu} = [S_{i,j}]_{n_u \times n_u}$ of the unlabeled data. Our algorithm is a "meta-learner" that uses a supervised learning algorithm as base learner. We assume that the labeled and unlabeled data are drawn independently from the same data distribution. In applications of semi-supervised learning normally $l \ll u$, where l is the number of labeled data and u is the number of unlabeled data.

B. Loss Function for Multiclass Semi-Supervised Boosting

The loss functions of the ASSEMBLE and MarginBoost algorithms for semi-supervised boosting are the sum of empirical loss on the labeled data and on the unlabeled data. Our approach is based on a different idea. Unlike ASSEMBLE and MarginBoost, instead of a term based on pseudo-margin for the unlabeled data, we add terms for consistency over the labeled and unlabeled data to the loss function. If examples that are similar, in terms of the similarity matrix, are assigned different classes then this adds a penalty to the loss function. This gives three components for the loss function: (1) the empirical loss on the labeled data, (2) the consistency among labeled and unlabeled data, and (3) the consistency among unlabeled data. Combining them results in a minimization problem with the following objective function:

$$F(Y, H) = C_1 F_l(Y, H) + C_2 F_{lu}(Y, S^{lu}, H) + C_3 F_{uu}(S^{uu}, H) \tag{2}$$

where C_1, C_2, and C_3 are the weights for the three components. To derive the algorithm it is mathematically convenient to use an exponential loss function for margin cost on the labeled data (as in multiclass AdaBoost [12]) which gives:

$$F_l(Y, H) = \sum_{i=1}^{n_l} exp(-\frac{1}{K} Y_i . H(x_i)) \tag{3}$$

where $H(.)$ is a multiclass ensemble classifier.

In (2), $F_{lu}(Y, S^{lu}, H)$ measures the inconsistency between the similarity information and the classifier predictions on labeled and unlabeled data as follows:

$$F_{lu}(Y, S^{lu}, H) = \sum_{i=1}^{n_l} \sum_{j=1}^{n_u} S^{lu}(x_i, x_j) exp(\frac{-1}{K-1} Y_i . H(x_j)) \tag{4}$$

where n_l and n_u are the number of labeled and unlabeled data respectively.

The third term of (2) measures the inconsistency between the unlabeled examples in terms of the similarity information and the classifier predictions. We use the harmonic function to define $F_{uu}(S^{uu}, H)$. This is a popular way to define the inconsistency in many graph-based methods, for example [13].

$$F_{uu}(S^{uu}, H) = \sum_{i,j \in n_u} S^{uu}(x_i, x_j) e^{\left(\frac{1}{K-1}(H(x_i) - H(x_j)).\overrightarrow{1}\right)} \tag{5}$$

The resulting loss function is presented as the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \mathrm{F(Y, S, H)} \\ \text{subject to} \quad & H_1(x_i) + ... + H_k(x_i) = 0, \\ & H(x_i) = Y_i, \ i = 1, 2, .., n_l \end{aligned} \tag{6}$$

The above objective function is non-linear, which makes it difficult to solve. One valuable point in that function is that it uses an exponential function and since $exp(x)$ is convex, thus it can be shown that $F(S, Y, H)$ maintains the convexity. Maintaining the convexity of the objective function requires that the kernel or similarity metrics be positive semi-definite. As mentioned before, $exp(x)$ is a convex function and the similarity function $S(x_i, x_j)$ is non-negative for each i, j. Hence, the function $F(S, Y, H)$ is a convex function.

III. The Multiclass Semi-Supervised Boosting Algorithm

In this section we derive the Multi-SemiAdaBoost algorithm. Given a set of labeled and unlabeled data, a similarity metric, and a supervised (multiclass) learning algorithm, Multi-SemiAdaBoost effectively minimizes the objective function (2). In Multi-SemiAdaBoost, a series of classifiers is constructed for the same classification task by applying a supervised base learner to varying sets of (pseudo-)labeled training data. Each classifier gets a weight.

When a termination condition is reached, the weighted combination of classifiers becomes the final hypothesis. The algorithm starts with the labeled data and in each iteration adds some unlabeled data that received a "pseudo-label" from the hypothesis constructed in the previous iteration. For this algorithm we need to find the weight for the classifiers and the confidence for the predictions. For this, let $H^t(x) : X \longrightarrow R^k$ define the linear combination of classification models after t iterations. At the t-th iteration, $H^t(x)$ is computed as:

$$H^t(x) = H^{t-1}(x) + \beta^t h^t(x) \tag{7}$$

where $\beta^t \in R$ is the weight of the base classifier $h^t(x)$, and $h^t(x)$ is determined by $P(x)$, which is a multiclass base classifier. Let also $h^t(x)$ denote a classifier as follows:

$$\forall x \in R^P \quad h^t(x) : X \to Y \tag{8}$$

Two main approaches to solve the optimization problem (6) are: gradient descent [2] and bound optimization [7]. A difficulty of the gradient descent method is to determine the step size at each iteration. We use the bound optimization approach to derive the boosting algorithm for (6), which automatically determines the step size at each iteration of boosting process.

