Robust Multi-View Boosting with Priors*

Amir Saffari, Christian Leistner, Martin Godec, and Horst Bischof

Institute for Computer Graphics and Vision, Graz University of Technology,
Inffeldgasse 16, 8010 Graz, Austria
{saffari,leistner,godec,bischof}@icg.tugraz.at
http://www.icg.tugraz.at

Abstract. Many learning tasks for computer vision problems can be described by multiple views or multiple features. These views can be exploited in order to learn from unlabeled data, a.k.a. "multi-view learning". In these methods, usually the classifiers iteratively label each other a subset of the unlabeled data and ignore the rest. In this work, we propose a new multi-view boosting algorithm that, unlike other approaches, specifically encodes the uncertainties over the unlabeled samples in terms of given priors. Instead of ignoring the unlabeled samples during the training phase of each view, we use the different views to provide an aggregated prior which is then used as a regularization term inside a semi-supervised boosting method. Since we target multi-class applications, we first introduce a multi-class boosting algorithm based on maximizing the mutli-class classification margin. Then, we propose our multi-class semi-supervised boosting algorithm which is able to use priors as a regularization component over the unlabeled data. Since the priors may contain a significant amount of noise, we introduce a new loss function for the unlabeled regularization which is robust to noisy priors. Experimentally, we show that the multi-class boosting algorithms achieves state-of-the-art results in machine learning benchmarks. We also show that the new proposed loss function is more robust compared to other alternatives. Finally, we demonstrate the advantages of our multi-view boosting approach for object category recognition and visual object tracking tasks, compared to other multi-view learning methods.

1 Introduction

In recent years, the development and design of classification algorithms has led to significant progress in various computer vision domains. In most applications supervised learning algorithms are applied. Usually, these methods require large amounts of training samples along with their class labels in order to train a classification function that yields low prediction errors. In practice, the class labels are provided by a human labeler. As the tedious hand labeling cannot take pace with the growing amount of data, *e.g.*, digital images, web sites, research has started to focus on semi-supervised learning (SSL) methods [1, 2] that can

* This work has been supported by the Austrian FFG project MobiTrick (825840) and Outlier (820923) under the FIT-IT program.

learn from a small set of labeled data and simultaneously a huge amount of unlabeled data.

This paper deals with a special case of SSL called multi-view learning [3, 4]. In multi-view learning (MVL), the data can be expressed by several views or multiple features. For each view a classifier is trained on some labeled data. Then the classifiers iteratively train each other on the unlabeled data. The underlying assumption of MVL is that the unlabeled data can be exploited in a way that enforces the selection of hypotheses that lead to an agreement among the classifiers on the unlabeled data while minimizing the training error on labeled data [5, 6]. Overall, this leads to an increased classification margin and thus lower generalization errors. MVL is especially interesting for many computer vision tasks as multiple views are often naturally provided. For instance, in object detection and categorization different features can be considered as different views [7, 8]. Multi-view learning can also lead to more stable tracking results [9, 10]. Also images collected from the web naturally provide different views, because additional to the visual data text is also frequently provided [11].

Current multi-view methods work by primarily exchanging the information via label predictions on a subset of the unlabeled data. However, this ignores the uncertainty in each estimated label and ignores the information that each view has over the entire set of unlabeled data. In this paper, we propose a novel multi-view boosting algorithm that, on the one hand, performs MVL in the classical sense; *i.e.*, the classifiers provide each other labels for some selected unlabeled samples. On the other hand, however, we regularize each classifier on the rest of the unlabeled samples in a way that it encourages the agreement between the views. In our algorithm, we use an aggregated prior that is set up by the corresponding views; *i.e.*, the iteratively trained classifiers serve each other as priors in order to exploit the rest of the unlabeled samples. However, since the priors can be wrong, we also propose a robust loss function for the semi-supervised regularization which can handle noisy priors. Additionally, most previous MVL methods mainly deal with two-classifier scenarios and are thus mainly co-training variants. Our method is general enough to incorporate not only two, but even an arbitrary number of views.

2 Multi-View Boosting with Priors

In the following sections, we first introduce the concept of multi-view learning with priors. Next, we explain how we can develop multi-class semi-supervised boosting which uses priors provided from multiple views as a regularization term. Finally, we show how robustness can be incorporated into the learning algorithm in terms of proper loss functions.

2.1 Multi-View Learning with Priors

Assume we have a multi-class semi-supervised classification task where the problem domain can be split into V different views [1]. Let $\mathbf{x} = [\mathbf{x}_1^T | \cdots | \mathbf{x}_V^T]^T$ be a data sample which is constructed from V different views, each expressed by D_v-dim feature vector $\mathbf{x}_v \in \mathbb{R}^{D_v}$. In multi-view learning, we train a classifier per view $\mathbf{f}_v(\mathbf{x}_v) : \mathbb{R}^{D_v} \to \mathbb{R}^K$ where K is the number of classes and $\mathcal{F} = \{\mathbf{f}_v\}_{v=1}^V$ is the set of the classifiers. Let $p_v(k|\mathbf{x}_v)$ be the posterior estimate for classifying sample $\mathbf{x}_v$ in k-th class by the v-th learner. The goal of multi-view learning is to produce a set of classifiers which have low mis-classification rates over the labeled samples while having a high consensus over the unlabeled samples. One can express these goals as the following optimization problem

$$\mathcal{F}^* = \arg\min_{\mathcal{F}} \sum_{(\mathbf{x}\in\mathcal{X}_l, y)} \ell(\mathbf{x}, y; \mathcal{F}) + \gamma \sum_{\mathbf{x}\in\mathcal{X}_u} d(\mathbf{x}; \mathcal{F}). \tag{1}$$

