Online Multi-Class LPBoost*

Amir Saffari Martin Godec Thomas Pock Christian Leistner
Horst Bischof
Institute for Computer Graphics and Vision, Graz University of Technology, Austria
{saffari,godec,pock,leistner,bischof}@icg.tugraz.at

Abstract

Online boosting is one of the most successful online learning algorithms in computer vision. While many challenging online learning problems are inherently multi-class, online boosting and its variants are only able to solve binary tasks. In this paper, we present Online Multi-Class LPBoost (OMCLP) which is directly applicable to multi-class problems. From a theoretical point of view, our algorithm tries to maximize the multi-class soft-margin of the samples. In order to solve the LP problem in online settings, we perform an efficient variant of online convex programming, which is based on primal-dual gradient descent-ascent update strategies. We conduct an extensive set of experiments over machine learning benchmark datasets, as well as, on Caltech101 category recognition dataset. We show that our method is able to outperform other online multi-class methods. We also apply our method to tracking where, we present an intuitive way to convert the binary tracking by detection problem to a multi-class problem where background patterns which are similar to the target class, become virtual classes. Applying our novel model, we outperform or achieve the state-of-the-art results on benchmark tracking videos.

1. Introduction

Online learning is an area of machine learning concerned with estimation problems with limited access to the entire data domain. It is a sequential decision making task where the objectives for the learner are revealed over time. Classical online learning problems can be formulated as a game between the learner and an adversary environment (or teacher). In this repeated game, at any time t the following steps happen: 1) The environment chooses a new sample $\mathbf{x}_t \in \mathbb{R}^d$. 2) The learner responds with its prediction $\hat{y}_t$. 3) The environment reveals the label of the sample y_t. 4) The learner suffers the loss ℓ_t and updates its model. The goal of the learner is to achieve a low cumulative loss over time by updating its internal representation of the problem.

Online learning is an essential tool for learning from dynamic environments, from very large scale datasets, or from streaming data sources. It has been studied extensively in the machine learning community (for a comprehensive overview we refer to [5, 25] and references therein). In computer vision, online learning has been used in applications such as object recognition [14, 4], object detection [22, 30] and tracking [8, 16, 13].

Historically, Oza and Russell [21] were the first to extend the AdaBoost [11] to operate in an online learning scenario. Their formulation and many variants have been used in various computer vision applications [16, 13, 22, 30, 18].

Almost all recent work on online boosting algorithms focus on binary decision problems, while many interesting problems are inherently multi-class. These algorithms tackle the multi-class problems by using a set of decomposed binary tasks, usually obtained by typical approaches like 1-vs.-all, 1-vs.-1, and error correcting output codes [2]. However, such approaches have major drawbacks. First, by considering only the binary sub-problems, the algorithms often fail to completely capture the true structures and relations between the classes in the feature space. In online learning tasks, this problem is even more severe because the learner only has access to a limited amount of data. Second, one has to train at least

*This work has been supported by the Austrian FFG project EVis (813399) and Outlier (820923) under the FIT-IT program and the Austrian Science Fund (FWF) under the doctoral program Confluence of Vision and Graphics W1209.

a number of classifiers equivalent to the number of classes. For problems with a large number of classes, such approaches have computational disadvantages. For an online learning scenario, because of these constraints, such approaches might not be applicable. Third, the commonly used 1-vs.-all approach introduces additional problems, such as producing unbalanced datasets or uncalibrated classifiers.

Boosting with convex loss functions is proven to be sensitive to outliers and label noise [19]. This inherent problem of boosting is even more important in online learning problems where the label given by the environment might be quite noisy. Hence, training such a sensitive algorithm in noisy environments usually leads to inferior classifiers. Recently, there has been a great effort to remedy this weakness, *e.g.* by introducing more robust loss functions [19, 27, 18]. There exists theoretical evidences that many boosting algorithms are only able to maximize the *hard-margin* [23] or *average margin* [26] of data samples. Such problems are addressed in other learning methods, specially in support vector machines, by introducing the *soft-margins*. Fortunately, for offline boosting, there exist a few methods which are able to use the soft-margins, notably, the Linear Programming Boosting (LPBoost) [9] and its variants [28, 29, 12].