At t-th iteration the optimization problem (2) is derived by substituting (7) in (6):

$$\begin{aligned} F = {} & C_1 \sum_{i=1}^{n_l} exp(-\frac{1}{K} Y_i.\big(H^{t-1}(x_i) + \beta^t h^t(x_i)\big) \\ & + C_2 \sum_{i=1}^{n_l} \sum_{j=1}^{n_u} S^{lu}(x_i, x_j) e^{-\frac{1}{K-1} Y_i.\big(H^{t-1}(x_j) + \beta^t h^t(x_j)\big)} \\ & + C_3 \sum_{i,j \in n_u} S^{uu}(x_i, x_j) \\ & e^{\frac{1}{K-1} \sum_{k \in l} \big((H^{t-1}(x_i) + \beta^t h^t(x_i)) - (H^{t-1}(x_j) + \beta^t h^t(x_j))\big).e_k} \end{aligned} \tag{9}$$

To solve (9), we first simplify it and then with reformulation we derive a criterion to assign weights to data, which are in propositions 1-2. Finally, we use proposition 3 to find optimal class labels and β as weight for the new classifier $h^t(x)$. For this, we use a standard basis vector e_k to represent class membership. The resulting reformulation is in preposition 1.

Proposition 1: Minimizing (9) is equivalent to minimizing the following:

$$\begin{aligned} F_1 = {} & C_1 \sum_{i=1}^{n_l} exp\big(-\frac{1}{K} Y_i.(H_i^{t-1} + \beta^t h_i^t)\big) + C_2 \sum_{i=1}^{n_l} \sum_{j=1}^{n_u} \sum_{k \in l} \\ & S^{lu}(x_i, x_j) exp(\frac{-1}{K-1} H_j^{t-1}.e_k) exp(\frac{-\beta^t}{K-1} h_j^t.e_k) \delta(Y_i, e_k) \\ & + C_3 \sum_{i,j \in n_u} \sum_{k \in l} S^{uu}(x_i, x_j) exp\Big(\frac{1}{K-1}(H_i^{t-1} - H_j^{t-1}).e_k\Big) \\ & exp\Big(\frac{\beta^t}{K-1}(h_i^t - h_j^t).e_k\Big) \end{aligned} \tag{10}$$

Subject to:

$$H_1^t(x_i) + ... + H_k^t(x_i) = 0$$

$$H^t(x_i) = Y_i \quad for \quad i = 1, ..., n_l$$

where $\delta(Y_i, e_k)$ is:

$$\delta(Y_i, e_k) = \begin{cases} 1 & \text{if } i = k \\ 0 & \text{if } i \neq k \end{cases} \tag{11}$$

F_1 is an upper bound on F and minimizing F_1 will also minimize F.

Proof: To simplify the notation we write H_i^{t-1} for $H^{t-1}(x_i)$ and h_i^t for $h^t(x_i)$, see appendix. ■

For now assume that $C_1 = C_2 = C_3 = 1$. As mentioned before, (10) is a nonlinear optimization problem. To solve this problem and also to find a criterion for assigning "pseudo-labels" to the unlabeled examples we use Proposition 2.

Proposition 2: Minimizing F_1 is equivalent to minimizing the following objective function F_2:

$$F_2 = \sum_{i=1}^{n_l} W_i exp(\frac{-\beta^t}{K} Y_i.h_i^t) + \sum_{i=1}^{n_u} \sum_{k \in l} P_{i,k} exp(\frac{-\beta^t}{K-1} h_i^t.e_k) \tag{12}$$

where

$$W_i = exp(\frac{-1}{K} Y_i.H_i^{t-1}) \tag{13}$$

and

$$\begin{aligned} P_{i,k} = {} & \sum_{j=1}^{n_l} S^{lu}(x_i, x_j) e^{(\frac{-1}{K-1} H_i^{t-1}.e_k)} \delta(Y_i, e_k) \\ & + \sum_{j=1}^{n_u} S^{uu}(x_i, x_j) e^{\big(\frac{1}{K-1}(H_j^{t-1} - H_i^{t-1}).e_k\big)} e^{\frac{1}{K-1}} \end{aligned} \tag{14}$$

Proof: See appendix. ■

In the optimization problem (12), $P_{i,k}$ can be used for weighting the unlabeled examples and W_i is used for weighting labeled data. According to these criteria the algorithm decides whether to select a sample for the new training set or not. The expression in (12) is in terms of β^t and $h^t(x)$ and hence make it difficult to solve. The following proposition simplifies it by using a bound.