The first term expresses the loss $\ell(\cdot)$ for the labeled samples where we have the true class label y, while the last term is a measure of the agreement of views over the unlabeled samples, and γ steers the effect of the unlabeled samples over the entire optimization problem. In this work, we propose to use the posterior estimates for defining the loss over the unlabeled samples. Assume we have a function $\jmath(p\|q)$ for measuring the divergence between two probabilities p and q. Using this divergence measure, we express the unlabeled loss as $d(\mathbf{x}; \mathcal{F}) = \sum_{v=1}^V d_v(\mathbf{x}; \mathcal{F})$ with

$$d_v(\mathbf{x}; \mathcal{F}) = \jmath(p_v(\mathbf{x}_v) \| \frac{1}{V-1} \sum_{s\neq v} p_s(\mathbf{x}_s)), \tag{2}$$

where $p_v(\mathbf{x}_v) = [p_v(1|\mathbf{x}_v), \cdots, p_v(K|\mathbf{x}_v)]^T$. This loss function measures the divergence of the posterior estimates by computing the distance of each view to the average estimate of all other views. For example, if we use $\jmath(p\|q) = \sum_{k=1}^K (p(k|\mathbf{x}) - q(k|\mathbf{x}))^2$ the last term will measure the variance over different views (the proof is omitted due to lack of space). As it will be shown later, we use the *Jensen-Shannon Divergence* as $\jmath(p\|q)$ in our algorithm because of its robustness to noise. We will also refer to

$$q_v(k|\mathbf{x}_v) = \frac{1}{V-1} \sum_{s\neq v} p_s(k|\mathbf{x}_s),\ \forall v \in \{1, \cdots, V\}, k \in \mathcal{Y} \tag{3}$$

as the prior for the v-th view. In order to observe the advantages gained by using this approach over the traditional multi-view learning where the consensus is only encouraged by iterative labeling of the unlabeled data, we propose the following algorithm:

[1] For clarity, we always use the co-training settings [3] where the data is represented by different views, while the algorithm can be applied to multiple-learners scenario as well [4].

1. For each view independently, optimize Eq. (1) by using Eq. (2) and using the Eq. (3) as the priors.
2. Label a subset of unlabeled samples and add them to the labeled set.
3. Compute the posteriors and update the priors for each view.
4. If stopping condition is not satisfied, proceed to step 1, otherwise stop.

If we set $\gamma = 0$ in this procedure, then we obtain a classical multi-view learning algorithm, similar to co-training [3]. Therefore, by observing the performance of both of these algorithms, one can see the gain obtained by incorporating the priors. For the second step where we label some unlabeled samples, we use the following approach: 1) Each view proposes the N highest confident samples to be included in the labeled set. 2) If there are disagreements over the label of a sample between some of the views, we let the proposing views to vote with their confidence for the label of this sample, and we select the resulting class which has the highest aggregated confidence. Again, if we would have only two views, this would be equivalent to the original co-training algorithm [3]. In the following sections, we will develop a semi-supervised multi-class boosting algorithm which can be used to solve the first step of this algorithm.

2.2 Multi-Class Boosting

Let $\mathcal{X}_l = \{(\mathbf{x}, y)|\mathbf{x} \in \mathbb{R}^D, y \in \{1, \cdots, K\}\}$ to be the set of *i.i.d.* labeled training examples from an unknown probability distribution $P(y, \mathbf{x})$. Suppose we are given a set of unlabeled samples $\mathcal{X}_u = \{\mathbf{x}|\mathbf{x} \in \mathbb{R}^D\}$ which are also sampled *i.i.d.* from the marginal distribution $P(\mathbf{x}) = \sum_{y \in \mathcal{Y}} P(y, \mathbf{x})$. With $\mathcal{X}$ we refer to the union of both labeled and unlabeled data samples. The data sample $\mathbf{x}$ is represented as a D-dimensional feature vector and its label for a K-class problem y is coming from the set of labels $\mathcal{Y} = \{1, \ldots, K\}$. Boosting can be considered as a *meta-learning* algorithm, which accepts another learning algorithm (often known as *base* or *weak* learner) and constructs a new function class out of it, *i.e.*, by constructing additive models in form of

$$\mathbf{f}(\mathbf{x}; \boldsymbol{\beta}) = \sum_{t=1}^{T} \alpha_t \, \mathbf{g}(\mathbf{x}; \boldsymbol{\theta}_t), \tag{4}$$

where $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \cdots, f_K(\mathbf{x})]^T$ is the multi-class classifier[2], $\boldsymbol{\beta} = [\boldsymbol{\alpha}|\boldsymbol{\theta}]$ is the collection of model parameters, $\boldsymbol{\alpha}$ are the parameters of boosting algorithm, $\boldsymbol{\theta} = \{\boldsymbol{\theta}_t\}_{t=1}^T$, and $\boldsymbol{\theta}_t$ represents the parameters of the t-th base learner $\mathbf{g}(\mathbf{x}; \boldsymbol{\theta}_t) \in \mathcal{G} : \mathbb{R}^D \rightarrow \mathbb{R}^K$. Without loss of generality, we require the following symmetry condition: $\forall \mathbf{x} : \sum_{k \in \mathcal{Y}} f_k(\mathbf{x}) = 0$. Many machine learning algorithms rely on the notion of *margin*, popularized by *support vector machines* (SVMs). For a K-class problem, the multi-class margin can be described as

$$m(\mathbf{x}, y; \mathbf{f}) = f_y(\mathbf{x}) - \max_{k \neq y} f_k(\mathbf{x}). \tag{5}$$

