Boosting with Online Binary Learners for the Multiclass Bandit Problem

Shang-Tse Chen SCHEN351@GATECH.EDU
School of Computer Science, Georgia Institute of Technology, Atlanta, GA

Hsuan-Tien Lin HTLIN@CSIE.NTU.EDU.TW
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan

Chi-Jen Lu CJLU@IIS.SINICA.EDU.TW
Institute of Information Science, Academia Sinica, Taipei, Taiwan

Abstract

We consider the problem of online multiclass prediction in the bandit setting. Compared with the full-information setting, in which the learner can receive the true label as feedback after making each prediction, the bandit setting assumes that the learner can only know the correctness of the predicted label. Because the bandit setting is more restricted, it is difficult to design good bandit learners and currently there are not many bandit learners. In this paper, we propose an approach that systematically converts existing online binary classifiers to promising bandit learners with strong theoretical guarantee. The approach matches the idea of boosting, which has been shown to be powerful for batch learning as well as online learning. In particular, we establish the weak-learning condition on the online binary classifier, and show that the condition allows automatically constructing a bandit learner with arbitrary strength by combining several of those classifiers. Experimental results on several real-world data sets demonstrate the effectiveness of the proposed approach.

1. Introduction

Recently, machine learning problems that involve partial feedback have received an increasing amount of attention (Auer et al., 2002; Flaxman et al., 2005). These problems occur naturally in many modern applications, such as online advertising and recommender systems (Li et al., 2010). For instance, in a recommender system, the partial feedback represents whether the user likes the content recommended by the system, whereas the user's preference for the other contents that have not been displayed remain unknown. In this paper, we consider one particular learning problem related to partial feedback: the online multiclass prediction problem in the bandit setting (Kakade et al., 2008). In the problem, the learner iteratively interacts with the environment. In each iteration, the learner observes an instance and is asked to predict its label. The main difference between the traditional full-information setting and the bandit setting is the feedback received after each prediction. In the full-information setting, the true label of the instance is revealed, whereas in the bandit setting, only whether the prediction is correct is known. That is, in the bandit setting, the true label remains unknown if the prediction is incorrect. Our goal is to make as few errors as possible in the harsh environment of the bandit setting.

With more restricted information available, it becomes harder to design good learning algorithms in the bandit setting, except for the case of online binary classification, in which the bandit setting and full-information setting coincide. Thus, it is desirable to find a systematic way to transform the existing online binary classification algorithms, or combine many of them, to get an algorithm that effectively deals with the bandit setting. The motivation calls for *boosting* (Schapire, 1990), which is one of the most popular and well-developed ensemble methods implemented in the traditional batch supervised classification framework. While most studies on boosting focus on the batch setting (Freund & Schapire, 1997; Schapire & Singer, 1999), some works have extended the success of boosting to the online setting (Oza & Russell, 2001; Chen et al., 2012). However, to the best of our knowledge, there is no boosting algorithm yet for the problem of online multiclass prediction in the bandit setting. In this paper, we study the possibility of

Proceedings of the 31^{st} International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

extending the promising theoretical and empirical results of boosting to the bandit setting.

As in the design and analysis of boosting algorithms in other settings, we need an appropriate assumption on weak learners in order for boosting to work. A stronger assumption makes the design of a boosting algorithm easier, but at the expense of more restricted applicability. To weaken the assumption, we consider binary full-information weak learners, instead of multiclass bandit ones, with the given binary examples constructed through the one-versus-rest decomposition from the multiclass examples (Chen et al., 2009). Following (Chen et al., 2012), we propose a similar assumption which requires such binary weak learners to perform better than random guessing with respect to "smooth" weight distributions over the binary examples. Then we prove that boosting is possible under this assumption by designing a strong bandit algorithm using such binary weak learners. Our bandit algorithm is extended from the full-information one of (Chen et al., 2012), which provides a method to generate such smooth example weights for updating weak learners, as well as some appropriate voting weights for combining the predictions of weak learners.

Nevertheless, our extension in this paper is non-trivial. To compute these weights exactly in (Chen et al., 2012), one needs the full-information feedback, which is not available in our bandit setting. With the limited information of bandit feedback, we show how to find good estimators for the example weights as well as for the voting weights, and we prove that they can in fact be used to replace the true weights to make boosting work in the bandit setting.