Therefore, by formulating the LPBoost for online multi-class learning problems, we are able to directly address these inherent weaknesses of online boosting methods. Experimentally, we show that our algorithm in fact holds to these promises and is able to outperform or at least achieve state-of-the-art results compared to other online multi-class learning methods on various pattern recognition and computer vision tasks. We conduct a set of experiments to compare our method with other online and offline multi-class learning methods on standard machine learning benchmarks. Additionally, we apply our method to the object category classification problem on Caltech101 dataset. As a side effect of having an online multi-class classifier, we are able to perform multi-target tracking efficiently and robustly. For single object tracking with a complex background, we propose to formulate the problem as a multi-target tracking by assigning separate *virtual* classes for non-target objects with high similarities to the target, and hence, improve the tracking results to achieve state-of-the-art over benchmark videos.

2. Online Multi-Class LPBoost

In this section, we formulate the online multi-class boosting as a linear programming optimization problem. We first state the problem and then present our learning method which is based on a primal-dual gradient descent-ascent strategy.

In online learning scenarios, the data samples are presented sequentially. Following the repeated game analogy of online learning, the goal of the learner is to achieve a low cumulative loss over time by updating its model incrementally. Let the loss at iteration t be ℓ_t, which measures how bad was the prediction of the learner $\hat{y}_t$ with respect to the true class label of the newest sample y_t. In our formulation, we assume that the number of classes is not known in advance, and the learner should be able to incorporate new classes on-the-fly. Since the classes are presented over time, we do not penalize the learner when a new class is introduced.

Let $\mathcal{C}_t \subseteq \mathcal{C}$ be the set of known classes up to time t and $\mathcal{C}$ be the total label set (unknown to the learner). Also let $K_t = |\mathcal{C}_t|$ be the number of known classes at time t. In our formulation, the learner maintains a model $f_t : \mathbb{R}^d \rightarrow \mathbb{R}^{K_t}$ which is a mapping from the input feature space to the multi-class hypothesis domain. We represent the confidence of the learner for the k-th class $f_{t,k}(\mathbf{x}_t)$ as the k-th element of the output vector $f_t(\mathbf{x}_t) = [f_{t,1}(\mathbf{x}_t), \ldots, f_{t,K_t}(\mathbf{x}_t)]^T$. The following decision rule is applied in order to obtain the classification

$$\hat{y}_t = \underset{k \in \mathcal{C}_t}{\arg\max}\ f_{t,k}(\mathbf{x}_t). \tag{1}$$

We define the *hard margin* of a sample $\mathbf{x}_t$ as

$$m_{y_t}(\mathbf{x}_t) = f_{t,y_t}(\mathbf{x}_t) - \max_{\substack{k \in \mathcal{C}_t \\ k \neq y_t}} f_{t,k}(\mathbf{x}_t), \tag{2}$$

which measures the difference between the classification confidence of the true class and the closest non-target class

$$y'_t = \underset{\substack{k \in \mathcal{C}_t \\ k \neq y_t}}{\arg\max}\ f_{t,k}(\mathbf{x}_t).$$

Note that based on the decision rule of Eq (1), $m_{y_t}(\mathbf{x}_t) < 0$ means a wrong prediction, while a positive margin means a correct classification.

In this work, we use the *hinge* loss function

$$\ell_t(m_{y_t}) = \mathbb{I}(y_t \in \mathcal{C}_t) \max(0, 1 - m_{y_t}(\mathbf{x}_t)), \tag{3}$$

where $\mathbb{I}(\cdot)$ is an indicator function, which is introduced so that we do not penalize the model if there

is a novel class introduced. Note that hinge loss is an upper bound on the miss-classification error

$$\sum_{t=1}^{T} \mathbb{I}(y_t \neq \hat{y}_t) \leq \sum_{t=1}^{T} \ell_t(m_{y_t}), \tag{4}$$

and hence, its minimization results in minimizing the miss-classification error rate.