[2] When the context is clear, we interpret $\mathbf{f}(\mathbf{x}; \boldsymbol{\beta})$ and $\mathbf{f}(\mathbf{x})$ as the same representation for a classifier.

Note that for a correct classification via the decision rule $c(\mathbf{x}) = \arg\max_{k \in \mathcal{Y}} f_k(\mathbf{x})$, the margin should be positive $m(\mathbf{x}, y; \mathbf{f}) > 0$. In other words, a negative margin would result in a mis-classification. In this work, we introduce a boosting algorithm which relies on maximizing the true multi-class margin. This is accomplished by minimizing a loss function which uses the multi-class margin. In details, our boosting algorithm minimizes the following *empirical risk*

$$R_{emp}(\boldsymbol{\beta}) = \sum_{(\mathbf{x},y) \in \mathcal{X}_l} \ell(\mathbf{x}, y; \boldsymbol{\beta}), \tag{6}$$

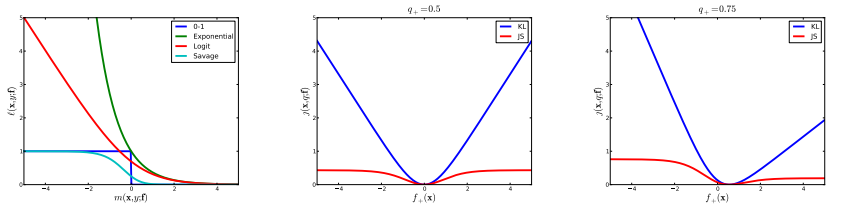
where $\ell(\cdot)$ is a loss function. Since our learning strategy is based on functional gradient descent technique, it is possible to use a lot of different loss functions here [12]. The usual choices are *Exponential loss* [13], *Logit loss* [14], and *Savage loss* [15]. Figure 1(a) plots the shape of these loss functions with respect to the margin of an example. As it has been shown in previous studies, in the presence of label noise, Savage and Logit loss perform significantly better than exponential loss [14, 15]. Since in this work, the multi-view algorithm might introduce noise into the labeled set, we use the Savage loss function for the supervised loss, *i.e.*, $\ell(\mathbf{x}, y; \mathbf{f}) = \frac{1}{(1+e^{2m(\mathbf{x},y;\mathbf{f})})^2}$.

2.3 Semi-Supervised Boosting with Robust Loss Functions

We now focus on developing the multi-class semi-supervised boosting algorithms based on the concept of learning from priors [16, 17]. Assume we are given a prior probability in form of $q(\cdot|\mathbf{x})$, *e.g.* Eq. (3). We model the posterior estimates of the model by a *multi-nomial logistic regression* model defined as

$$p(k|\mathbf{x}; \boldsymbol{\beta}) = \frac{e^{f_k(\mathbf{x};\boldsymbol{\beta})}}{\sum_{j \in \mathcal{Y}} e^{f_j(\mathbf{x};\boldsymbol{\beta})}}, \tag{7}$$

where $p(k|\mathbf{x}; \boldsymbol{\beta})$ is the posterior probability of assigning sample $\mathbf{x}$ to the k-th class, estimated by a model parameterized by $\boldsymbol{\beta}$.



(a) Sup. Losses (b) Unsup. Losses: $q_+ = 0.5$ (c) Unsup. Losses: $q_+ = 0.75$

Fig. 1. (a) Common loss functions used for supervised boosting methods. (b, c) Divergence measures used for prior regularization with two different priors: (b) $q_+ = 0.5$ and (c) $q_+ = 0.75$.

We formulate the semi-supervised learning as an optimization problem with two goals: 1) the model should attain low mis-classification errors on the labeled data (*e.g.*, by means of maximizing the margin), 2) the model should be able to match the prior probability over the unlabeled training samples. From an *empirical risk minimization* perspective, these goals can be written as the following risk functional

$$R_{emp}(\boldsymbol{\beta}) = \sum_{(\mathbf{x},y)\in\mathcal{X}_l} \ell(\mathbf{x}, y; \boldsymbol{\beta}) + \gamma \sum_{\mathbf{x}\in\mathcal{X}_u} \jmath(\mathbf{x}, q; \boldsymbol{\beta}), \tag{8}$$

where $\jmath$ is a loss function which measures the deviations of the model from prior for a given sample $\mathbf{x}$, and γ tunes the effect of semi-supervised regularization. Since the goal of the regularization in Eq. (8) is to measure the deviations between two probabilities, it is natural to use loss functions which measure the divergence between two given distributions. In this work, we propose the *Jensen-Shannon Divergence* for the regularization term in Eq. (8). The Jensen-Shannon divergence for measuring the deviations of the model from the prior can be expressed as

