SERBoost: Semi-supervised Boosting with Expectation Regularization

Amir Saffari, Helmut Grabner, and Horst Bischof

Institute for Computer Graphics and Vision, Graz University of Technology, Austria
`saffari,hgrabner,bischof@icg.tugraz.at`

Abstract. The application of semi-supervised learning algorithms to large scale vision problems suffers from the bad scaling behavior of most methods. Based on the *Expectation Regularization* principle, in this paper we propose a novel semi-supervised boosting method, called SERBoost that can be applied to large scale vision problems and its complexity is dominated by the base learners. The algorithm provides a margin regularizer for the boosting cost function and shows a principled way of utilizing prior knowledge. We demonstrate the performance of SERBoost on the Pascal VOC2006 set and compare it to other supervised and semi-supervised methods, where SERBoost shows improvements both in terms of classification accuracy and computational speed.

1 Introduction

Semi-supervised learning addresses the problem of "How to improve the performance of an adaptive model using unlabeled data together with the labeled data?". Many supervised approaches obtain high recognition rates if enough labeled training data is available. However, for most practical problems there is simply not enough labeled data available, whereas hand-labeling is tedious and expensive, in some cases not even feasible, while a large amount of unlabeled data is available. This is especially true for applications in computer vision like object recognition and categorization.

Therefore, the central issue of semi-supervised learning is to find a way to exploit this huge amount of obscured information from unlabeled data. Due to a considerably large amount of literature in this subject and lack of space, we refrain to mention most of the well-known semi-supervised methods and encourage the interested readers to refer to a comprehensive overview of this field [1] and also to the recent book of Chapelle *et al.* [2]. However, since we are directly addressing the semi-supervised boosting methods, it should be noted that there has been a few attempts in formulating the semi-supervised learning as a boosting procedure, started by [3, 4]. Recently, based on the idea of graph-based manifold regularization methods, Mallapragada *et al.* [5] and Chen and Wang [6] have proposed other approaches for boosting models. Furthermore, in computer vision, Cohen *et al.* [7] use both labeled and unlabeled data to improve the face detectors. In [8] a semi-supervised approach for detecting objects in aerial images has been developed. Also related but only inspired by semi-supervised learning is the work of Fei-Fei *et al.* [9] which presents an incremental approach to learn object categories using Internet search as an additional information.

Recently, Mann and McCallum [10] have analyzed many semi-supervised learning algorithms and noted that despite the vast amount of literature there are not many practical applications, and they pointed out two main reasons for that: 1) many algorithms (especially those based on EM) are fragile and are heavily dependent on hyper-parameters, and 2) many algorithms are computationally expensive with a scaling behavior of $\mathcal{O}(n^3)$, where n is the number of unlabeled samples. This is counter productive since the full power of semi-supervised learning can only be obtained with a large amount of unlabeled data. Mann and McCallum proposed a method called *Expectation Regularization* on exponential-family of parametric models which does not suffer from these problems. The basic idea is to augment the label-likelihood objective function with a term that encourages the model predictions on unlabeled data to match certain expectations.

Based on this idea, we propose a novel semi-supervised boosting algorithm which has the following properties:

– It scales reasonably with respect to the number of labeled and unlabeled samples, and in fact provides the same complexity as the traditional supervised boosting methods, while being very easy to implement.
– It naturally provides a margin regularizers for the boosting algorithm [11] which has relations to the principles of maximum entropy learning [12].
– It provides a principled way of incorporating prior knowledge, *e.g.*, [13], into the learning process of the semi-supervised boosting model.
– It is robust with respect to the variations of its hyper-parameter.
– It is a generalization of the GentleBoost [14] algorithm to the semi-supervised domain.
– On Pascal VOC2006 datasets, it outperforms the Linear SVM, TSVM [15], Random Forests [16], and GentleBoost [14] by a large margin and gives comparable results to the χ^2-SVM [17–19] classifier while being considerably faster.

This paper is organized as follows: we first derive the novel boosting formulation based on the idea of expectation regularization in Section 2. In Section 3, we demonstrate on the performance of our model and compare it to a few other supervised and semi-supervised methods, while Section 4 provides a conclusion and points out the future work.

2 Boosting with Expectation Regularization

We address the problem of semi-supervised binary classification. Assume that we are given a *prior* conditional probability in the form of $P_p(y|\mathbf{x})$ where $y \in \{-1, 1\}$ is the binary class label and $\mathbf{x} \in \mathbb{R}^D$ is a sample instance. This prior knowledge expresses our *belief* regarding the conditional distribution of the labels given the input features. This prior knowledge can be obtained in different ways: it can be only the label priors $P_p(y)$ [10], as it will be shown later in this paper, it can be as weak as a maximum entropy prior $\forall \mathbf{x}, y : P_p(y|\mathbf{x}) = 0.5$, or it can be the output of another learning method. The later case is very interesting and important in practice as it provides solutions for knowledge transfer and incorporation of prior knowledge scenarios.