2.1. Multi-Class LPBoost Model

Our learner is a boosting model, *i.e.* a linear combination of some weak learners (or bases)

$$f_t(\mathbf{x}_t) = \sum_{m=1}^{M} w_{t,m} g_{t,m}(\mathbf{x}_t), \tag{5}$$

where $g_{t,m} : \mathbb{R}^d \rightarrow \mathbb{R}^{K_t}$ is the m-th weak learner, M represents the number of weak learners, and $w_{t,m}$ is the weight of m-th base. It is convenient to write this formulation in a more compact form as

$$f_t(\mathbf{x}_t) = G_t(\mathbf{x}_t)\mathbf{w}_t, \tag{6}$$

where $\mathbf{w}_t = [w_{t,1}, \dots, w_{t,M}]^T \in \mathbb{R}^M$ is the weight vector of all bases and

$$G_t(\mathbf{x}_t) = [g_{t,1}(\mathbf{x}_t), \dots, g_{t,M}(\mathbf{x}_t)] \in \mathbb{R}^{K_t} \times \mathbb{R}^M \tag{7}$$

is the response matrix of all weak learners for all the known classes. We denote $G_t(y, \cdot)$ to be the y-th row of this matrix, and $G_t(y, m)$ to be the element in the y-th row and the m-th column.

Offline boosting sequentially adds base learners to the whole model. However, in our online boosting formulation, the model utilizes a fixed set of online base learners and updates them sequentially by adjusting the weight of a sample.

Let $\mathcal{B}_{\Delta T}$ be a cache with size ΔT. A cache of size $\Delta T = 1$ will correspond to the case that learner discards the sample after updating on it. Considering our boosting model and the loss function presented in Eq (3), we propose the following regularized multi-class LPBoost problem to be optimized online

$$\begin{aligned}
&\min_{\mathbf{w}_T, \boldsymbol{\xi}} C \sum_{t \in \mathcal{B}_{\Delta T}} \sum_{k \neq y_t} \xi_{t,k} + \|\mathbf{w}_T\|_1 \qquad (8)\\
&\text{s.t. } \forall m : w_{T,m} \geq 0\\
&\forall t, \forall k \neq y_t : \xi_{t,k} \geq 0\\
&\forall t, \forall k \neq y_t : \big(G_t(y_t, \cdot) - G_t(k, \cdot)\big)\mathbf{w}_T + \xi_{t,k} \geq 1
\end{aligned}$$

where C is the capacity parameter, and slack variables $\xi_{t,k}$ are added to create *soft margins* for boosting. Note that this formulation is a direct generalization of the original formulation of LPBoost [9] to the multi-class case. Furthermore, if we would share the slack between all the classes, then it would be closely related to the multi-class variant of ν-LPBoost proposed in [12]. In an offline scenario, such problems can be easily solved by standard optimization techniques. However, in the online setting, usually it is infeasible to solve this problem from scratch for every new sample added to the system. Therefore, an incremental solution is desired. Fortunately, due to convexity of the problem, one can benefit from previously proposed *online convex programming* approaches [33].

2.2. Online Learning

Our online learning method performs primal-dual gradient descent-ascent iteratively. In detail, we first convert the problem to its *augmented Lagrangian* form [20]. By each new sample, we first perform a dual ascent, which is equivalent of finding sample weights for each iteration of training weak learners. After finishing that step, we do a primal descent over the weights of weak learners.

The Lagrange dual function of optimization problem Eq (8) is

$$\begin{aligned}
D(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{d}) = &\sum_{t \in \mathcal{B}_{\Delta T}} \sum_{k \neq y_t} d_{t,k} + \qquad (9)\\
&+ \inf_{\mathbf{w}_T, \boldsymbol{\xi}} \Big(\sum_{m=1}^{M} (1 - \alpha_m) w_{T,m} +\\
&+ \sum_{t \in \mathcal{B}_{\Delta T}} \sum_{k \neq y_t} (C - d_{t,k} - \beta_{t,k}) \xi_{t,k} -\\
&- \sum_{t \in \mathcal{B}_{\Delta T}} \sum_{k \neq y_t} d_{t,k} \big(G_t(y_t, \cdot) - G_t(k, \cdot)\big)\mathbf{w}_T \Big),
\end{aligned}$$

where $\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{d}$ are the Lagrange multipliers of the constraints. Due to linearity of the inner problem of Eq (9), for a set of finite solutions the following conditions must hold

$$\begin{aligned}
&\forall t, \forall k \neq y_t : C - d_{t,k} - \beta_{t,k} = 0\\
&\forall m : 1 - \alpha_m - \sum_{t \in \mathcal{B}_{\Delta T}} \sum_{k \neq y_t} d_{t,k} \Delta G_{t,k}(m) = 0,
\end{aligned}$$

where $\Delta G_{t,k}(m) = G_t(y_t, m) - G_t(k, m)$. Using the positivity conditions on Lagrange multipliers, we

can derive the dual formulation of Eq (8) as

$$\max_{\mathbf{d}} \sum_{t\in\mathcal{B}_{\Delta T}} \sum_{k\neq y_t} d_{t,k} \tag{10}$$
$$\text{s.t. } \forall m : \sum_{t\in\mathcal{B}_{\Delta T}} \sum_{k\neq y_t} d_{t,k}\Delta G_{t,k}(m) \leq 1$$
$$\forall t, \forall k \neq y_t : 0 \leq d_{t,k} \leq C.$$