Our proposed bandit boosting algorithm enjoys nice theoretical properties similar to those of its batch counterpart. In particular, the proposed algorithm can achieve a small error rate if the performance of each weak learner is better than that of random guessing with respect to the carefully generated weight distributions. In addition, the algorithm reaches promising empirical performance on real-world data sets, even when using very simple full-information weak learners.

Finally, let us stress the difference between our work and existing ones on the bandit problem. Unlike existing works, our goal is not to construct one specific bandit algorithm and analyze its regret. Instead, our goal is to study the possibility of a general paradigm for designing bandit algorithms in a systematic way. Note that there are currently only a very small number of bandit algorithms for the multiclass prediction problem, and most seem to be based on linear models (Kakade et al., 2008; Hazan & Kale, 2011). With the limited power of such linear models, a high error rate is unavoidable in general, so the focus of these works was to reduce the regret, regardless of whether the actual error rate is high. Our result, on the other hand, works for a broader class of classifiers beyond linear ones. We show how to construct a strong bandit algorithm with an error rate close to zero, when we have weak learners which can perform slightly better than random guessing. Here we allow any weak learners, not just linear ones, that only need to work in the simpler full-information setting rather than in the more challenging bandit setting. Constructing such weak learners may look much less daunting, but we show that they in fact suffice for constructing strong bandit algorithms. We hope that this could open more possibilities for designing better bandit algorithms in the future.

2. Boosting in different settings

Before formally describing our boosting framework in the online bandit setting, let us first review the traditional batch setting as well as the online full-information setting.

In the batch setting, the boosting algorithm has the whole training set $S = \{(x_1, y_1), \ldots, (x_T, y_T)\}$ available at the beginning, where each x_t is the feature vector from some space $\mathcal{X} \subseteq \mathbb{R}^d$ and y_t is its label from some space $\mathcal{Y}$. For the case of binary classification, we assume $\mathcal{Y} = \{-1, +1\}$, and the boosting algorithm repeatedly calls the batch weak learner for a number of rounds as follows. In round i, it feeds S as well as a probability distribution $p^{(i)}$ over S to the weak learner, which then returns a weak hypothesis $h^{(i)}$ after seeing the whole S and $p^{(i)}$. It stops at some round N when the strong hypothesis $H(x) = \operatorname{sign}(\sum_{i=1}^{N} \alpha^{(i)} h^{(i)}(x))$, with $\alpha^{(i)} \in \mathbb{R}$ being the voting weight of $h^{(i)}$, achieves a small error rate over S, defined as $|\{t : H(x_t) \neq y_t\}|/T$.

For the case of multiclass classification, we assume $\mathcal{Y} = \{1, \ldots, K\}$, and for simplicity we adopt the one-versus-rest approach to reduce the multiclass problem to a binary one. More precisely, each multiclass example (x_t, y_t) is decomposed into K binary examples $((x_t, k), y_{tk})$, $k = 1, \ldots, K$, where y_{tk} is 1 if $y_t = k$ and -1 otherwise. One can then apply the boosting algorithm to such binary examples and use $H(x) = \arg\max_k \sum_i \alpha_k^{(i)} h^{(i)}(x, k)$ as the strong hypothesis for the original multiclass problem.

In the online full-information setting, the examples of S are usually considered as chosen adversarially and they arrive one at a time. The online boosting algorithm must decide on some number N of online weak learners to start with. At step t, the boosting algorithm receives x_t and it predicts $H(x_t) = \arg\max_k \sum_i \alpha_{tk}^{(i)} h_t^{(i)}(x_t, k)$, where $h_t^{(i)}$ is the weak hypothesis provided by the i'th weak learner and $\alpha_{tk}^{(i)}$ is its voting weight. After the prediction, the true label y_t is revealed, and to update each weak learner, we would like to feed it with a probability measure on each binary example, as in the batch boosting. However, in the

online setting it is hard to determine a good measure of an example without seeing the remaining examples, so we instead only generate a weight $w_{tk}^{(i)}$ for $((x_t, k), y_{tk})$, which after normalization corresponds to the measure $p_{tk}^{(i)}$, for the i-th weak learner. The goal is again to achieve a small error rate over S, given that each weak learner has some positive advantage, defined as $\sum_{t,k} p_{tk}^{(i)} y_{tk} h_t^{(i)}(x_t, k)$. Chen et al. (2012) proposed an online boosting algorithm that achieves this goal in the binary case, which can be easily adapted to the multiclass case here.