Given a set of labeled, $\mathcal{X}_L$, and unlabeled, $\mathcal{X}_U$, samples as:

$$\begin{aligned} \mathcal{X}_L =& \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_{N_L}, y_{N_L})\}, \mathbf{x}_i \in \mathbb{R}^D, y_i \in \{-1, 1\} \\ \mathcal{X}_U =& \{\mathbf{x}_1, \dots, \mathbf{x}_{N_U}\}, \mathbf{x}_i \in \mathbb{R}^D \end{aligned} \tag{1}$$

we denote $\mathcal{X} = \mathcal{X}_L \cup \mathcal{X}_U$ as the overall collection of data samples. Also let the $P_p(y|\mathbf{x})$ be the prior probability and the $\hat{P}(y|\mathbf{x})$ be the estimated probability by the learning model. The goal is to use boosting [20, 14] to learn an additive model $F(\mathbf{x}) = \sum_{t=1}^{T} f_t(\mathbf{x})$ in a way that its classification accuracy is as high as possible while its probabilistic predictions over the unlabeled samples resembles the given prior.

2.1 Loss Function

We define a loss function for the learning process which contains two components corresponding to the labeled and unlabeled data as:

$$\mathcal{L}(F(\mathbf{x}), \mathcal{X}) = \mathcal{L}_L(F(\mathbf{x}), \mathcal{X}_L) + \alpha \mathcal{L}_U(F(\mathbf{x}), \mathcal{X}_U) \tag{2}$$

where $\mathcal{L}_L$ and $\mathcal{L}_U$ are the loss functions for the labeled and unlabeled samples, respectively, and $\alpha \geq 0$ defines the contribution rate of the unlabeled loss.

Loss for the Labeled Samples Since we are trying to formulate our model as a boosting method, we use the traditional exponential loss function for the labeled data samples:

$$\mathcal{L}_L(F(\mathbf{x}), \mathcal{X}_L) = \mathbb{E}(e^{-yF(\mathbf{x})}) = \sum_{\mathbf{x} \in \mathcal{X}_L} e^{-yF(\mathbf{x})}. \tag{3}$$

Note that we drop the scaling factors, *e.g.* $\frac{1}{N_L}$, in calculating the expectations as these parameters can be easily integrated into α. Since the boosting algorithms are designed to minimize the exponential loss, it is known [14] that the minimizer of Eq.(3) is:

$$F(\mathbf{x}) = \frac{1}{2} \log \frac{\hat{P}(y = 1|\mathbf{x})}{\hat{P}(y = -1|\mathbf{x})}. \tag{4}$$

Therefore, we can see that:

$$\begin{aligned} P^+(\mathbf{x}) =& \hat{P}(y = 1|\mathbf{x}) = \frac{e^{F(\mathbf{x})}}{e^{F(\mathbf{x})} + e^{-F(\mathbf{x})}}, \\ P^-(\mathbf{x}) =& \hat{P}(y = -1|\mathbf{x}) = 1 - \hat{P}(y = 1|\mathbf{x}) = \frac{e^{-F(\mathbf{x})}}{e^{F(\mathbf{x})} + e^{-F(\mathbf{x})}}. \end{aligned} \tag{5}$$

Note that for the notation brevity, we use the super-scripts symbols to refer to the class labels in conditional probabilities.

Loss for the Unabeled Samples It is natural to define the unlabeled cost function as the Kullback-Leibler (KL) divergence between the prior probability and the optimized model [10]:

$$\mathcal{L}_U(F(\mathbf{x}), \mathcal{X}_U) = \mathbb{E}(D(P_p \| \hat{P})) \tag{6}$$

where:

$$\begin{aligned} D(P_p \| \hat{P}) &= \sum_{y \in \{-1,1\}} P_p(y|\mathbf{x}) \log \frac{P_p(y|\mathbf{x})}{\hat{P}(y|\mathbf{x})} = \\ &= \sum_{y \in \{-1,1\}} P_p(y|\mathbf{x}) \log P_p(y|\mathbf{x}) - \sum_{y \in \{-1,1\}} P_p(y|\mathbf{x}) \log \hat{P}(y|\mathbf{x}) = \\ &= -H(P_p) + H(P_p, \hat{P}) \end{aligned} \tag{7}$$

is the KL-divergence. $H(P_p, \hat{P})$ is the cross entropy between the target and calculated model and $H(P_p)$ is the entropy of the target distribution. Since $H(P_p)$ is a constant and does not depend on the optimized model, we can simply drop it. Furthermore, since we are dealing with a binary classification problem, by using Eq.(5) we can write:

$$\begin{aligned} H(P_p, \hat{P}) &= -\sum_{y \in \{-1,1\}} P_p(y|\mathbf{x}) \log \hat{P}(y|\mathbf{x}) = \\ &= -\Big[P_p^+(\mathbf{x}) \log \hat{P}^+(\mathbf{x}) + (1 - P_p^+(\mathbf{x})) \log(1 - \hat{P}^+(\mathbf{x}))\Big] = \\ &= -\Big[P_p^+(\mathbf{x}) \log \frac{\hat{P}^+(\mathbf{x})}{1 - \hat{P}^+(\mathbf{x})} + \log(1 - \hat{P}^+(\mathbf{x}))\Big] = \\ &= -\Big[2P_p^+(\mathbf{x})F(\mathbf{x}) - F(\mathbf{x}) - \log(e^{F(\mathbf{x})} + e^{-F(\mathbf{x})})\Big] = \\ &= -\Big[(2P_p^+(\mathbf{x}) - 1)F(\mathbf{x}) - \log(e^{F(\mathbf{x})} + e^{-F(\mathbf{x})})\Big]. \end{aligned} \tag{8}$$