$$\jmath(\mathbf{x}, q; \boldsymbol{\beta}) = \frac{1}{2}(D_{KL}(q\|m) + D_{KL}(p\|m)), \tag{9}$$

where $D_{KL}(\cdot\|\cdot)$ is the Kullback-Leibler Divergence, $m = \frac{1}{2}(p+q)$. Figure 1(b) plots the shape of the *Kullback-Leibler Divergence* (KL), which has been used previously by Saffari *et al.* [17] for developing a multi-class semi-supervised boosting algorithm, together with the *Jensen-Shannon Divergence* (JS) for a binary problem when the prior is 0.5 for both classes. Figure 1(c) shows the same loss functions when the prior is 0.75 for the positive class. By comparing these loss functions to the supervised loss functions in Figure 1(a), one can see that KL resembles a behavior similar to the Logit loss while JS is very similar to the Savage loss function. Therefore, we could expect the JS loss function to be more robust when faced with noisy priors; *i.e.*, the prior is wrong about the label of an unlabeled sample.

2.4 Learning with Functional Gradient Descent

Given the loss functions from previous sections, the learning process for boosting is defined by the Eq. (8). This requires finding the parameters of the base learners $\boldsymbol{\theta}$ together with their weights $\boldsymbol{\alpha}$. Finding a global solution for this problem is hard, therefore, many boosting algorithms adopt an approximate solution called *stagewise additive modeling* [14]. One of the commonly used techniques is the *functional gradient descent* method [18]. This is a generalization of the traditional gradient descent method to the space of functions. In details, at the t-th iteration of boosting, we find the *steepest descent* direction as $-\nabla R_{emp}(\boldsymbol{\beta}_{t-1})$, where ∇ is the gradient operator. Then the optimization problem for learning t-th weak learner can be written as

$$\boldsymbol{\theta}_t^* = \arg\max_{\boldsymbol{\theta}_t} \langle -\nabla R_{emp}(\boldsymbol{\beta}_{t-1}), \mathbf{g}(.; \boldsymbol{\theta}_t)\rangle, \tag{10}$$

where $\langle , \rangle$ is the inner product operator. Since we have a risk which has labeled and unlabeled samples, we need to compute the gradients for two different loss functions.

Gradients for Labeled Samples Using the chain rule, we can write the k-th element of the gradient vector $\nabla R_{emp}(\boldsymbol{\beta}_{t-1})$ as [3]

$$\frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_k(\mathbf{x})} = \frac{\partial \ell(\mathbf{x}, y; \boldsymbol{\beta}_{t-1})}{\partial m(\mathbf{x}, y; \mathbf{f})} \frac{\partial m(\mathbf{x}, y; \mathbf{f})}{\partial f_k(\mathbf{x})}. \tag{11}$$

Note that the margin term includes a max operator, therefore, the derivatives of the margin can be written as

$$\frac{\partial \max\limits_{j \neq y} f_j(\mathbf{x})}{\partial f_k(\mathbf{x})} = \mathbb{I}(k \neq y) \frac{\mathbb{I}(k \in \mathcal{S}_y(\mathbf{f}(\mathbf{x}))}{|\mathcal{S}_y(\mathbf{f}(\mathbf{x}))|}, \tag{12}$$

where $\mathbb{I}$ is the indicator function, and $\mathcal{S}_y(\mathbf{f}(\mathbf{x})) = \{k | f_k(\mathbf{x}) = \max\limits_{j \neq y} f_j(\mathbf{x}))\}$ is the set of classes which are the closest to the target class y. By using these results, we compute the gradients of the margin as

$$\frac{\partial m(\mathbf{x}, y; \mathbf{f})}{\partial f_k(\mathbf{x})} = \mathbb{I}(k = y) - \mathbb{I}(k \neq y) \frac{\mathbb{I}(k \in \mathcal{S}_y(\mathbf{f}(\mathbf{x}))}{|\mathcal{S}(\mathbf{f}(\mathbf{x}))|}. \tag{13}$$

Now we only need to find the derivatives of the loss function with respect to the margin term, which for the Savage loss can be written as

$$\frac{\partial \ell(\mathbf{x}, y; \boldsymbol{\beta}_{t-1})}{\partial m(\mathbf{x}, y; \mathbf{f})} = -\frac{4e^{2m(\mathbf{x}, y; \mathbf{f}_{t-1})}}{(1 + e^{2m(\mathbf{x}, y; \mathbf{f}_{t-1})})^3}. \tag{14}$$

Gradients for Unlabeled Samples The Jensen-Shannon divergence for semi-supervised part can be written as [4]

$$\begin{aligned} \jmath(\mathbf{x}, q; \boldsymbol{\beta}) \approx & H(p, m) + H(q, m) - H(p) = \\ = & -2 \sum_{j \in \mathcal{Y}} m(j|\mathbf{x}; \boldsymbol{\beta}) \log m(j|\mathbf{x}; \boldsymbol{\beta}) + \sum_{j \in \mathcal{Y}} p(j|\mathbf{x}; \boldsymbol{\beta}) \log p(j|\mathbf{x}; \boldsymbol{\beta}). \end{aligned} \tag{15}$$

where $H(\cdot, \cdot)$ is the cross-entropy between two distributions and $H(\cdot)$ is the entropy. Note that since $H(q)$ is fixed and does not depend on the model, it is dropped from this equations. We will need to compute the gradients of the posterior estimates in this equation, therefore, first we develop this term. We