In this paper, we consider the online multiclass prediction problem in the bandit setting. The setting is similar to the full-information one, except that at step t the boosting algorithm only receives the bandit information of whether its prediction is correct or not. The goal is essentially the same—to achieve a small error rate, given that each weak learner has some positive advantage.

Several issues arise in designing such a bandit boosting algorithm. The standard approach in designing a bandit algorithm is to use a full-information algorithm as a black box, with its needed information replaced by some estimated one. Usually, the only information needed by a full-information algorithm is the gradient of the loss function at each step, and this information is used only once, for updating its next strategy or action. As a result, the performance (regret) of such a bandit algorithm can be easily analyzed based on that of the full-information one, as it is usually expressed as a simple function of the gradients. For our boosting problem, we would also like to follow this approach, and the only available full-information boosting algorithm with theoretical guarantee is that of (Chen et al., 2012). However, it is not obvious what to estimate now since that algorithm involves three online processes which all need the information y_t, but for different purposes. First, the boosting algorithm needs y_t to compute the example weights $w_{tk}^{(i)}$'s. Second, the boosting algorithm needs y_t to compute the voting weights $\alpha_{tk}^{(i)}$'s. Third, the weak learners also need y_t, in addition to $w_{tk}^{(i)}$'s, to update its next hypothesis. Can one single bit of bandit information about y_t be used to get good estimators for all the three processes? Furthermore, as y_t is used in several places and in a more involved way, the bandit algorithm may not be able to use the full-information one as a simple black box, and its performance (error rate) may not be easily based on that of the full-information one. Finally, it is not clear what the appropriate assumption one should make on the weak learners in order for boosting to work in the bandit setting. In fact, it is not even clear what type of weak learners one should use. Perhaps the most natural choice is to use multiclass bandit algorithms. That is, starting from weak multiclass bandit algorithms, we "boost" them into strong multiclass bandit ones. Surprisingly, we will show that it suffices to use binary full-information algorithms with a positive advantage as weak learners. This not only gives us a stronger result in theory, as a weaker assumption on weak learners is needed, but also provides us more possibilities of designing weak learners (and thus strong bandit algorithms) in practice, as most existing multiclass bandit algorithms are linear ones.

We will use the following notation and convention. For a positive integer n, we let $[n]$ denote the set $\{1, \ldots, n\}$. For a condition π, we use the notation $\mathbf{1}[\pi]$ which gives the value 1 if π holds and 0 otherwise. For simplicity, we assume that each x_t has length $\|x_t\|_2 \leq 1$ and each hypothesis h_t comes from some family $\mathcal{H}$ with $h_t(x_t, k) \in [-1, 1]$.

3. Online weak learners

In this section, we study reasonable assumptions on weak learners for allowing boosting to work in the bandit setting. As mentioned in the previous section, instead of using multiclass bandit algorithms as weak learners, we will use binary full-information ones. A natural assumption to make is for such a binary full-information algorithm to achieve a positive advantage with respect to any example weights. However, as noted in (Chen et al., 2012), this assumption is too strong to achieve, as one cannot expect an online algorithm to achieve a positive advantage in extreme cases, such as when only the first example has a nonzero weight. Thus, some constraints must be put on the example weights.