Proposition 3: The class label for unlabeled example x_i that minimizes F_2 is:

$$\hat{Y}_i = \arg\max_k (P_{i,k}) \tag{15}$$

and the weight for that will be $w'_i = \max |P_{i,k}|$. Meanwhile the optimal value for β that minimizes F_2 is:

$$\beta = \frac{(K-1)^2}{K}\Bigg(log(K-1) \\ + log\Big(\frac{\sum_{i\in n_u}\sum_{k\in l} P_{i,k}\delta'(h_i^t.e_k, P_i = k) + \sum_{\substack{i\in n_l \\ h_i^t = Y_i}} W_i}{\sum_{i\in n_u}\sum_{k\in l} P_{i,k}\delta'(h_i^t.e_k, P_i \neq k) + \sum_{\substack{i\in n_l \\ h_i^t \neq Y_i}} W_i}\Big)\Bigg) \tag{16}$$

where $P_i \equiv P(x_i)$ was defined in (7).

Proof: See appendix. ■

Now, let ϵ^t denote the weighted error made by the classifier h^t. Then the ϵ^t will be as follows:

$$\epsilon^t = \frac{\sum_{i\in n_u}\sum_{k\in l} P_{i,k}\delta'(h_i.e_k, P_i \neq k) + \sum_{\substack{i\in n_l \\ h_i^t \neq Y_i}} W_i}{\sum_{i\in n_u}\sum_{k\in l} P_{i,k} + \sum_{i\in n_l} W_i} \tag{17}$$

Replacing (17) in (16) leads to the following β^t:

$$\beta^t = \frac{(K-1)^2}{K}\Big(log(K-1) + log(\frac{1-\epsilon^t}{\epsilon^t})\Big) \tag{18}$$

which is a generalization of the weighting factor of Multiclass AdaBoost [12]. Our formulation can thus be seen as a generalization of AdaBoost to multiclass semi-supervised data.

A. Multi-SemiAdaBoost Algorithm

Based on the previous discussion, we can provide the details of the algorithm. In Algorithm 1, first the values of $P_{i,k}$ and W_i are computed for each unlabeled and labeled example. In order to compute the value of $P_{i,k}$, we use the similarity information among data and the classifier prediction and W_i is computed as in multiclass AdaBoost (SAMME). Then (15) is used to assign a "pseudo-label" to the unlabeled example and the value of (15) becomes the weight. Next, a set of high-confidence newly-labeled data besides the labeled data are used based on the weights for training a new classifier, called $h^t(x)$. The algorithm then uses the result of Proposition (2) to sample data which will lead to a decrease of the value of the objective function. This new set is consistent with two criteria, classifier prediction and similarity between examples. The boosting process is repeated until it reaches a stopping condition. After finishing, the final hypothesis is the weighted combination of the generated classifiers. We use $\beta^t \leq 0$ and a fix number of iterations as stopping conditions.

Algorithm 1 Multi-SemiAdaBoost

```
L, U, S, $H^0(x) = 0$; L, U: Labeled and Unlabeled data;
S: Similarity Matrix; $H^0(x)$: Ensemble of Classifiers; $t \leftarrow 1$;
while ($\beta^t > 0$) and ($t < M$) do// M is the number of iterations
    for each $x_i \in L$ do
        Compute $W_i$ for labeled example $x_i$ based on (13)
    for each $x_i \in U$ do
        - Compute $P_{i,k}$ for unlabeled example $x_i$ based on (14)
        - Assign pseudo-labels for unlabeled data based on (15)
    - Normalize the weights of labeled and unlabeled examples
    - Sample a set of high-confidence examples from data
    - Build a new classifier $h^t(x)$
    - Compute the weights and $\beta^t$ using (16)
    - Update $H^t \leftarrow H^{t-1} + \beta^t h^t$; $t \leftarrow t + 1$
end while
Output: Generate final hypothesis based on the weights and
        classifiers
```

IV. Experiments

In the experiments we compare Multi-SemiAdaBoost (MSAB) with several other algorithms. One comparison is with using the base learner on the labeled data only and a second is with (mutliclass) AdaBoost (SAMME [12]) using the same base learner. The purpose is to evaluate if MSAB exploits the information in the unlabeled data. We also include comparisons with the state-of-the-art algorithms for semi-supervised boosting, in particular ASSEMBLE, SemiMultiAdaBoost [9], RegBoost [2], and SemiBoost [7]. Like MSAB, SemiBoost and RegBoost use smoothness regularization but they are limited to binary classification. For comparison, we use the one-vs-all method to handle the multiclass classification problem with RegBoost and SemiBoost. In our experiments, we use WEKA implementation of the classifiers with default parameter settings [5].