[3] For notational brevity, we simply write $\frac{\partial \ell(\mathbf{x}, y; \mathbf{f})}{\partial f_k(\mathbf{x})}$ instead of the more correct form $\frac{\partial \ell(\mathbf{x}, y; \mathbf{f})}{\partial f_k(\mathbf{x})}|_{\mathbf{f}(\mathbf{x}) = \mathbf{f}_{t-1}(\mathbf{x})}$.

[4] Note that we drop the $\frac{1}{2}$ multiplier of the Eq. (9) as it can be incorporated into γ.

write the gradients of the Eq.(7) as

$$\frac{\partial p(j|\mathbf{x};\boldsymbol{\beta})}{\partial f_k(\mathbf{x})} = \frac{\mathbb{I}(k=j)e^{f_j(\mathbf{x})}\sum_{i=1}^{K} e^{f_i(\mathbf{x})} - e^{f_j(\mathbf{x})}e^{f_k(\mathbf{x})}}{(\sum_{i=1}^{K} e^{f_i(\mathbf{x})})^2} = \tag{16}$$
$$= \frac{e^{f_j(\mathbf{x})}}{\sum_{i=1}^{K} e^{f_i(\mathbf{x})}}(\mathbb{I}(k=j) - \frac{e^{f_k(\mathbf{x})}}{\sum_{i=1}^{K} e^{f_i(\mathbf{x})}}) = p(j|\mathbf{x};\boldsymbol{\beta})(\mathbb{I}(k=j) - p(k|\mathbf{x};\boldsymbol{\beta})).$$

Using this result, we compute the gradients of the prior regularization term as

$$\frac{\partial \jmath(\mathbf{x}, q;\boldsymbol{\beta})}{\partial f_k(\mathbf{x})} = p(k|\mathbf{x};\boldsymbol{\beta})\Big(\log\frac{p(k|\mathbf{x};\boldsymbol{\beta})}{m(k|\mathbf{x};\boldsymbol{\beta})} - \sum_{j\in\mathcal{Y}} p(j|\mathbf{x};\boldsymbol{\beta})\log\frac{p(j|\mathbf{x};\boldsymbol{\beta})}{m(j|\mathbf{x};\boldsymbol{\beta})}\Big)$$
$$= p(k|\mathbf{x};\boldsymbol{\beta})(\log\frac{p(k|\mathbf{x};\boldsymbol{\beta})}{m(k|\mathbf{x};\boldsymbol{\beta})} - D_{KL}(p\|m)). \tag{17}$$

Learning with Multi-Class Base Classifiers Given the gradients of the loss functions, the learning process of the t-th base classifier in Eq.(10) can be written as

$$\boldsymbol{\theta}_t^* = \arg\max_{\boldsymbol{\theta}_t} \; -\sum_{\mathbf{x}\in\mathcal{X}}\sum_{k\in\mathcal{Y}} \frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_k(\mathbf{x})} g_k(\mathbf{x};\boldsymbol{\theta}_t). \tag{18}$$

The solution of this problem will select a base function which has the highest correlation with the steepest descent direction of the risk. Since we want to use multi-class base learners, we have to develop a single label and a single weight for each sample. The following theorem shows the best possible choices for the weights and pseudo-labels.

Theorem 21 *The solution of Eq.*(18) *using a multi-class classifier* $c(\mathbf{x};\boldsymbol{\theta}_t) \in \mathcal{Y}$ *can be obtained by solving*

$$\boldsymbol{\theta}_t^* = \arg\min_{\boldsymbol{\theta}_t} \sum_{\mathbf{x}\in\mathcal{X}} w_{\mathbf{x}}\mathbb{I}(c(\mathbf{x}) \neq \hat{y}) \tag{19}$$

where

$$w_{\mathbf{x}} = \max_{k\in\mathcal{Y}} -\frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_k(\mathbf{x})} \text{ and } \hat{y}_{\mathbf{x}} = \arg\max_{k\in\mathcal{Y}} -\frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_k(\mathbf{x})} \tag{20}$$

are the weight and the pseudo-label for the sample $\mathbf{x}$*, respectively.*

Proof. The proof is similar to the one presented by Saffari *et al.* [17].

The following lemmas show that for labeled and unlabeled samples, the weight is positive or zero and the chosen label for the labeled samples is the true class label. Note that this is an important step in boosting, as the derived weights should always be positive for all the samples.

Lemma 1. *For the labeled samples, the pseudo-label given in Eq.*(20) *is the true class label and the sample weight is positive.*

Proof. From Eq. (13) and Eq. (14) we have that $\forall k \neq y : -\frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_k(\mathbf{x})} < 0$ and only for the target class $-\frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_y(\mathbf{x})} > 0$. This also means that the pseudo-label is the true class.