The vector $\mathbf{d}$, which corresponds to sample weights, is the dual variable of weights on weak learners. The first set of constraints are equivalent of *edge* constraints of binary LPBoost. As it can be seen, there are $K-1$ weights per each sample. However, the weak learners usually accept only one weight per instance. Therefore, we only consider the most violating edge constraint for each example. This corresponds to finding the non-target class for which the margin is the smallest: y'_t [1]. Therefore, we will concentrate on the following problem

$$\max_{\mathbf{d}} \sum_{t\in\mathcal{B}_{\Delta T}} d_{t,y'_t} \tag{11}$$
$$\text{s.t. } \forall m : \sum_{t\in\mathcal{B}_{\Delta T}} d_{t,y'_t}\Delta G_{t,y'_t}(m) \leq 1$$
$$\forall t : 0 \leq d_{t,y'_t} \leq C.$$

Optimizing the problem in Eq. (11) with an online convex programming technique [33], requires a projection step for finding solutions which are consistent with the constraints. However, in our case such a step is expensive to compute; therefore, we formulate its augmented Lagrangian [20] as

$$\max_{\mathbf{w}_T,\mathbf{d}} \sum_{t\in\mathcal{B}_{\Delta T}} d_{t,y'_t} + \tag{12}$$
$$+ \sum_{m=1}^{M} w_{T,m}(1 - \sum_{t\in\mathcal{B}_{\Delta T}} d_{t,y'_t}\Delta G_{t,y'_t}(m) - \zeta_m) -$$
$$- \frac{1}{2\theta}\sum_{m=1}^{M}(1 - \sum_{t\in\mathcal{B}_{\Delta T}} d_{t,y'_t}\Delta G_{t,y'_t}(m) - \zeta_m)^2$$
$$\text{s.t. } \forall m : \zeta_m \geq 0, w_{T,m} \geq 0$$
$$\forall t : 0 \leq d_{t,y'_t} \leq C,$$

by introducing a new set of slack variables ζ_m and using $\theta > 0$ as a constant. Note that the value of slacks can be easily found by computing the derivatives of the objective with respect to them and setting it to zero. This leads to

$$\zeta_m = \max(0, q_m),$$

where $q_m = 1 - \sum_{t\in\mathcal{B}_{\Delta T}} d_{t,y'_t}\Delta G_{t,y'_t}(m) - \theta w_{T,m}$.

Now we follow this procedure over time: when a new sample arrives, we set its weight to C and update the cache by removing the oldest sample and inserting the newest. Then, for training the m-th weak learner, we compute the sample weights by dual gradient ascent update

$$\forall t : \; e_t = d_{t,y'_t} + \nu_d\Big(1 + \frac{1}{\theta}\sum_{\substack{j=1\\ q_j<0}}^{m-1} q_j\Delta G_{t,y'_t}(j)\Big)$$
$$d_{t,y'_t} \leftarrow \max(0, \min(C, e_t)), \tag{13}$$

where ν_d is the dual learning rate. After updating the sample weight and training the m-th weak learner according to them, we compute an update for the weight of this weak learner by a primal gradient descent update

$$\forall m : \; z_m = w_{T,m} - \nu_p\Big(1 - \sum_{t\in\mathcal{B}_{\Delta T}} d_{t,y'_t}\Delta G_{t,y'_t}(m)\Big)$$
$$w_{T,m} \leftarrow \max(0, z_m), \tag{14}$$

where ν_p is the learning rate for the primal. This alternating primal-dual descent-ascent is continued for all the weak learners.