We define

$$y_p = 2P_p^+(\mathbf{x}) - 1, \quad y_p \in [-1, 1] \tag{9}$$

as the *prior-label confidence* for an unlabeled data sample induced from the prior knowledge $P_p(y|\mathbf{x})$. In order to facilitate the derivation of our boosting algorithm, we use the exponential transformation of Eq.(8) and write the unlabeled loss function as:

$$\mathcal{L}_U(F(\mathbf{x}), \mathcal{X}_U) = \frac{1}{2} \sum_{\mathbf{x} \in \mathcal{X}_U} e^{-y_p F(\mathbf{x})}(e^{F(\mathbf{x})} + e^{-F(\mathbf{x})}) = \sum_{\mathbf{x} \in \mathcal{X}_U} e^{-y_p F(\mathbf{x})} \cosh(F(\mathbf{x})). \tag{10}$$

This loss function has a very similar structure to the exponential loss of the labeled data, by interpreting the prior-label y_p as the target value of $F(\mathbf{x})$. The role of $\cosh(F(\mathbf{x}))$ can be also interpreted as a margin regularizer [11] for the boosting cost function. In fact, the $\cosh(\cdot) \geq 1$ function has a convex form with a minimum at $F(\mathbf{x}) = 0$, which can prevent the learning function to become *over-confident* and hence, can prevent over-fitting. Furthermore, if we set the prior knowledge to be a maximum entropy prior, *i.e.*$P_p(y|\mathbf{x}) = 0.5$, then for all unlabeled samples $y_p = 0$ and this loss reduces to a margin cost functional [11]. Therefore, our formulation also explains the relations of this margin regularizers to the maximum entropy learning principles [12] which is best stated by Jaynes [21] as:

Information theory provides a constructive criterion for setting up probability distributions on the basis of partial knowledge, and leads to a type of statistical inference which is called the maximum entropy estimate. It is least biased estimate possible on the given information; i.e.*, it is maximally noncommittal with regard to missing information.*

2.2 Learning

We adopt the functional gradient descent view of boosting [11, 14] to derive the loss function of the base models during each iteration of the boosting. According to the gradient descent principles, at each step of boosting, we are looking for a function $f_t(\mathbf{x})$ which if added to the current ensemble, $F(\mathbf{x})$, would result in an improvement in terms of the underlying objective function. We can write the overall loss function of Eq.(2) as:

$$\mathcal{L}(F(\mathbf{x}), \mathcal{X}) = \sum_{\mathbf{x} \in \mathcal{X}_L} e^{-yF(\mathbf{x})} + \alpha \sum_{\mathbf{x} \in \mathcal{X}_U} e^{-y_p F(\mathbf{x})} \cosh(F(\mathbf{x})). \tag{11}$$

The gradients of $\mathcal{L}(F(\mathbf{x}), \mathcal{X})$ with respect to the current models $F(\mathbf{x})$ can be written as:

$$\begin{aligned} \nabla\mathcal{L}_F = & \frac{\partial \mathcal{L}(F(\mathbf{x}), \mathcal{X})}{\partial F(\mathbf{x})} = \sum_{\mathbf{x} \in \mathcal{X}_L} -ye^{-yF(\mathbf{x})} + \\ & + \alpha \sum_{\mathbf{x} \in \mathcal{X}_U} -y_p e^{-y_p F(\mathbf{x})} \cosh(F(\mathbf{x})) + \alpha \sum_{\mathbf{x} \in \mathcal{X}_U} e^{-y_p F(\mathbf{x})} \sinh(F(\mathbf{x})). \end{aligned} \tag{12}$$

Therefore, the overall optimization problem for adding a function at t^{th} stage of the boosting can be formulated as:

$$f_t(\mathbf{x}) = \arg\max_{f(\mathbf{x})} \; -\langle \nabla\mathcal{L}_F, f(\mathbf{x}) \rangle \tag{13}$$

where $\langle A, B \rangle := \sum_{\mathbf{x}} A(\mathbf{x})B(\mathbf{x})$ is an inner product. We introduce the sample weights as

$$\forall \mathbf{x} \in \mathcal{X}_L : w_L(\mathbf{x}) = e^{-yF(\mathbf{x})} \quad \text{and} \quad \forall \mathbf{x} \in \mathcal{X}_U : w_P(\mathbf{x}) = e^{-y_p F(\mathbf{x})} \tag{14}$$

and define the pseudo-labels for the unlabeled samples as

$$\hat{y} = y_p \cosh(F(\mathbf{x})) - \sinh(F(\mathbf{x})). \tag{15}$$

Using Eq.(12), we can write the loss function $\langle \nabla \mathcal{L}_F, f(\mathbf{x}) \rangle = \mathcal{L}_f$ to be *minimized* as:

$$\mathcal{L}_f(\mathcal{X}) = \sum_{\mathbf{x} \in \mathcal{X}_L} -y w_L(\mathbf{x}) f(\mathbf{x}) + \alpha \sum_{\mathbf{x} \in \mathcal{X}_U} -\hat{y} w_P(\mathbf{x}) f(\mathbf{x}). \tag{16}$$