To identify an appropriate constraint, let us follow (Chen et al., 2012) and consider the case that each hypothesis h_t consists of K linear functions with $h_t(x, k) = \langle h_{tk}, x \rangle$, the inner product of two vectors h_{tk} and x, with $\|h_{tk}\|_2 \leq 1$. When given an example (x_t, k), the weak learner uses h_{tk} to predict the binary label y_{tk}. After that, it receives y_{tk} as well as the example weight w_{tk}, and uses them to update h_{tk} into a new $h_{(t+1)k}$. We can reduce the task of such a weak learner to the well-known online linear optimization problem, by using the reward function $r_{tk}(h_{tk}) = w_{tk} y_{tk} \langle h_{tk}, x_t \rangle$, which is linear in h_{tk}. Then we can apply the online gradient descent algorithm of (Zinkevich, 2003) to generate h_{tk} at step t, and a standard regret analysis shows that for some constant $c > 0$,

$$\sum_t w_{tk} y_{tk} \langle h_{tk}, x_t \rangle \geq \sum_t w_{tk} y_{tk} \langle h_k^*, x_t \rangle - \sqrt{c \sum_t w_{tk}^2}$$

for any h_k^* with $\|h_k^*\|_2 \leq 1$. Summing over $k \in [K]$ and using Cauchy-Schwarz inequality, we get

$$\sum_{t,k} w_{tk} y_{tk} \langle h_{tk}, x_t \rangle \geq \sum_{t,k} w_{tk} y_{tk} \langle h_k^*, x_t \rangle - \sqrt{cK \sum_{t,k} w_{tk}^2}.$$

Let $|w|$ denote the total weight $\sum_{t,k} w_{tk}$, so that $p_{tk} = \frac{w_{tk}}{|w|}$ is the measure of example (x_t, k). Then by dividing both

sides of the inequality above by $|w|$, we obtain

$$\sum_{t,k} p_{tk} y_{tk} \langle h_{tk}, x_t \rangle \geq \sum_{t,k} p_{tk} y_{tk} \langle h^*_k, x_t \rangle - \sqrt{cK \sum_{t,k} \frac{w_{tk}^2}{|w|^2}}.$$

Note that $\sum_{t,k} p_{tk} y_{tk} \langle h^*_k, x_t \rangle$ is the advantage of the offline learner, and suppose that it is at least $3\gamma > 0$. Moreover, suppose the example weights are large, in the sense that they satisfy the following condition:

$$|w| \geq cKB/\gamma^2, \tag{1}$$

where $B \geq \max_{t,k} w_{tk}$ is a constant that will be fixed later. Then the advantage of the online weak learner becomes

$$\sum_{t,k} p_{tk} y_{tk} \langle h_{tk}, x_t \rangle \geq 3\gamma - \sqrt{cK \frac{B|w|}{|w|^2}} \geq 3\gamma - \gamma = 2\gamma.$$

This motivates us to propose the following assumption on weak learners, which need not be linear ones.

Assumption 1. *There is an online full-information weak learner which can achieve an advantage $2\gamma > 0$ for any sequence of examples and weights satisfying condition (1).*

From the discussion above, we have the following.

Lemma 1. *Suppose for any sequence of examples and weights satisfying condition (1), there exists an offline linear hypothesis with an advantage $3\gamma > 0$. Then Assumption 1 holds.*

Let us make two remarks on Assumption 1. First, the assumption that a weak learner has a positive advantage is just the assumption that it predicts better than random guessing, which is the standard assumption used by (almost) all previous batch boosting algorithms. Second, the condition (1) on example weights actually makes our assumption weaker, which in turn makes the boosting task harder and our boosting result in the next section stronger. More precisely, we only require the weak learner to perform well (having a positive advantage) when the weights are large, and we do not care how bad it may perform with small weights. In fact, we will make our boosting algorithm call the weak learner with large weights.

4. Our bandit boosting algorithm

In this section we show how to design a bandit boosting algorithm under Assumption 1. Let WL be such an online full-information weak learner and we will run N copies of WL, for some N to be determined later. We follow the approach of reducing the multiclass problem to the binary one as described in Section 2, and we base our bandit boosting algorithm on the full-information one of (Chen et al., 2012) that works for binary classification in the full-information setting. More precisely, at step t we do the following after receiving the feature vector x_t. For each class $k \in [K]$, a new feature vector (x_t, k) is created, we obtain a binary weak hypothesis $h^{(i)}_{tk}(x) = h^{(i)}_t(x, k)$ from the i'th weak learner, for $i \in [N]$, and we form the strong hypothesis