Two main steps in Multi-SemiAdaBoost are: (i) the similarity between examples and (ii) sample from the unlabeled examples. There are different distance-based approaches to compute the similarity between the data. In this paper we employ the Radial Basis Function (RBF) which is used effectively as similarity metric in many domains. For two examples x_i and x_j the RBF similarity is computed as:

$$S(x_i, x_j) = exp(-\frac{\|x_i - x_j\|_2^2}{\sigma^2}) \tag{19}$$

where σ is the scale parameter which plays a major role in the performance of the kernel, and should be carefully tuned to the problem [13].

In the sampling process only the high-confidence data points must be selected for training. Finding the best selection is a difficult problem [10]. On one hand, selecting a small set of newly-labeled examples might lead to slow convergence, and on the other hand, selecting a large set of newly-labeled examples may include some poor examples. One possible solution for that is to use a threshold or even a fixed number which is optimized through the training process. Sampling data for labeling is based on the following criterion: for confidence

$$P_d(x_i) = \frac{\hat{Y}_i - \max\{Y_{i,k}|Y_{i,k} \neq \hat{Y}_i, \quad k = 1, ..., K\}}{\sum_{i=1}^{n_u}(\hat{Y}_i - \max\{Y_{i,k}|Y_{i,k} \neq \hat{Y}_i, \quad k = 1, ..., K\})} \tag{20}$$

where $\hat{Y}_i$ is computed from (15). $P_d(x_i)$ is viewed as the probability distribution of classes of the example x_i, which amounts to a measure of confidence. For the labeled data we use W_i in (13) as weight:

$$P_d(x_i) = \frac{W_i}{\sum_{i=1}^{n_l} W_i} \tag{21}$$

where $x_i \in L$. We use learning from weighed examples as in AdaBoost and in each iteration we select the top 15% of the unlabeled data based on the weights and add them to the training set.

A. Supervised Base Learner

As mentioned earlier, we assume that the base learner as a black box in the algorithm. It means that the proposed algorithm does not need to know the inner process of the base learner. However, the performance of the method depends on the base learner. We experiment with a Decision Tree learner (J48, the Java implementation of C4.5 decision tree classifier) and Naive Bayes as base classifiers.

B. UCI datasets

We use the UCI datasets for assessing the proposed algorithms. Recently several UCI datasets have been extensively used for evaluating semi-supervised learning methods [10], [7], and [9]. Sixteen benchmark datasets from the UCI data repository are used in our experiments. For each dataset, 30 percent of the data are kept as test set, and the rest is used as training data. Training data in each experiment are first partitioned into 90 percent unlabeled data and 10 percent labeled data, keeping the class proportions in all sets similar to the original data set. We run each experiment 10 times with different subsets of training and testing data. The results reported refer to the test set.

V. Results

Tables I and II give the results of all experiments. The first column shows the specification of datasets, such as name, number of examples and classes. The second and third columns in these Tables give the performance of supervised multiclass base classifiers (DT for Decision Trees and Naive Bayes) and multiclass Adaboost meta classifier using the classifiers as base learner. The columns ASSEMBLE, SMBoost, RegBoost, and SemiBoost show the performance of four semi-supervised boosting algorithms ASSEMBLE, SemiMultiAdaBoost, RegBoost, and SemiBoost respectively.

MSAB and Supervised Learning

Multi-SemiAdaBoost significantly improves the performance of supervised learning with different base classifiers for nearly all the datasets. Using statistical t-test, we observed that Multi-SemiAdaBoost improves the performance of J48 and Naive Bayes base classifiers on 16 out of 16 datasets. The results show that Multi-SemiAdaBoost is also better than multiclass AdaBoost meta classifier trained using only labeled data. We also observe that the classification models generated by using Multi-SemiAdaBoost are relatively more stable than J48 base classifier because of lower standard deviation in classification accuracy.

MSAB and Semi-Supervised Boosting Algorithms

Table I and II show that Multi-SemiAdaBoost gives better results than all four semi-supervised algorithms ASSEMBLE, SemiMultiAdaBoost, RegBoost, and SemiBoost on 14 (for J48) or 13 (for Naive Bayes) out of 16 datasets. The improvement of MASB in most of the used datasets are significant and it outperforms the ASSEMBlE and SMBoost on 15 out of 16 datasets, when the base classifier is J48. The same results can be seen with Naive Bayes classifier.