Therefore, if we define the pseudo-weights for unlabeled samples as $w_U(\mathbf{x}) = |\hat{y}| w_P(\mathbf{x})$, we can write the Eq.(16) as:

$$\mathcal{L}_f(\mathcal{X}) = \sum_{\mathbf{x} \in \mathcal{X}_L} -y w_L(\mathbf{x}) f(\mathbf{x}) + \alpha \sum_{\mathbf{x} \in \mathcal{X}_U} -s(\hat{y}) w_U(\mathbf{x}) f(\mathbf{x}) =$$

$$= \sum_{\substack{\mathbf{x} \in \mathcal{X}_L \\ yf(\mathbf{x})=1}} -w_L(\mathbf{x}) + \sum_{\substack{\mathbf{x} \in \mathcal{X}_L \\ yf(\mathbf{x})=-1}} w_L(\mathbf{x}) + \alpha \Big(\sum_{\substack{\mathbf{x} \in \mathcal{X}_U \\ s(\hat{y})f(\mathbf{x})=1}} -w_U(\mathbf{x}) + \sum_{\substack{\mathbf{x} \in \mathcal{X}_U \\ s(\hat{y})f(\mathbf{x})=-1}} w_U(\mathbf{x}) \Big). \tag{17}$$

where $s(\cdot)$ is the sign function. If we normalize the sample weights to sum to one, *i.e.*$\sum_{\mathbf{x}} w(\mathbf{x}) = 1$, then we know that:

$$\sum_{\substack{\mathbf{x} \in \mathcal{X}_L \\ yf(\mathbf{x})=1}} w_L(\mathbf{x}) + \sum_{\substack{\mathbf{x} \in \mathcal{X}_L \\ yf(\mathbf{x})=-1}} w_L(\mathbf{x}) = 1, \quad \sum_{\substack{\mathbf{x} \in \mathcal{X}_U \\ s(\hat{y})f(\mathbf{x})=1}} w_U(\mathbf{x}) + \sum_{\substack{\mathbf{x} \in \mathcal{X}_U \\ s(\hat{y})f(\mathbf{x})=-1}} w_U(\mathbf{x}) = 1. \tag{18}$$

As a result, we can simplify the Eq.(2.2) further as:

$$\mathcal{L}_f(\mathcal{X}) = 2 \sum_{\substack{\mathbf{x} \in \mathcal{X}_L \\ yf(\mathbf{x})=-1}} w_L(\mathbf{x}) + 2\alpha \sum_{\substack{\mathbf{x} \in \mathcal{X}_U \\ s(\hat{y})f(\mathbf{x})=-1}} w_U(\mathbf{x}) - (1+\alpha). \tag{19}$$

The first and the second term in this equation corresponds to the weighted sum of the mis-classifications of $f(\mathbf{x})$ with respect to the labeled samples and the unlabeled (pseudo-labeled) samples, respectively. The last term is a constant and thus, minimizing the loss function of Eq.(16) is equivalent to minimizing the weighted mis-classification rate. Consequently, we can use any ordinary classification model as a weak learner by taking into account the sample weights.

The overall boosting procedure is depicted in Algorithm 1. The computational complexity of our boosting method is mainly dominated by the complexity of its base models, as the overhead operations has a linear complexity in terms of number of samples.

2.3 Priors

As discussed earlier, the prior probability can be obtained in different ways, and since our method is general enough, one can use any source of information in this respect. In order to show this fact, we use the following two priors in our experiments:

Algorithm 1 SERBoost: Semi-supervised Expectation Regularization based Boosting

```
Require: Training samples: $\mathcal{X}_L$ and $\mathcal{X}_U$.
Require: Prior knowledge: $\forall \mathbf{x} \in \mathcal{X}_U, y : P_p(y|\mathbf{x})$.
Require: $T$ as the number of base models and $\alpha$ as the unlabeled loss parameter.
 1: Set the model $F(\mathbf{x}) = 0$.
 2: Set the weights
        $\forall \mathbf{x} \in \mathcal{X}_L : w_L(\mathbf{x}) = \frac{1}{|\mathcal{X}_L|}$ and $\forall \mathbf{x} \in \mathcal{X}_U : w_P(\mathbf{x}) = \frac{1}{|\mathcal{X}_U|}$
 3: Set the prior labels
        $\forall \mathbf{x} \in \mathcal{X}_U : y_p = 2P_p(y = 1|\mathbf{x}) - 1$
 4: for $t = 1$ to $T$ do
 5:     Compute the pseudo-labels
            $\forall \mathbf{x} \in \mathcal{X}_U : \hat{y} = y_p \cosh(F(\mathbf{x})) - \sinh(F(\mathbf{x}))$.
 6:     Compute the weights
            $\forall \mathbf{x} \in \mathcal{X}_U : w_U(\mathbf{x}) = |\hat{y}| w_P(\mathbf{x})$.
 7:     Normalize the weights
            $\forall \mathbf{x} \in \mathcal{X}_L : w_L(\mathbf{x}) \leftarrow w_L(\mathbf{x}) / \sum_{\mathbf{x} \in \mathcal{X}_L} w_L(\mathbf{x})$ and
            $\forall \mathbf{x} \in \mathcal{X}_U : w_U(\mathbf{x}) \leftarrow w_U(\mathbf{x}) / \sum_{\mathbf{x} \in \mathcal{X}_U} w_U(\mathbf{x})$.
 8:     Find the base function
            $f_t(\mathbf{x}) = \arg\min_{f(\mathbf{x})} \sum_{\mathbf{x} \in \mathcal{X}_L} -y w_L(\mathbf{x}) f(\mathbf{x}) + \alpha \sum_{\mathbf{x} \in \mathcal{X}_U} -\mathrm{sign}(\hat{y}) w_U(\mathbf{x}) f(\mathbf{x})$.
 9:     Update the model
            $F(\mathbf{x}) \leftarrow F(\mathbf{x}) + f_t(\mathbf{x})$.
10:     Update the weights
            $\forall \mathbf{x} \in \mathcal{X}_L : w_L(\mathbf{x}) \leftarrow w_L(\mathbf{x}) e^{-y f_t(\mathbf{x})}$ and
            $\forall \mathbf{x} \in \mathcal{X}_U : w_P(\mathbf{x}) \leftarrow w_P(\mathbf{x}) e^{-y_p f_t(\mathbf{x})}$.
11: end for
12: Output the final model: $F(\mathbf{x})$
```

Maximum Entropy This is the simplest prior one can think of: $\forall \mathbf{x}, y : P_p(y|\mathbf{x}) = 0.5$. This can be stated as the maximum entropy prior which requires no knowledge from the underlying problem. By using this prior, we can study the effect of our margin regularizer term in Eq.(10).

Knowledge Transfer In a knowledge transfer scenario, we have a previously estimated model, and with minimal supervision effort, we would like to include its knowledge for training a new model (*e.g.* [13]). To show how this procedure can be incorporated into our framework, we train another classifier over the labeled data set, and use its predictions over the unlabeled samples as priors.

3 Experiments

3.1 Data Sets and Evaluation Methodology

We test the performance of our method on the challenging object category recognition data sets of Pascal Visual Object Class Challenge 2006 [22]. This dataset consists of 2615 training and 2686 test images coming from 10 different categories. In our experiments, we solve the multi-class problems with a one-vs-rest binary classification strategy.

In order to observe the effect of including the unlabeled data into the learning process of our boosting algorithm, we randomly partition the training set into two disjoint sets of labeled and unlabeled samples. The size of the labeled partition is set to be $r = 0.01$, 0.05, 0.1, 0.25, and 0.5 times of the number of all training samples. We repeat the procedure of producing random partitions for 10 times and report the average of the area under the curve (AUC) for each model described in Section 3.3.

3.2 Feature Extraction

For feature extraction, we use a *bag-of-words* model which is partially similar to the top-ranked participants of Pascal challenge in 2006 [22]. We first extract three sets of interest points with complementary behaviours: the Harris-Laplacian (*HL*) points [23] for corner-like regions, the Difference of Gaussians (*DoG*) points [24] for blob-like regions, and a regular dense sampling (*Reg*) with a grid size of 8 pixels. Then we use SIFT [24] to describe these regions. For the dense sampling method, we apply the SIFT-descriptor to patches with multiple scales of 8, 16, 24, and 32 pixels. Following [17], we form three channels of *HL-SIFT*, *DoG-SIFT*, and *Reg-SIFT*. For each channel, we find the class-specific visual vocabulary by randomly selecting 50000 interest regions from 10 training images of the target class and by forming 100 cluster centers using k-means method. The final vocabulary is the concatenation of all class-specific cluster centers. Afterwards, each interest point descriptor is assigned to its closest cluster center. We use the normalized 2-level spatial pyramids [25] to represent each image and this way we construct 2 channels of $L0$ and $L1$ from each of *HL-SIFT*, *DoG-SIFT*, and *Reg-SIFT* channels. Finally, we concatenate the image descriptors of six channels to create our feature space. Hence, the dimensions of the feature space is 15000 for VOC2006. As a preprocessing, we normalize each sample to have a unit L1-norm.

3.3 Models

We compare the performance of the following supervised and semi-supervised classification models:

χ^2-SVM This model is a popular classifier for the bag-of-words approach, with an excellent performance in object categorization problems [17–19]. The feature kernel is constructed by a combination of χ^2 distances between each level of spatial pyramids with the same weightings suggested in [25] as:

$$K_F(\mathbf{x}_i, \mathbf{x}_k) = \sum_{l=0}^{1} \exp(-d_l(\mathbf{x}_i, \mathbf{x}_k)/\sigma_l) \tag{20}$$

where d_l is the χ^2 distance, and σ_l is the average distance of l^{th} level, respectively. The *LibSVM* package [26] is utilized for training and testing of SVMs. It should be noted that after a few cross-validation experiments, we fix the hinge loss parameter C of SVMs to be 5 in all experiments as it gives equally good performance for all classes.