$$H_t(x) = \arg\max_{k \in [K]} f_{tk}(x), \text{ with } f_{tk}(x) = \sum_{i=1}^{N} \alpha^{(i)}_{tk} h^{(i)}_{tk}(x),$$

where $\alpha^{(i)}_{tk}$ is some voting weight for the i'th weak learner. Then we make our prediction $\hat{y}_t$ based on $H_t(x_t)$ in some way and receive the feedback $\mathbf{1}[\hat{y}_t = y_t]$. Using the feedback, we prepare some example weight $w^{(i)}_{tk}$ to update the i'th weak learner, as well as to compute the next voting weight $\alpha^{(i)}_{(t+1)k}$, for $i \in [N]$. It remains to show how to set the example weights and the voting weights, as well as how to choose $\hat{y}_t$, which we describe and analyze in detail next. The complete algorithm is given in Algorithm 1.

Algorithm 1 Bandit boosting algorithm with online weak learner WL

Input: Streaming examples $(x_1, y_1), \ldots, (x_T, y_T)$.
Parameters: $0 < \delta < 1$, $0 \leq \theta < \gamma < \frac{1}{2}$.
Choose $\alpha^{(i)}_{1k} = \frac{1}{N}$ and random $h^{(i)}_{1k}$ for $k \in [K], i \in [N]$.
for $t = 1$ **to** T **do**
  Let $H_t(x) = \arg\max_{k \in [K]} \sum_{i \in [N]} \alpha^{(i)}_{tk} h^{(i)}_{tk}(x)$.
  Let $p_t(k) = (1 - \delta)\mathbf{1}[k = H_t(x_t)] + \frac{\delta}{K}$ for $k \in [K]$.
  Predict $\hat{y}_t$ according to the distribution p_t.
  Receive the information $\mathbf{1}[\hat{y}_t = y_t]$.
  for $k = 1$ **to** K **and** $i = 1$ **to** N **do**
    Update $w^{(i)}_{tk}$ according to (4).
    If $\hat{y}_t = k$, call $\mathsf{WL}(h^{(i)}_{tk}, (x_t, k), y_{tk}, w^{(i)}_{tk})$ to obtain $h^{(i)}_{(t+1)k}$; otherwise, let $h^{(i)}_{(t+1)k} = h^{(i)}_{tk}$.
    Update $\alpha^{(i)}_{(t+1)k}$ according to (6).
  end for
end for

The example weight of (x_t, k) for the i'th weak learner used by the full-information algorithm of (Chen et al., 2012) is

$$\bar{w}^{(i)}_{tk} = \min\left\{(1 - \gamma)^{z^{(i-1)}_{tk}/2}, 1\right\}, \tag{2}$$

where $z^{(0)}_{tk} = 0$ and $z^{(i-1)}_{tk}$, for $i - 1 \geq 1$, is defined as

$$z^{(i-1)}_{tk} = \sum_{j=1}^{i-1} (y_{tk} h^{(j)}_{tk}(x_t) - \theta), \text{ with } \theta = \gamma/(2 + \gamma), \tag{3}$$

which depends on the information y_{tk}. As we are in the bandit setting, we do not have y_{tk} to compute such weights in general. Thus, we balance exploitation with exploration

by independently predicting $H_t(x_t)$ with probability $1-\delta$ and a random label with probability δ, with $\delta \le \gamma$; let $\hat{y}_t$ denote our prediction. For $k \in [K]$, let $p_t(k)$ denote the probability that $\hat{y}_t = k$. Then we replace the example weight $\bar{w}_{tk}^{(i)}$ by the estimator

$$w_{tk}^{(i)} = \begin{cases} \bar{w}_{tk}^{(i)}/p_t(k) & \text{if } \hat{y}_t = k, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

which we can compute, because when $\hat{y}_t = k$, we do have y_{tk} to compute $\bar{w}_{tk}^{(i)}$. As $p_t(k) \ge \delta/K$, we can choose $B = K/\delta$ and have $w_{tk}^{(i)} \le B$ for any t, k, i. Note that $w_{tk}^{(i)}$ and $\bar{w}_{tk}^{(i)}$ are random variables, and the following shows that $w_{tk}^{(i)}$'s are in fact good estimators for $\bar{w}_{tk}^{(i)}$'s.