Discussion Although the update rules presented in Eq (13) and Eq (14) look complicated, in fact, they present intuitive learning strategies which are closely related to the boosting way of learning from data. In Eq (13), the inner sum shows the total confidence of the weak learners trained so far with respect to the classification margin of the current sample. Note that since $q_j < 0$, for a sample which many of the weak learners obtain a positive margin, this sum will be a large negative value. Hence, for such a sample with large positive margin, the weight will decrease for the training of the next weak learner. Similarly, for the update in Eq (14), if a weak learner has a high weighted average margin over all the samples in the cache, the inner sum will be high, which will lead to an increase in its weight. Therefore, the weight of successful weak learners will increase.

3. Experiments

We evaluate the proposed Online Multi-Class LPBoost (OMCLP) algorithm by comparing its performance to other online learning algorithms. In the first

[1] Note that this is a limitation imposed by the weak learners. The following derivations can be easily generalized for all the weights.

two sets of experiments, we mainly compare with other multi-class online and offline algorithms, while in last section we will conduct tracking experiments.

3.1. Machine Learning Benchmark

Since there is no other online multi-class boosting algorithm available in literature for comparison, we convert the recently proposed offline multi-class boosting algorithm of Zou *et al.* [34] to online formulation. Based on their formulation, we define a margin vector based on the current classifier as

$$\forall \mathbf{x}_t : \sum_{i=1}^{K} f_{t,i}(\mathbf{x}_t) = 0. \quad (15)$$

We then use a Fisher consistent convex loss function [34], which guarantees that by training over a large number of samples the boosting model is able to recover the unknown Bayes decision rule. For this work, we experiment with two different loss functions: the exponential loss $e^{-f_{t,y_t}(\mathbf{x}_t)}$ and the Logit loss $\log(1+e^{-f_{t,y_t}(\mathbf{x}_t)})$. For updating the m-th weak learner, we perform a functional gradient descent as

$$g_{t,m}(\mathbf{x}) = \arg\max_{g} \nabla\ell(f_{t,y_t}^{m-1}(\mathbf{x}_t))g_{y_t}(\mathbf{x}_t), \quad (16)$$

where $\nabla\ell(f_{t,y_t}^{m-1}(\mathbf{x}_t))$ is the gradient of the loss function at the m-th stage of boosting. As it will be shown later, in principle, this is also a novel and successful online multi-class boosting algorithm. We call this algorithm Online Multi-Class Gradient Boost (OMCGB).

Additionally, we compare with Online Random Forests (ORF) [24] and the highly successful online multi-class support vector machine algorithm of Bordes *et al.* [6] named Linear LaRank. Note that both of these algorithms are inherently multi-class, so they provide a fair comparison. We also performed experiments with the online AdaBoost formulation of Oza *et al.* [21] by training 1-vs-all classifiers. However, its performance was not comparable to these baseline methods; therefore, due to lack of space we omit reporting them. We also compare our method with the following offline trained multi-class classifiers: Random Forests [7], three multi-class formulations of AdaBoost namely SAMME [32], AdaBoost.ECC [15], and the recent algorithm of AdaBoost.SIP [31][2]. We also compare with the multi-class support vector machine algorithms of Keerthi *et al.* [17] with a linear kernel and the multi-class SVM from LibSVM with RBF kernel.

The OMCLP and OMCGB use small ORFs with 10 trees as their weak learners and we set $M = 10$. ORF when used as a single model uses 100 trees trained online. For our algorithm, we fix the cache size to 1 and set $\nu_d = \theta = 2$, $\nu_p = 1e^{-6}$, and $C = 5$. Note that these set of parameters will be used for all the datasets in this section and the next section. For offline methods, we use a 5-fold cross-validation to obtain their hyperparameters.

Table 1 shows the classification error on 4 benchmark datasets chosen from the UCI repository. All the experiments are run for 5 independent runs and the results are the average classification error on the held out test set. In order to simulate large scale datasets, we conduct the experiments in different number of epochs: each epoch corresponds to seeing all the data points once in random order. As it can be seen, our algorithm outperforms other online learning methods in 6 out of 8 cases and comes very close to the performance of the best offline methods. Another interesting observation is the fact that the performance of the OMCLP at the first epoch is very close to the performance of the ORF after 10 epochs. This shows that given a slow converging algorithm like ORF, we are able to speed up its convergence rate as well. Our C++ implementation of OMCLP and OMCGB is freely available from the following link[3].