There are datasets where the proposed algorithms may not significantly improve the performance of the base classifiers. In these cases the supervised algorithm outperforms all the semi-supervised algorithms, for example in Tables II the AdaBoost meta classifier performs the same as the proposed methods on Car, Cmc, and $Diabetes$ datasets and outperforms the other methods. These kinds of results emphasize that the unlabeled examples do not guarantee that they always are useful and improve the performance.

VI. Conclusion and Discussion

We presented a boosting algorithm for multiclass semi-supervised learning with a novel loss function involving the empirical error on labeled data and the consistency between classifier predictions and similarity information. Specifically we derived the weights for the data and for the classifiers. The resulting algorithm is shown to perform better than existing algorithms for multiclass problems. Multi-SemiAdaBoost can use any multiclass supervised learning algorithm as base classifier.

An open problem for future work is how to efficiently find and tune a good similarity function. Another issue is the encoding for the multiclass setting. The encoding that is used here is not the only possibility and it seems interesting to exploit other encoding schemes.

References

[1] K. Bennett, A. Demiriz, and R. Maclin. Exploiting unlabeled data in ensemble methods. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 289–296. ACM, 2002.

Table I: The classification accuracy (± s.d) with 10% labeled examples and J48 as base learner

DataSets (#Samples,#Classes)	Supervised Learning		Semi-Supervised Learning				
	DT	SAMME	ASSEMBLE	SMBoost	RegBoost	SemiBoost	MSAB
Balance(625,3)	70.72±5.5	74.75±3.1	78.01±3.5	77.73±3.1	75.34±4.1	76.56±3.7	**81.90±2.0**
Car(1728,4)	76.93±1.6	78.34±2	78.97±1.6	72.88±1.3	78.40±1.3	78.59±1.0	**80.40±1.7**
Cmc(1473,3)	42.03±3.8	44.89±4.2	40.69±3.2	39.04±3.1	47.89±2.9	48.46±2.2	**49.26±2.7**
Dermatology(366,6)	73.00±4.1	76.99±5.0	87.28±4.2	87.77±3.1	87.10±3.9	86.68±3.9	**88.37±2.2**
Diabetes(768,2)	68.81±5.0	70.97±2.5	71.98±4.5	69.78±5.0	71.50±4.1	73.20±2.7	**74.96±2.6**
Ecoli(336,8)	65.88±9.9	66.82±8.5	75.54±4.0	77.41±7.7	79.43±3.8	77.41±4.6	**85.98±3.4**
Glass(214,6)	49.79±9.6	50.94±6.1	52.84±7.1	54.62±9.5	60.21±4.4	61.55±4.6	**66.38±3.4**
Iris(150,3)	71.13±8.9	71.13±8.9	77.08±6.3	88.69±8.1	91.4±6.2	93.45±5.6	**94.64±4.4**
Liver(345,2)	54.86±7.5	56.93±7.6	51.03±5.7	51.17±2.8	62.01±2.8	62.46±3.4	**63.27±4.1**
Optdigits(1409,10)	63.26±2.4	76.52±3.4	79.42±2.6	**87.71±1.4**	76.41±2.1	74.65±1.3	80.28±2.5
Sonar(208,2)	60.08±9.2	59.87±6.9	57.98±9.0	60.5±8.1	72.23±5.1	**74.42±2.3**	73.68±5.0
Soybean(686,19)	51.00±4.4	47.83±4.6	**71.37±2.5**	68.51±2.0	67.10±2.5	61.49±3.4	71.06±2.4
Vowel(990,11)	35.92±4.7	42.57±2.6	26.52±3.2	26.70±2.5	44.31±2.7	47.94±3.0	**57.02±3.0**
Wave(5000,3)	70.10±1.7	78.04±0.8	76.95±1.2	75.53±1.7	77.82±1.3	77.82±1.3	**79.15±1.1**
Wine(178,3)	67.36±9.8	67.36±9.8	91.92±4.4	93.27±5.3	94.10±4.9	94.73±4.8	**96.49±3.3**
Zoo(101,7)	66.19±5.2	56.66±8.1	85.23±4.2	85.23±4.2	87.94±5.4	87.14±5.5	**90.00±4.7**

Table II: The classification accuracy (± s.d) with 10% labeled examples and Naive Bayes as base learner