Lin-SVM From computational complexity point of view, the χ^2-SVM model is the heaviest amongst all methods we study in this paper. In order to provide a similar model which behaves better with respect to the number of samples, we also use a linear SVM by utilizing the *LibLinear* [27] software package. We also set the hinge loss parameter C to be 5 by conducting a 5-fold cross-validation for this model.

TSVM We also compare the performance of our model with the Transductive Support Vector Machines (TSVM) [15] which is a popular semi-supervised formulation of SVMs. For this model, we use the *SVMLight* [15] package. Due to its computational costs, we use the same parameter settings of the Lin-SVM model here as well.

Random Forest The Random Forest (RF) classifier [16] is a collection of binary decision trees. These trees are grown separately and by introducing randomness at different levels of their learning process, one can get an ensemble of trees which collectively has a reasonable performance. Due to their efficiency and fast training/testing characteristics, RFs are gaining more attention in vision community too (*e.g.* [28, 29]).

Following the original idea of Breiman [16], we grow the trees to a maximum depth without pruning by computing a set of random tests at each decision node. For each decision node, we first select a number of features randomly, and then select randomly a few linear hyperplanes constructed from these features. Afterwards, we select the best test according to their Gini indexes. Bosch *et al.* [29] used 100 deep trees with depth of 20, computed considerably a large number of random tests, and performed a random selection of different channels for each test. It should be noted that decision trees are not efficient with respect to their depth, both from memory and computational complexity point of view, and additionally, in semi-supervised experiments, the number of labeled samples could be too low to create deep trees. Therefore, we grow shallow trees with maximum depth 2, use 10 random hypotheses, and construct a random forests separately for each feature channel and average their predictions. As a result, one of the main benefits of our approach is that we can effortlessly create huge forests with as high as 10000 trees in our experiments.

GentleBoost If we ignore the unlabeled part of our algorithm (or set the α to zero), we end up with the GentleBoost method of Friedman *et al.* [14]. As a result, a comparison of our method with the GentleBoost enables us to study the

Method	χ^2-SVM	Lin-SVM	RF	GB	QMUL_LSPCH
AUC	0.9243	0.8911	0.8456 ± 0.0025	0.8978 ± 0.0012	0.936
Time	885	82	98	116	-

Table 1. First row: the average AUC for the χ^2-SVM, Lin-SVM, Random Forest (RF), GentleBoost (GB) models, and the winner of VOC2006 (QMUL_LSPCH). Second row: the average computation time for each model in minutes.

effect of including the unlabeled data into the learning process of boosting. As weak learners, we use small random forests with 40 shallow trees grown exactly as specified previously. The only difference is that at each stage of boosting, we construct a separate RF for each feature channel, and instead of averaging their results, we let the boosting to select the best one. We iterate the GentleBoost algorithm for 250 iterations, as with our implementation this gives a comparable computation time compared to the Lin-SVM model.

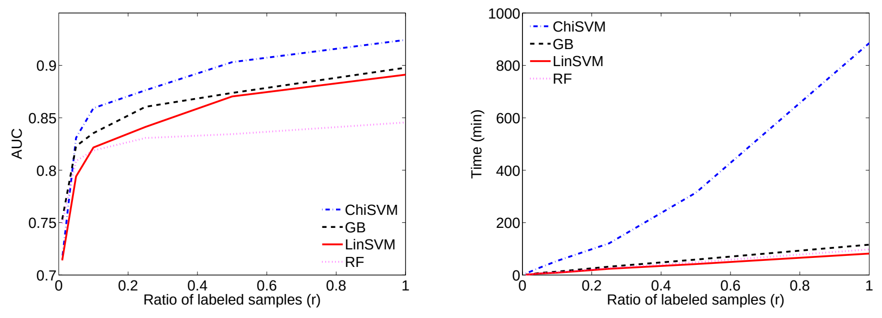
SERBoost The same settings as for Gentle Boosting are used for training the weak learners of the SERBoost algorithm. We also iterate SERBoost for 250 rounds. We use the predictions of the χ^2-SVM model over the unseen unlabeled training data (the χ^2-SVM is trained only on labeled partition) for simulating a knowledge transfer scenario. One should note that this is just an example of how other classifiers could contribute to the learning process of SERBoost, and from application point of view the same principles can be applied to other knowledge transfer scenarios.

3.4 Results

Fully Supervised Methods Table 1 shows the performance of the supervised models: χ^2-SVM, Lin-SVM, Random Forest (RF), and GentleBoost (GB), trained over the full training set ($r = 1$), together with the average computation time. We also provide the performance of the winner of VOC2006 challenge [22] as a reference.

Comparing the performance of the different models, it is clear that the χ^2-SVM produces the best result, followed by Lin-SVM and GentleBoost, while the Random Forest does not seem to be competitive. However, looking into the timings, the χ^2-SVM has considerably larger computation burden compared to all other methods. It should be noted that for the Random Forest, GentleBoost, and SERBoost methods we use our naive and unoptimized C++ implementation, while the other packages used for χ^2-SVM, Lin-SVM, and TSVM are heavily optimized regarding computation speed.