Lemma 2. *For the unlabeled samples, the sample weight given in Eq.*(20) *is positive or zero.*

Proof. First we show that the sum of the gradients for an unlabeled sample over different classes is always zero. Note that

$$-\sum_{k\in\mathcal{Y}} \frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_k(\mathbf{x})} = -\gamma \sum_{k\in\mathcal{Y}} p(k|\mathbf{x};\boldsymbol{\beta})(\log \frac{p(k|\mathbf{x};\boldsymbol{\beta})}{m(k|\mathbf{x};\boldsymbol{\beta})} - D_{KL}(p\|m)) =$$
$$=\gamma(D_{KL}(p\|m) - D_{KL}(p\|m)) = 0. \tag{21}$$

Since the sum of the negative of the gradients is zero, therefore, either all the gradients are equal to zero, or if there are some non-zero gradients, then their maximum over different classes is positive, as it is not possible that the sum of a set of negative terms is zero. Therefore,

$$w_{\mathbf{x}} = \max_{k\in\mathcal{Y}} -\frac{\partial R_{emp}(\boldsymbol{\beta}_{t-1})}{\partial f_k(\mathbf{x})} \geq 0. \tag{22}$$

3 Experiments

3.1 Multi-Class Boosting Experiments

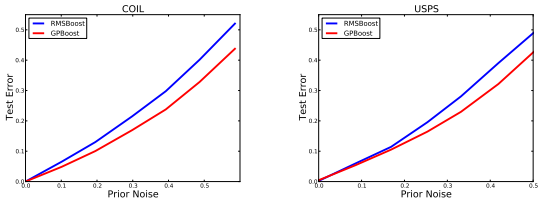
We compare the performance of the proposed multi-class boosting algorithm with other state-of-the-art methods on a set of multi-class machine learning benchmark datasets obtained from UCI repository. In these experiments, we compare with the following multi-class classifiers: Random Forests (RF) [19], three multi-class formulations of AdaBoost namely SAMME [20], AdaBoost.ECC [21], and the recent algorithm of AdaBoost.SIP [22][5]. As the last algorithm, we also compare with the multi-class support vector machine algorithm. For Random Forests, we train 250 randomized trees. For the SVM we use the RBF kernel and perform model selection by a grid search for selecting the kernel width σ and capacity parameter C. For our GBoost algorithm, we use 5 extremely randomized trees as weak learners, and set the number of weak learners $T = 50$ and fix the shrinkage factor to $\nu = 0.05$ for all the experiments. We repeat the experiments for 5 times and report the average test error.

The results over *DNA*, *Letter*, *Pendigit*, and *USPS* datasets are shown in Table 1. As it can be seen, our algorithm achieves results comparable to other multi-class classifiers. The best performing method is the SVM with RBF kernel. However, our algorithm achieves these results without any need for model

[5] For these algorithms we report the results presented in [22].

Dataset/Method	GBoost	RF	SVM	SAMME [22]	AdaBoost.ECC [22]	AdaBoost.SIP [22]
DNA	0.0582	0.0683	0.0559	0.1071	**0.0506**	*0.0548*
Letter	**0.0265**	0.0468	*0.0298*	0.4938	0.2367	0.1945
Pendigit	*0.0387*	0.0496	**0.0360**	0.3391	0.1029	0.0602
USPS	*0.0524*	0.0610	**0.0424**	N/A	N/A	N/A

Table 1. Classification error on machine learning benchmark datasets. The bold-face shows the best performing method, while the italic font shows the second best.



(a) COIL (b) USPS

Fig. 2. Classification error for (a) COIL and (b) USPS dataset with noisy priors.

selection (we use a fixed setting for all the experiments in this section and the next two sections), and is considerably faster during both training and testing. For example, for the Letter dataset with 15000 training and 4000 samples, our unoptimized Python/C++ implementation[6] finishes the training and testing in 54 seconds, while the training of the SVM using Shogun LibSVM interface [23] takes around 156 seconds.

3.2 Robustness Experiments

In order to show the increased robustness experimentally, we compare our semi-supervised boosting algorithm (GPBoost) with the RMSBoost [17] which uses the Kullback-Leibler divergence. In these experiments, we use the hidden labels of the unlabeled samples in order to produce a prior and then we introduce random label noise into these priors and train both semi-supervised boosting algorithms with the same settings. In order to make the comparison fair, we also change the supervised loss function of the RMSBoost to Savage loss. For these experiments, we choose two semi-supervised learning benchmark datasets [1]. The results averaged over 12 splits provided in the dataset for COIL and USPS datasets are shown in Figure 2. As it can be seen, our algorithm retains lower test errors compared to the RMSBoost. It should be noted that specially for the COIL set which is multi-class dataset, the gap is larger from early on.