DataSets	Supervised Learning		Semi-Supervised Learning				
	Naive Bayes	SAMME	ASSEMBLE	SMBoost	RegBoost	SemiBoost	MSAB
Balance	75.88±3.5	76.71±2.5	79.77±2.0	79.98±2.5	78.24±3.5	77.36±4.1	**82.09±3.2**
Car	77.5±1.9	78.51±0.5	77.64±1.9	80.42±2.5	79.12±9.5	78.66±2.1	**80.75±1.8**
Cmc	46.98±1.4	48.05±1.3	47.35±3.1	46.94±2.2	48.2±2.1	49.09±1.4	**51.11±1.7**
Dermatology	81.52±7.4	81.52±7.4	86.94±7.5	85.25±4.5	86.44±3.3	86.44±3.9	**90±2.4**
Diabetes	73.45±1.1	73.05±1.5	73.64±1.6	75.09±2.4	72.12±3.1	74.90±1.9	**75.29±2.8**
Ecoli	81.68±3.3	78.69±6.3	81.3±1.9	76.07±8.0	80±2.7	80.74±2.8	**83.92±3.2**
Glass	50±7.0	50.21±6.7	45.58±9.9	51.26±8.7	57.01±7.0	56.51±7.1	**60.08±3.4**
Iris	81.25±5.5	80.95±6.6	87.49±3.4	91.07±5.9	87.32±4.7	90.47±6.3	**93.45±4.7**
Liver	55.39±5.5	55.04±7.2	52.74±4.1	55.22±8.8	55.06±7.0	55.75±6.3	**60.17±5.7**
Optdigits	72.04±1.9	70.57±1.7	84.6±1.0	**87.35±2.0**	80.54±3.1	77.95±2.5	83.00±2.0
Sonar	64.41±7.1	63.23±5.1	62.64±7.6	65.58±5.6	65.5±5.1	**70.94±5.8**	70.58±6.5
Soybean	70.00±1.4	68.44±7.1	**76.85±1.2**	73.07±3.9	67.77±1.1	57.77±1.1	74.3±2.9
Vowel	47.52±3.4	48.02±2.0	25.82±3.3	32.78±3.5	47.34±2.7	44.5±2.2	**52.53±3.0**
Wave	80.44±1.4	80.91±1.9	80.35±1.1	79.94±0.7	81.01±1.7	81.83±1.3	**83.43±0.6**
Wine	84.56±5.8	84.56±5.8	94.38±2.2	93.68±2.9	93.69±3.0	92.28±2.6	**95.17±2.9**
Zoo	87.77±4.0	87.77±4.0	89.44±2.5	**94.00±2.7**	89.52±3.8	89.52±3.8	91.11±3.4

[2] K. Chen and S. Wang. Semi-supervised learning via regularized boosting working on multiple semi-supervised assumptions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(1):129–143, 2011.

[3] F. dAlché Buc, Y. Grandvalet, and C. Ambroise. Semi-supervised marginboost. *Advances in neural information processing systems*, 14:553–560, 2002.

[4] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *ICML*, pages 148–156, 1996.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11:10–18, November 2009.

[6] R. Jin and J. Zhang. Multi-class learning by smoothed boosting. *Machine learning*, 67(3):207–227, 2007.

[7] P. Mallapragada, R. Jin, A. Jain, and Y. Liu. Semiboost: Boosting for semi-supervised learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(11):2000–2014, 2009.

[8] M. Saberian and N. Vasconcelos. Multiclass boosting: Theory and algorithms. *In Proc. Neural Information Processing Systems (NIPS)*, 2011.

[9] E. Song, D. Huang, G. Ma, and C. Hung. Semi-supervised multi-class adaboost by exploiting unlabeled data. *Expert Systems with Applications*, 2010.

[10] J. Tanha, M. van Someren, and H. Afsarmanesh. Disagreement-based co-training. In *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on*, pages 803–810. IEEE, 2011.

[11] H. Valizadegan, R. Jin, and A. Jain. Semi-supervised boosting for multi-class classification. *Machine Learning and Knowledge Discovery in Databases*, pages 522–537, 2008.

[12] J. Zhu, S. Rosset, H. Zou, and T. Hastie. Multi-class adaboost. *Statistics and its Interface*, 2,:349–360, 2009.

[13] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, pages 912–919, 2003.