Figure 1(a) shows the performance of the fully supervised models with respect to the ratio of labeled samples in the training set. As expected, the χ^2-SVM is producing the best performance by paying the price of heavier computations. It is



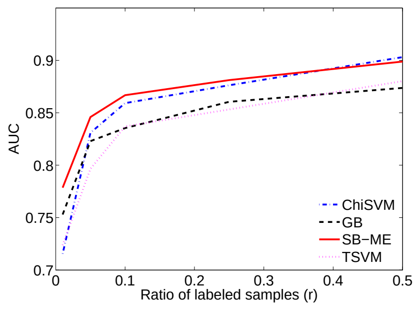
(a) (b)

Fig. 1. The performance (a) and computation times (b) of the χ^2-SVM, Lin-SVM, Random Forest (RF), and GentleBoost (GB) models with respect to the ratio of the labeled samples in training set.

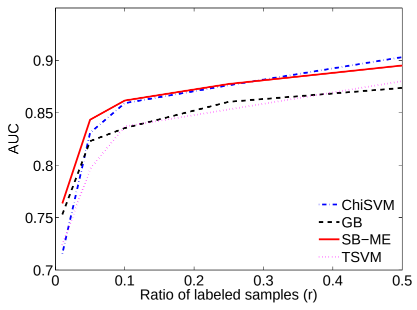
also clear that the GentleBoost has usually a better or comparable performance compared to Lin-SVM.

Maximum Entropy Prior We turn our attention to study the behaviour of the TSVM and our SERBoost models. Figure 2 shows the performance of the SERBoost for two different values of unlabeled loss parameter $\alpha = 0.1, 0.25$ and when the maximum entropy prior is used. This figure also shows the performance of TSVM and the performance of χ^2-SVM and GentleBoost from Figure 1(a) as references. The first considerable observation is that the SERBoost is performing better or comparable to χ^2-SVM even when there is no prior knowledge included in its learning process. As a matter of fact, SERBoost outperforms χ^2-SVM when the number of labeled images are very low, and as we continue to add more labels their performances become very close, and eventually after approximately $r = 0.5$, χ^2-SVM starts to perform better. It is also clear that TSVM is not competitive neither in terms of performance nor in terms of computation time with requiring 518 minutes for a single run. It should be noted that SERBoost has on average 14 minutes computation overhead compared to the GentleBoost.

Comparison to GentleBoost From Figure 2, it is also clear that SERBoost opens a large gap compared to its fully supervised counter-part, GentleBoost. In order to investigate this fact, we conducted another set of experiments where we duplicated the full training set and used the first half as the labeled set and the second half as the unlabeled partition. We applied SERBoost with maximum entropy prior to this data set and report their results in Table 2. One can clearly observe that even with using the full training set and no additional information, SERBoost outperforms GentleBoost by a nice margin in terms of AUC. It is also



(a) $\alpha = 0.1$



(b) $\alpha = 0.25$

Fig. 2. The performance of SERBoost (SB) with maximum entropy prior (ME) for two different values of unlabeled loss parameter, α.

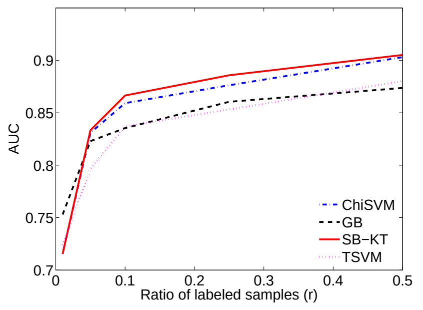
Method	GB	SB, $\alpha = 0.01$	SB, $\alpha = 0.1$	SB, $\alpha = 0.25$
AUC	0.8978 ± 0.0012	0.9125 ± 0.0014	0.9153 ± 0.0016	0.914 ± 0.0015

Table 2. The performance comparison of GentleBoost (GB) and SERBoost (SB) when trained on full labeled training set.

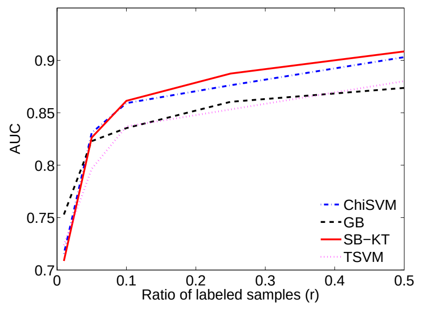
interesting to see that with this simple approach, SERBoost comes closer to the results of the winner of VOC2006.

Knowledge Transfer Prior Since our method provides a principled way of including the prior knowledge as an additional source of information, we conduct experiments by training the χ^2-SVM over the labeled partition and use its predictions over the unlabeled partition as priors for SERBoost. The results are shown in Figure 3 for two different values of α. When the number of labeled samples are low, the predictions of the prior are mostly wrong, and therefore the performance of SERBoost is inferior to those of trained with maximum entropy priors. However, as the χ^2-SVM starts to produce more reliable predictions, our method also starts to improve its performance. As it can be seen, by moving towards larger labeled sets, our method utilizes the priors well and outperforms the maximum entropy based model.