3.2. Object Category Recognition

Online multi-class learning is essential when dealing with large-scale image databases. For example, typical image or video search engines often need to update their internal model when a set of new data is available. However, rebuilding the entire model is infeasible in practice. Considering the fact that the problem of object category recognition is inherently multi-class, therefore such systems can benefit from an online multi-class learner.

We evaluate on *Caltech101* object category recognition task, which is a challenging task for an online learner, since the number of classes is large and the number of training samples per class is small. For these experiments, we use the Linear LaRank as the weak learners of the online boosting methods, due to the fact that ORFs were performing poorly on this task. We convert the SVM scores of LaRank to prob-

[2]For these algorithms we report the results presented in [31].

[3]http://www.ymer.org/amir/software/online-multiclass-lpboost/

Methods - Dataset	DNA		Letter		Pendigit		USPS	
# Epochs	1	10	1	10	1	10	1	10
OMCLP	0.0983	**0.0565**	**0.1202**	**0.0362**	**0.0747**	**0.0241**	0.1185	**0.0809**
OMCGB-Log	0.2648	0.0777	0.3033	0.1202	0.1666	0.0599	0.2418	0.1241
OMCGB-Exp	0.1395	0.0616	0.2484	0.0853	0.1282	0.0501	0.1926	0.1103
ORF	0.2243	0.0786	0.2696	0.0871	0.1343	0.0464	0.2066	0.1085
LaRank	**0.0944**	0.0818	0.5656	0.5128	0.1712	0.2109	**0.0964**	0.1004
RF	0.0683		0.0468		0.0387		0.0610	
MCSVM-Lin	0.0727		0.2575		0.1266		0.0863	
MCSVM-RBF	0.0559		*0.0298*		*0.0360*		*0.0424*	
SAMME [31]	0.1071		0.4938		0.3391		N/A	
AdaBoost.ECC [31]	*0.0506*		0.2367		0.1029		N/A	
AdaBoost.SIP [31]	0.0548		0.1945		0.0602		N/A	

Table 1. Classification error on machine learning benchmark datasets for 1 and 10 epochs. The bold-face shows the best performing *online* method, while the italic-font shows the best *offline* method.

# Train	15		30	
# Epochs	1	10	1	10
OMCLP	**0.7437**	**0.6093**	**0.6672**	**0.5406**
OMCGB	0.7520	0.6226	0.6860	0.5693
ORF	0.8969	0.8265	0.8880	0.8142
LaRank	0.7856	0.6353	0.7205	0.5803

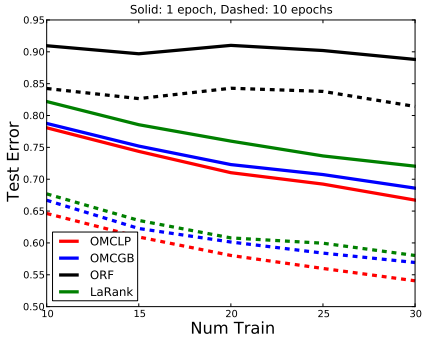
Table 2. Classification error on Caltech101 datasets for 1 and 10 epochs. The bold-face shows the best performing *online* method.

abilities via a multinomial logistic regression. All other settings are the same as experiments presented in Section 3.1.

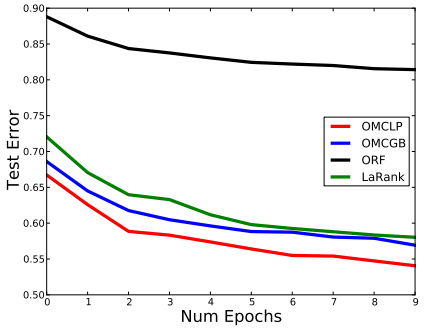
We present the results using the standard Caltech101 settings of training on 15 and 30, and testing on 50 images per class. For feature extraction, we use the precomputed 360 degree Level2-PHOG features from Gehler and Nowozin [12]. Table 2 shows the results obtained by various online methods, averaged over 5 independent runs on 5 different train and test splits (total 25 runs per algorithm)[4]. As it can be seen, our algorithm performs the best compared to other methods on this difficult task.

Figure 1(a) shows how the performance varies by the number of training images per category. As expected, more training samples help algorithms to improve over time. However, it is notable that our method consistently obtains lower errors compared to other algorithms over different amount of labeled data. Figure 1(b) shows the dynamics of the learners when the same training data is reshuffled and presented as new samples. We can see that although all methods benefit from revisiting the samples, our algorithms makes the most out of the epochs, and as

[4]We only report the performance of OMCGB-Exp as with Logit loss the results were similar.