Appendix

Proposition 1:

Proof: Let the function $Y_i.e_k$ be equal to 1, if a data point with Y_i belongs to class k. Then (9) can be rewritten as follows:

$$\begin{aligned}\min F = {} & C_1 \sum_{i=1}^{n_l} exp(-\frac{1}{K} Y_i.\big(H^{t-1}(x_i) + \beta^t h^t(x_i)\big)) \\ & + C_2 \sum_{i=1}^{n_l}\sum_{j=1}^{n_u} S^{lu}(x_i,x_j) e^{-\frac{1}{K-1}\sum_{k\in l}(Y_i.e_k)(H_j^{t-1}+\beta^t h_j^t).e_k} \\ & + C_3 \sum_{i,j\in n_u} S^{uu}(x_i,x_j) \\ & e^{\frac{1}{K-1}\sum_{k\in l}\Big((H_i^{t-1}+\beta^t h_i^t)-(H_j^{t-1}+\beta^t h_j^t)\Big).e_k}\end{aligned} \tag{22}$$

Using the following inequality in (9):

$$(c_1 c_2 ... c_n)^{\frac{1}{n}} \leq \frac{c_1 + c_2 + ... + c_n}{n} \tag{23}$$

and then replacing the value of Y_i results in the (10), where δ is defined in (11). ■

Preposition 2:

Proof: According to the formulation that we mentioned earlier, $\frac{-1}{K-1} \leq \beta^t h(x_i).e_k \leq 1$, then multiplying it with $\frac{1}{K-1}$ and using the exponential function gives the following inequality:

$$e^{\frac{-1}{(K-1)^2}} \leq e^{\frac{\beta^t}{K-1}h(x_i)} \leq e^{\frac{1}{K-1}} \tag{24}$$

Using (24) for decomposing the third term of (**??**) leads to the following expression:

$$e^{\frac{\beta^t}{K-1}(h(x_i)-h(x_j)).e_k} \leq e^{\frac{1}{K-1}} e^{\frac{-\beta^t}{K-1}h(x_j).e_k} \tag{25}$$

Replacing the inequality (25) in the last term of (**??**) gives:

$$\begin{aligned}F_1 \leq {} & \sum_{i=1}^{n_l} exp(-\frac{1}{K} Y_i.H_i^{t-1}) exp(-\frac{\beta^t}{K} Y_i.h_i^t) \\ & + \sum_{i=1}^{n_l}\sum_{j=1}^{n_u}\sum_{k\in l} S^{lu}(x_i,x_j) exp(\frac{-1}{K-1} H_j^{t-1}.e_k) \\ & exp(\frac{-\beta^t}{K-1} h_j^t.e_k)\delta(Y_i,e_k) \\ & + \sum_{i,j\in n_u}\sum_{k\in l} S^{uu}(x_i,x_j) exp\Big(\frac{1}{K-1}(H_i^{t-1} - H_j^{t-1}).e_k\Big) \\ & \Big(e^{\frac{1}{K-1}} e^{\frac{-\beta^t}{K-1}h_j^t.e_k}\Big)\end{aligned} \tag{26}$$

Factoring out the common term $e^{\frac{-\beta^t}{K-1}h(x_j).e_k}$ and re-arranging terms then results in:

$$\begin{aligned}F_1 \leq {} & \sum_{i=1}^{n_l} exp(-\frac{1}{K} Y_i.H_i^{t-1}) exp(-\frac{\beta^t}{K} Y_i.h_i^t) \\ & + \sum_{j=1}^{n_u}\sum_{k\in l} exp(\frac{-\beta^t}{K-1} h_j^t.e_k)\Big(\sum_{i=1}^{n_l} S^{lu}(x_i,x_j) \\ & exp(\frac{-1}{K-1} H_j^{t-1}.e_k)\delta(Y_i,e_k) \\ & + \sum_{i\in n_u} S^{uu}(x_i,x_j) exp(\frac{1}{K-1}(H_j^{t-1} - H_i^{t-1}).e_k) e^{(\frac{1}{K-1})}\Big)\end{aligned} \tag{27}$$

As a results, it gives:

$$F_1 \leq \sum_{i=1}^{n_l} W_i exp(\frac{-\beta^t}{K} Y_i.h_i^t) + \sum_{i=1}^{n_u}\sum_{k\in l} P_{i,k} exp\Big(\frac{-\beta^t}{K-1} h_i^t.e_k\Big) \tag{28}$$

where

$$W_i = exp(\frac{-1}{K} Y_i.H_i^{t-1}) \tag{29}$$

and

$$\begin{aligned}P_{i,k} = {} & \sum_{j=1}^{n_l} S^{lu}(x_i,x_j) e^{(\frac{-1}{K-1}H_i^{t-1}.e_k)}\delta(Y_i,e_k) \\ & + \sum_{j=1}^{n_u} S^{uu}(x_i,x_j) e^{\Big(\frac{1}{K-1}(H_j^{t-1}-H_i^{t-1}).e_k\Big)} e^{\frac{1}{K-1}}\end{aligned} \tag{30}$$