[6] Code is available at `http://www.ymer.org/amir/software/multi-class-semi-supervised-boosting/`

3.3 Object Category Recognition

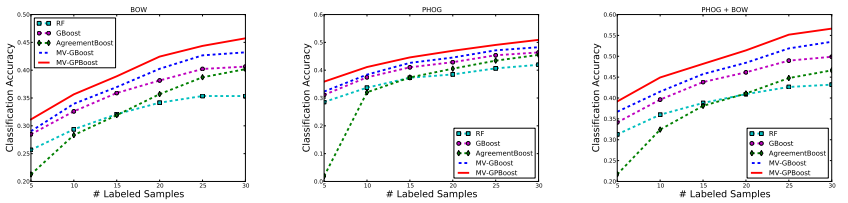
We evaluate various multi-view semi-supervised boosting algorithms on *Caltech101* object category recognition task. This dataset represent a challenging task for the semi-supervised learners, since the number of classes is large and the number of training samples per class is rather low. For these experiments, we randomly choose up to 80 images from each class and label $\{5, 10, 15, 20, 25, 30\}$ images for each of them. We use the rest of the samples as the test set. Since many of the classes do not have enough images to form a separate unlabeled set, we resort to the transductive settings where the test set is used as the unlabeled set. We repeat this procedure 5 times and report the average classification accuracy per each class.

For feature extraction, we use the precomputed dense SIFT-based bag-of-words and PHOG features from Gehler and Nowozin [24] to form different views. In details, for BOW features, we use a vocabulary of size 300 extracted from gray level and individual color channels. We use a level-2 spatial histogram to represent these 2 views (BOW-grey and BOW-Color). Additionally, we use level-2 PHOG features formed from the oriented (PHOG-360) and unoriented (PHOG-180) gradients. Therefore, in total we have 4 different views for this dataset.

In these experiments, we use the Random Forests (RF), our supervised multi-class boosting algorithm (GBoost), multi-view boosting using GBoost as the basic learners (MV-GBoost), and our multi-view algorithm MV-GPBoost. Additionally, we extended the AgreementBoost algorithm [6] to cope with multi-class problems and report the results for this algorithm as well.

If we set the $\gamma = 0$ in MV-GPBoost, we will end up in exactly the MV-GBoost algorithm. Therefore, the performance gains seen here are totally due to the incorporation of the prior regularization term. The settings for the RFs and our boosting algorithms is exactly the same settings used for machine learning benchmark experiments. For the multi-view algorithms, we iterate the learning process for 10 iterations and label 100 unlabeled samples (1 from each class) in each iteration. Since RFs and GBoost cannot use the views directly, we concatenate the features into a single feature vector.

Figure 3 shows the results for three different settings: (a) only using the two views provided from BOW features, (b) using two views from PHOG features, and (c) using all 4 views. The first observation is that the GBoost algorithm successfully boosts the performance of the random forest and the accuracy gap can be as high as 5%. Comparing the performance of the GBoost and the MV-GBoost, we can see that in general the multi-view learning strategy by labeling a subset of unlabeled samples iteratively, works and there is a clear performance gain between these two algorithms. However, the highest accuracy is obtained by MV-GPBoost which has a considerable gap in classification accuracy compared to the MV-GBoost algorithm. Another observation here is that, as expected, the combination of all 4 views achieves the highest performance, compared to using either two views from BOW or PHOGs. Furthermore, the performance of the AgreementBoost which uses the variance of the classifiers over different views to



(a) BOW views (b) PHOG views (c) PHOG and BOW views

Fig. 3. Caltech101 classification accuracy for: (a) BOW, (b) PHOG, and (c) BOW and PHOG.

Methods - Views	BOW		PHOG		BOW+PHOG	
# Labels	15	30	15	30	15	30
SVM	0.5545	0.6415	0.4612	0.5264	0.6123	0.6888
MV-GPBoost	0.5605	0.6805	0.4745	0.5411	0.6496	0.7158

Table 2. Classification accuracy on Caltech101 with kernel SVMs for 15 and 30 labeled samples per class.

regularize the training process of boosting is not comparable to the performance of other learning methods.

Similar to [24], when we use χ^2 kernels over each of the views and use SVMs as the weak learners of the boosting classifiers, we improve the classification accuracy on this dataset. These results are reported in Table 2.

3.4 Object Tracking

Recently, boosting-based methods have achieved high accurate tracking performances running in real-time [25]. In these methods, usually an appearance-based classifier is trained with a marked object at the first frame versus its local background. The object is then tracked by performing re-detection in the succeeding frames. In order to handle rapid appearance and illumination changes, the classifiers perform on-line self-updating [26]. However, during this self-updating process it is hard to decide where to select the positive and negative updates. If the samples are selected wrongly, slight errors can accumulate over time and cause drifting. Therefore, recent approaches applied on-line extensions of boosting that can handle the uncertainty in the update process, such as CoBoost [9], SemiBoost [27] or MILBoost [28]. The main idea of these approaches is to define a region around the current tracking position and leave it up to the learner which samples to incorporate as positives or negatives in order to stabilize the tracking. In the following, we compare our method to the state-of-the-art.

We use eight publicly available sequences including variations in illumination, pose, scale, rotation and appearance, and partial occlusions. The sequences *Sylvester* and *David* are taken from [29] and *Face Occlusion 1* is taken from [30],

Approach	*sylv*	*david*	*faceocc2*	*tiger1*	*tiger2*	*coke*	*faceocc1*	*girl*
MV-GPBoost	17	**20**	**10**	**15**	**16**	*20*	**12**	**15**
CoBoost	*15*	33	*11*	22	19	**14**	*13*	*17*
SemiBoost	22	59	43	46	53	85	41	52
MILBoost	**11**	*23*	20	**15**	*17*	21	27	32

Table 3. Tracking results on the benchmark sequences measured as average center location errors (in pixels) over 5 runs per sequence. Best performing method is marked in bold face, while the second best is shown in italic.

respectively. *Face occlusion 2*, *Girl*, *Tiger1,Tiger2* and *Coke* are taken from [28]. All video frames are gray scale and of size 320×240. We report the tracking accuracy in terms of average center location error in pixel to the groundtruth.