(a)



(b)

Figure 1. (a) Classification error on Caltech101 for 1 epoch (solid) and 10 epochs runs when the number of training images per class is varying. (b) Classification error over different number of epochs when using 30 images per category for training.

can be seen towards the end of the 10-th epoch, it has the highest gap to the second best algorithm, OMCGB.

(a) Addition of a virtual class (b) No negative update (c) Updating a virtual class

Figure 2. Tracking with virtual background classes.

3.3. Tracking with Virtual Classes

Object Tracking is a common application for online learning algorithms in computer vision. Within this section, we will show the performance of our algorithm in a *tracking by detection* scenario. When training a discriminative object detector, the problem is usually formulated as binary classification. In tracking, we have usually fast changing, cluttered, complex background, which has to be described with a single background model. However, our approach is to break this binary task into a multi-class problem and utilize our robust online learner to discriminate between these classes.

From the classification margin point of view, the background might have samples which are very close to the decision boundaries of the target object. These samples are usually potential false positive detections during the tracking, specially when there are fast appearance changes or occlusions. Since, we know that our classifier can maximize the soft-margin of data instances, we sample densely from the decision boundaries of the target class in the feature space for potential false positive background regions. Then, each of these background regions is assigned to a separate *virtual* class. Hence, our online multi-class classifier will maximize its margin with respect to all these classes, while also in image domain it will keep tracking them as they were indeed target objects. Since our learner is able to accommodate new classes on-the-fly, we can keep track of any new object entering the scene which might cause possible confusions. Figure 2 shows this procedure in action: Figure 2(a) depicts the addition of a virtual background class, and Figure 2(c) indicates the update of an existing virtual class[5].

We conduct an extensive evaluation of our tracking method. Since we want to show that the performance gain comes from our robust multi-class classifier and from the addition of novel virtual background classes, we only use simple Haar-like Features without any post-processing. For the evaluation of our tracker we use the detection-criterion of the VOC Challenge [10], which is defined as $|R_T \cap R_{GT}|/|R_T \cup R_{GT}|$, where R_T is the tracking rectangle and R_{GT} the ground-truth. The advantage of this score is that it truly shows how accurate is the detection of the model, rather than computing the raw distance measure between the target and background. We measure the accuracy of a tracker by computing the average detection score for the entire video. We run each tracker 5 times and report the median average score. Table 3 lists the results for several publicly available benchmark sequences in comparison to other state-of-the-art tracking methods: MILTracker [3], FragTracker [1], and AdaBoostTracker [13]. In 5 out of 8 videos we outperform other methods, while for the remaining 3 videos we are the second best method. Our unoptimized C++ implementation of OMCLP algorithm reaches near real-time performance (around 10 to 15 frames/second on average).

Sequence	OMCLP	MIL [3]	Frag [1]	OAB [13]
Sylvester	**0.67**	0.60	*0.62*	0.520
Face 1	*0.80*	0.60	**0.88**	0.48
Face 2	**0.78**	*0.68*	0.44	*0.68*
Girl	**0.64**	0.53	*0.60*	0.40
Tiger 1	**0.53**	*0.52*	0.19	0.23
Tiger 2	*0.44*	**0.53**	0.15	0.28
David	**0.61**	*0.57*	0.43	0.26
Coke	*0.24*	**0.33**	0.08	0.17

Table 3. Average detection score: bold-face shows the best method, while italic-font indicates the second best.

4. Conclusions

In this paper, we presented the Online Multi-Class LPBoost algorithm, which is able to build in an online setting a robust multi-class boosting model with maximizing the soft-margin of the data samples. We solved the optimization problem by performing a variant of online convex programming technique, based on primal-dual gradient descent-ascent strategy. Based on an extensive set of experiments, we showed that our method outperforms the state-of-the-art on a wide range of applications, such as pattern recognition tasks, object category recognition tasks, and object tracking. Our C++ implementation is freely available online.