■

Preposition 3:

Proof: We use the following inequality to decompose the elements of F_2:

$$\forall x \in [-1,1] \;\; exp(\beta x) \leq exp(\beta) + exp(-\beta) + \beta x - 1, \tag{31}$$

Replacing the inequality (31) in (12) gives:

$$\begin{aligned}F_2 \leq {} & \sum_{i=1}^{n_l} W_i (e^{\frac{-\beta^t}{K}} + e^{\frac{\beta^t}{K}} - 1) - \sum_{i=1}^{n_l} W_i (\frac{\beta^t}{K} Y_i.h_i^t) \\ & + \sum_{i=1}^{n_u}\sum_{k\in l} P_{i,k} (e^{\frac{-\beta^t}{K-1}} + e^{\frac{\beta^t}{K-1}} - 1) - \sum_{i=1}^{n_u}\sum_{k\in l} P_{i,k} \frac{\beta^t}{K-1} h_i^t.e_k\end{aligned} \tag{32}$$

Then, re-arranging the above inequality gives:

$$\begin{aligned}F_2 \leq {} & \Big(\sum_{i=1}^{n_l} W_i (e^{\frac{-\beta^t}{K}} + e^{\frac{\beta^t}{K}} - 1) + \sum_{i=1}^{n_u}\sum_{k\in l} P_{i,k} (e^{\frac{-\beta^t}{K-1}} + e^{\frac{\beta^t}{K-1}} - 1)\Big) \\ & - \Big(\sum_{i=1}^{n_l} W_i (\frac{\beta^t}{K} Y_i.h_i^t) + \sum_{i=1}^{n_u}\sum_{k\in l} P_{i,k} \frac{\beta^t}{K-1} h_i^t.e_k\Big)\end{aligned} \tag{33}$$

The first term in inequality (33) is independent of h_i. Hence, to minimize F_2, finding the examples with the largest $P_{i,k}$ and W_i is sufficient at each iteration of the boosting process.

We assign the value $P_{i,k}$ of selected examples as weight for the newly-labeled examples, hence $w'_i = \max |P_{i,k}|$. To prove the second part of the proposition, we expand F_2 as follows:

$$
\begin{aligned}
F_2 &= \sum_{i=1}^{n_l} W_i exp(\frac{-\beta^t}{K} Y_i.h_i^t) + \sum_{i=1}^{n_u} \sum_{k \in l} P_{i,k} exp(\frac{-\beta^t}{K-1} h_i^t.e_k) \\
&= \sum_{\substack{i \in n_l \\ h_i^t = Y_i}} W_i exp(\frac{-\beta^t}{K-1}) + \sum_{\substack{i \in n_l \\ h_i^t \neq Y_i}} W_i exp(\frac{\beta^t}{(K-1)^2}) \\
&+ \sum_{i=1}^{n_u} \sum_{k \in l} P_{i,k} exp(\frac{-\beta^t}{K-1}) \delta'(h_i^t.e_k, P_i = k) \\
&+ \sum_{i=1}^{n_u} \sum_{k \in l} P_{i,k} exp\Big(\frac{\beta^t}{(K-1)^2}\Big) \delta'(h_i^t.e_k, P_i \neq k)
\end{aligned}
\tag{34}
$$

where δ' is defined as:

$$
\delta'(h_i^t.e_k, T) = \begin{cases} 1 & \text{if T is true} \\ 0 & \text{otherwise} \end{cases} \tag{35}
$$

Differentiating the above equation w.r.t β and equating it to zero, gives:

$$
\begin{aligned}
\frac{\partial F_2}{\partial \beta} &= \frac{-1}{K-1} exp(\frac{-\beta}{K-1}) \sum_{\substack{i \in n_l \\ h_i^t = Y_i}} W_i \\
&+ \frac{1}{(K-1)^2} exp(\frac{\beta}{(K-1)^2}) \sum_{\substack{i \in n_l \\ h_i^t \neq Y_i}} W_i \\
&+ \frac{-1}{K-1} exp(\frac{-\beta}{K-1}) \sum_{i=1}^{n_u} \sum_{k \in l} P_{i,k} \delta'(h_i^t.e_k, P_i = k)) \\
&+ \frac{1}{(K-1)^2} exp(\frac{\beta^t}{(K-1)^2}) \sum_{i=1}^{n_u} \sum_{k \in l} P_{i,k} \delta'(h_i^t.e_k, P_i \neq k)) = 0
\end{aligned}
\tag{36}
$$

Simplifying the above equation results in (16). ■