Since our method is a multi-view approach, it is straight-forward to use different feature information. However, this would make the comparison to other methods that are based on single features unfair. So, in the following we report tracking results only for Haar-features and it should be clear to the reader (also by looking at previous experiments) that further improvement can be achieved by adding additional feature queues. In particular, we use 30 selectors with each 30 weak learners. The different views are generated by random sub-sampling from a large amount of Haar-features. In Table 3 we depict the results for all tracking sequences, *i.e.*, CoBoost [9], SemiBoost [27] and MILBoost [28]. As can be seen, MV-GPBoost performs best on five tracking sequences. The resulting tracking videos can be found in the supplementary material.

4 Conclusions

In this paper, we have introduced a new multi-view boosting algorithm. In contrast to previous approaches that select a subset of the unlabeled data and ignore the rest, we use all unlabeled samples and, we use the different views to provide an aggregated prior which regularizes a semi-supervised loss function. Since priors are noisy, we also propose a novel robust loss function for semi-supervised boosting. Finally, our method is inherently multi-class and can handle more than two views at the same time. We demonstrated the performance of our method on machine learning benchmark sets, Caltech-101 object categorization and object tracking.

References

1. Chapelle, O., Schölkopf, B., Zien, A.: Semi-Supervised Learning. Cambridge, MA (2006)
2. Zhu, X.: Semi-supervised learning literature survey. Technical report (2008)
3. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: COLT. (1998) 92–100
4. Brefeld, U., Büscher, C., Scheffer, T.: Multi-view discriminative sequential learning. In: ECML. (2005) 60–71

5. Sindhwani, V., Rosenberg, D.S.: An rkhs for multi-view learning and manifold co-regularization. In: ICML. (2008) 976–983
6. Leskes, B., Torenvliet, L.: The value of agreement a new boosting algorithm. J. Comput. Syst. Sci. **74** (2008) 557–586
7. Levin, A., Viola, P., Freund, Y.: Unsupervised improvement of visual detectors using co-training. In: ICCV. Volume I. (2003) 626–633
8. Christoudias, C.M., Urtasun, R., Darrell, T.: Unsupervised distributed feature selection for multi-view object recognition. In: CVPR. (2008)
9. Liu, R., Cheng, J., Lu, H.: A robust boosting tracker with minimum error bound in a co-training framework. In: ICCV. (2009)
10. Tang, F., Brennan, S., Zhao, Q., Tao, H.: Co-tracking using semi-supervised support vector machines. In: ICCV. (2007)
11. Sun, S., Zhang, Q.: Multiple-view multiple-learner semi-supervised learning. Technical report (2007)
12. Leistner, C., Saffari, A., Santner, J., Bischof, H.: Semi-supervised random forests. In: IEEE International Conference on Computer Vision (ICCV). (2009)
13. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: ICML. (1996) 148–156
14. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. The Annals of Statistics **38** (2000) 337–374
15. Shirazi, H.M., Vasconcelos, N.: On the design of loss functions for classification: theory, robustness to outliers, and savageboost. In: NIPS. (2008) 1049–1056
16. Saffari, A., Grabner, H., Bischof, H.: Serboost: Semi-supervised boosting with expectation regularization. In: ECCV. (2008)
17. Saffari, A., Leistner, C., Bischof, H.: Regularized multi-class semi-supervised boosting. In: CVPR. (2009)
18. Friedman, J.: Greedy function approximation: A gradient boosting machine. The Annals of Statistics **29** (2001) 1189–1232
19. Breiman, L.: Random forests. Machine Learning **45** (2001) 5–32
20. Zhu, J., Rosset, S., Zou, H., Hastie, T.: Multi-class adaboost. Technical report (2006)
21. Guruswami, V., Sahai: Multiclass learning, boosting, and error-correcting codes. In: COLT. (1999)
22. Zhang, B., Ye, G., Wang, Y., Xu, J., Herman, G.: Finding shareable informative patterns and optimal coding matrix for multiclass boosting. In: ICCV. (2009)
23. Sonnenburg, S., Rätsch, G., Schäfer, C., Schölkopf, B.: Large scale multiple kernel learning. JMLR **7** (2006) 1531–1565
24. Gehler, P., Nowozin, S.: On feature combination for multiclass object classication. In: ICCV. (2009)
25. Avidan, S.: Ensemble tracking. Volume 2. (2005) 494–501
26. Grabner, H., Bischof, H.: On-line boosting and vision. Volume 1. (2006) 260–267
27. Grabner, H., Leistner, C., Bischof, H.: On-line semi-supervised boosting for robust tracking. In: ECCV. (2008)
28. Babenko, B., Yang, M.H., Belongie, S.: Visual tracking with online multiple instance learning. In: CVPR. (2009)
29. Ross, D., Lim, J., Lin, R.S., Yang, M.H.: Incremental learning for robust visual tracking. IJCV (2008)
30. Adam, A., Rivlin, E., Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In: CVPR. (2006)