Claim 1. *For any t, k, i, $\mathbb{E}[\bar{w}_{tk}^{(i)}] = \mathbb{E}[w_{tk}^{(i)}]$. For any k, i, λ,*

$$\Pr\left[\sum_t \left(\bar{w}_{tk}^{(i)} - w_{tk}^{(i)}\right) > \lambda T\right] \le 2^{-\Omega(\lambda^2 T/B^2)}.$$

Proof. Observe that any fixing of the randomness up to step $t-1$ leaves $\bar{w}_{tk}^{(i)}$ and $w_{tk}^{(i)}$ with the same conditional expectation. Thus, $\mathbb{E}[\bar{w}_{tk}^{(i)}] = \mathbb{E}[w_{tk}^{(i)}]$. Moreover, as the random variables $M_t = \bar{w}_{tk}^{(i)} - w_{tk}^{(i)}$, for $t \in [T]$, form a martingale difference sequence, with $|M_t| \le B$, the probability bound follows from Azuma's inequality. □

This claim allows us to use $w_{tk}^{(i)}$ as the example weight of (x_t, k) to update the i'th weak learner. However, as each weak learner is assumed to be a full-information one, it also needs the label y_{tk} to update which we may not know. One may try to take a similar approach as before to feed the weak learner with an estimator which is $y_{tk}/p_t(k)$ when $\hat{y}_t = k$ and 0 otherwise, but this does not work as it does not take a value in $\{-1, 1\}$ needed by the binary weak learner. Instead, we take a different approach: we only call the weak learner to update when $\hat{y}_t = k$ so that we know y_{tk}. That is, when $\hat{y}_t = k$, we call the i'th weak learner with $w_{tk}^{(i)}$ and y_{tk}, which can then update and return the next weak hypothesis $h_{(t+1)k}^{(i)}$; otherwise, we do not call the i'th weak learner to update and we simply let the next hypothesis $h_{(t+1)k}^{(i)}$ be the current $h_{tk}^{(i)}$. Another issue is that a weak learner is only assumed to work well when given large example weights satisfying condition (1), and even then, it only works well on those examples which are given to it to update. This is dealt by the following.

Lemma 2. *Let $\delta \in [0, 1]$, let m be the largest number such that $\sum_{t,k} w_{tk}^{(i)} \ge \delta KT$ for every $i \le m$, and let*

$$\bar{f}_{tk}(x) = \sum_{i=1}^{m} \frac{1}{m} h_{tk}^{(i)}(x).$$

Then when $T \ge c_0(K^2/\delta^4)\log(K/\delta)$ for a large enough constant c_0,

$$\Pr\left[\left|\{(t,k) : y_{tk}\bar{f}_{tk}(x_t) < \theta\}\right| > 2\delta KT\right] \le \delta,$$

for the parameter $\theta = \gamma/(2+\gamma)$ introduced in (3)

Proof. Note that according to the definition, for any t and k, $\bar{w}_{tk}^{(m+1)} \ge 0$, and $\bar{w}_{tk}^{(m+1)} = 1$ if $y_{tk}\bar{f}_{tk}(x_t) < \theta$, as $z_{tk}^{(m)} = m(y_{tk}\bar{f}_{tk}(x_t) - \theta)$. This implies that

$$\left|\{(t,k) : y_{tk}\bar{f}_{tk}(x_t) < \theta\}\right| \le \sum_{t,k} \bar{w}_{tk}^{(m+1)}.$$

As $\sum_{t,k} w_{tk}^{(m+1)} < \delta KT$ by the definition of m, we have

$$\begin{aligned} &\Pr\left[\left|\{(t,k) : y_{tk}\bar{f}_{tk}(x_t) < \theta\}\right| > 2\delta KT\right] \\ \le\ &\Pr\left[\sum_{t,k} \bar{w}_{tk}^{(m+1)} - \sum_{t,k} w_{tk}^{(m+1)} > \delta KT\right], \end{aligned}$$

which by a union bound and Claim 1 is at most $K2^{-\Omega(\delta^2 T/B^2)} \le \delta$. □

The following lemma gives an upper bound on the parameter m defined in Lemma 2.