Our optimization technique was built on the well-known online convex programming technique which has efficient regret bounds. Therefore, we expect that similar bounds hold for our method as well. We will

[5]Please refer to supplementary material for videos describing our tracking method in details and its results.

present these results in an extended version of this paper.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, 2006. 7

[2] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, December 2000. 1

[3] B. Babenko, M. H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*, 2009. 7

[4] H. Bekel, I. Bax, G. Heidemann, and H. Ritter. Adaptive computer vision: Online learning for object recognition. In *DAGM*, pages 447–454, 2004. 1

[5] A. Blum. On-line algorithms in machine learning. In *Online Algorithms*, pages 306–325, 1996. 1

[6] A. Bordes, L. Bottou, P. Gallinari, and J. Weston. Solving multiclass support vector machines with larank. In *ICML*, 2007. 5

[7] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, October 2001. 5

[8] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *PAMI*, 27:1631–1643, 2005. 1

[9] A. Demiriz, K. P. Bennett, and J. S. Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1-3):225–254, 2002. 2, 3

[10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge 2007 (voc2007) results. 7

[11] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning (ICML)*, pages 148–156, 1996. 1

[12] P. Gehler and S. Nowozin. On feature combination for multiclass object classication. In *ICCV*, 2009. 2, 3, 6

[13] H. Grabner and H. Bischof. On-line boosting and vision. In *CVPR*, pages 260–267, 2006. 1, 7

[14] E. Granger, Y. Savaria, and P. Lavoie. A pattern reordering approach based on ambiguity detection for online category learning. *PAMI*, 25:525–529, 2003. 1

[15] V. Guruswami and Sahai. Multiclass learning, boosting, and error-correcting codes. In *COLT*, 1999. 5

[16] O. Javed, S. Ali, and M. Shah. Online detection and classification of moving objects using progressively improving detectors. In *CVPR*, pages 695–700, 2005. 1

[17] S. S. Keerthi, S. Sundararajan, K. W. Chang, C. J. Hsieh, and C. J. Lin. A sequential dual coordinate method for large-scale multi-class linear svms. In *KDD*, 2008. 5

[18] C. Leistner, A. Saffari, P. M. Roth, and H. Bischof. On robustness of on-line boosting - a competitive study. In *3rd IEEE ICCV Workshop on On-line Computer Vision*, 2009. 1, 2

[19] P. M. Long and R. A. Servedio. Random classification noise defeats all convex potential boosters. In *ICML*, volume 307, pages 608–615, 2008. 2

[20] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, April 2000. 3, 4

[21] N. Oza and S. Russell. Online bagging and boosting. In *AISTAT*, pages 105–112, 2001. 1, 5

[22] M. T. Pham and T. J. Cham. Online asymetric boosted clasifiers for object detection. In *CVPR*, 2007. 1

[23] G. Rätsch, T. Onoda, and K. R. Müller. Soft margins for adaboost. *Machine Learning*, 42(3):287–320, March 2001. 2

[24] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. On-line random forests. In *3rd IEEE ICCV Workshop on On-line Computer Vision*, 2009. 5

[25] S. Shalev-Shwartz. *Online Learning: Theory, Algorithms, and Applications*. PhD thesis, The Hebrew University of Jerusalem, July 2007. 1

[26] C. Shen and H. Li. On the dual formulation of boosting algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Dec 2010. 2

[27] H. M. Shirazi and N. Vasconcelos. On the design of loss functions for classification: theory, robustness to outliers, and savageboost. In *NIPS*, pages 1049–1056, 2008. 2

[28] M. K. Warmuth, K. Glocer, and G. Rätsch. Boosting algorithms for maximizing the soft margin. In *NIPS*, 2007. 2

[29] M. K. Warmuth, K. A. Glocer, and S. V. Vishwanathan. Entropy regularized lpboost. In *ALT*, pages 256–271, Berlin, Heidelberg, 2008. Springer-Verlag. 2

[30] B. Wu and R. Nevatia. Improving part based object detection by unsupervised, online boosting. In *CVPR*, 2007. 1

[31] B. Zhang, G. Ye, Y. Wang, J. Xu, and G. Herman. Finding shareable informative patterns and optimal coding matrix for multiclass boosting. In *ICCV*, 2009. 5, 6

[32] J. Zhu, S. Rosset, H. Zou, and T. Hastie. Multi-class adaboost. Technical report, 2006. 5

[33] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003. 3, 4

[34] H. Zou, J. Zhu, and T. Hastie. New multi-category boosting algorithms based on multi-category fisher-consistent losses. *Annals of Applied Statistics*, 2008. 5