Hyperparameter Sensitivity Another considerable fact is the robustness of the SERBoost with respect to the variations of α within a reasonable working range. If we compare the pairs of left and right plots in Figures 2 and 3, we can see that the performance changes smoothly when one varies α. For example in Figure 3, one can see that when the χ^2-SVM predictions are not reliable (lower values of r), having a smaller α results in a slight performance gain, while the figure is reversed when the χ^2-SVM starts to operate reasonably. However, the overall change in the performance is not significant.



(a) $\alpha = 0.1$



(b) $\alpha = 0.25$

Fig. 3. The performance of SERBoost (SB) with prior knowledge (KT) for two different values of unlabeled loss parameter, α.

4 Conclusion

In this paper, we derived a novel semi-supervised boosting method, called SERBoost, on the principles of expectation regularization. This algorithm provides a principled way of including the prior knowledge into learning process of a model and naturally explains the probabilistic interpretation of the provided boosting margin regularizer with the maximum entropy learning concept. SERBoost scales very well with respect to the number of labeled and unlabeled samples, is very easy to implement, and is robust with respect to the variations of its hyperparameter. The experimental results shows that SERBoost is able to exploit the obscured information hidden in unlabeled data and can benefit easily from a prior knowledge or domain expertise. SERBoost currently is designed for binary classification tasks, and we plan to investigate the possibility of extending it to multi-class/label problems.

References

1. Zhu, X.: Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison (2005)
2. Chapelle, O., Schölkopf, B., Zien, A., eds.: Semi-Supervised Learning. Cambridge, MA (2006)
3. Buc, D.F., Grandvalet, Y., Ambroise, C.: Semi-supervised marginboost. In: Proc. of NIPS. (2002) 553–560
4. Bennett, K.P., Demiriz, A., Maclin, R.: Exploiting unlabeled data in ensemble methods. In: Proc. of KDD. (2002) 289–296
5. Mallapragada, P.K., Jin, R., Jain, A.K., Liu, Y.: Semiboost: Bossting for semi-supervised learning. Technical report, Department of Computer Science, Michigan State University (2007)
6. Chen, K., Wang, S.: Regularized boost for semi-supervised learning. In: Proc. of NIPS. (2008)

7. Cohen, I., Sebe, N., F.G.Cozman, M.C.Cirelo, Huang, T.: Learning bayesian network classifiers for facial expression recognition using both labeled and unlabeled data. In: Proc. of CVPR. Volume 1. (2003) 595–604
8. Yao, J., Zhang, Z.: Semi-supervised learning based object detection in aerial imagery. In: Proc. of CVPR. Volume 1. (2005) 1011–1016
9. Li, L.J., Wang, G., Fei-Fei, L.: Optimol: automatic object picture collection via incremental model learning. In: Proc. of CVPR. (2007)
10. Mann, G.S., Mccallum, A.: Simple, robust, scalable semi-supervised learning via expectation regularization. In: Proc. of ICML. (2007) 593–600
11. Mason, L., Baxter, J., Bartlett, P., Frean, M. In: Advances in Large Margin Classifiers. MIT Press, Cambridge, MA. (1999) 221–247
12. Berger, A.L., Della Pietra, V.J., Della Pietra, S.A.: A maximum entropy approach to natural language processing. Comput. Linguist. **22** (1996) 39–71
13. Schapire, R.E., M.Rochery, Rahim, M., Gupta, N.: Incorporating prior knowledge into boosting. In: Proc. of ICML. (2002)
14. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. The Annals of Statistics **38** (2000) 337–374
15. Joachims, T.: Transductive inference for text classification using support vector machines. In: Proc. of ICML. (1999) 200–209
16. Breiman, L.: Random forests. Machine Learning **V45** (2001) 5–32
17. Marszalek, M., Schmid, C.: Spatial weighting for bag-of-features. In: Proc. of CVPR. (2006) 2118–2125
18. Zhang, J., Marszalek, M., Lazebnik, S., Schmid, C.: Local features and kernels for classification of texture and object categories: A comprehensive study. IJCV **73** (2007) 213–238
19. Bosch, A., Zisserman, A., Munoz, X.: Representing shape with a spatial pyramid kernel. In: Proc. of CIVR. (2007) 401–408
20. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: Proc. of ICML. (1996) 148–156
21. Jaynes, E.T.: Information theory and statistical mechanics. Physical Review **106** (1957) 620–630
22. Everingham, M., Zisserman, A., Williams, C.K.I., Van gool, L.: The pascal visual object classes challenge 2006 (voc2006) results. Technical report (2006)
23. Mikolajczyk, K., Schmid, C.: Scale and affine invariant interest point detectors. IJCV **60** (2004) 63–86
24. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
25. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Proc. of CVPR. (2006) 2169–2178
26. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm (2001)
27. Lin, C.J., Weng, R.C., Keerthi, S.S.: Trust region newton method for large-scale logistic regression. Technical report (2007)
28. Lepetit, V., Fua, P.: Keypoint recognition using randomized trees. PAMI **28** (2006) 1465–1479
29. Bosch, A., Zisserman, A., Munoz, X.: Image classification using random forests and ferns. In: Proc. of ICCV. (2007)