Lemma 3. *Suppose Assumption 1 holds and $T \ge cK/(\delta^2\gamma^2)$ for the constant c in the condition (1). Then the parameter m in Lemma 2 is at most $\mathcal{O}(K/(\delta^2\gamma^2))$.*

Proof. Note that for any $i \in [m]$, $\sum_{t,k} w_{tk}^{(i)} \ge \delta KT \ge cKB/\gamma^2$, with $B = K/\delta$, and the condition (1) is satisfied. Thus from Assumption 1, we have

$$\sum_{i,t,k} w_{tk}^{(i)} y_{tk} h_{tk}^{(i)}(x_t) \ge 2\gamma \sum_{i,t,k} w_{tk}^{(i)}, \tag{5}$$

with the sums over i above, as well as in the rest of the proof, being taken over $i \in [m]$. On the other hand, we have the following claim.

Claim 2. $\sum_{i,t,k} w_{tk}^{(i)} y_{tk} h_{tk}^{(i)}(x_t) \le \mathcal{O}(BKT/\gamma) + \gamma \sum_{i,t,k} w_{tk}^{(i)}$.

We omit its proof here as it is very similar to that for Lemma 5 in (Servedio, 2003).[1] Combining the bound in Claim 2 with the inequality (5), we have

$$\gamma \sum_{i,t,k} w_{tk}^{(i)} \le \mathcal{O}(BKT/\gamma).$$

[1] Although that lemma is for $\bar{w}_{tk}^{(i)}$'s, its proof can be easily modified to work for $w_{tk}^{(i)}$'s, but with an additional factor of B appearing in the term $\mathcal{O}(BKT/\gamma)$ here.

Since $\sum_{t,k} w_{tk}^{(i)} \geq \delta KT$ for $i \in [m]$ and $B = K/\delta$, we get $\gamma m \delta KT \leq \mathcal{O}(K^2 T/(\delta\gamma))$. From this, the required bound on m follows, and we have the lemma. □

Let us suppose that $T \geq c_0(K^2/\delta^4)\log(K/\delta)$ for a large enough constant c_0 so that both lemmas apply. Then Lemma 2 shows that one can obtain a strong learner by combining the first m weak learners. However, one cannot determine the number m before seeing all the examples, and in fact in our online setting, we need to decide the number N of weak learners even before seeing the first example. Following (Chen et al., 2012), we set N to be the upper bound given by Lemma 3. Then at step t, for each $k \in [K]$, we consider the function

$$f_{tk}(x) = \sum_{i=1}^{N} \alpha_{tk}^{(i)} h_{tk}^{(i)}(x)$$

and reduce the task of finding such $\alpha_{tk} = (\alpha_{tk}^{(1)}, \ldots, \alpha_{tk}^{(N)})$ to the *Online Convex Programming* problem. More precisely, we use the N-dimensional probability simplex, denoted by $\mathcal{P}_N$, as the feasible set and define the loss function as

$$L_{tk}(\alpha) = \max\left\{0, \theta - y_{tk} \sum_{i=1}^{N} \alpha^{(i)} h_{tk}^{(i)}(x_t)\right\},$$

which is a convex function of α. However, unlike in (Chen et al., 2012), we are in the bandit setting and thus may not know y_{tk}. To overcome this, we use a similar idea as before to estimate a subgradient $\nabla L_{tk}(\alpha_{tk})$ by

$$\ell_{tk} = \begin{cases} \nabla L_{tk}(\alpha_{tk})/p_t(k) & \text{if } \hat{y}_t = k, \\ 0 & \text{otherwise.} \end{cases}$$

One can then use $\ell_{tk} = (\ell_{tk}^{(1)}, \ldots, \ell_{tk}^{(N)})$ to perform gradient descent to update α_{tk} as in (Chen et al., 2012). However, to get a better theoretical bound, here we choose to perform a multiplicative update on α_{tk} to get $\alpha_{(t+1)k} = (\alpha_{(t+1)k}^{(1)}, \ldots, \alpha_{(t+1)k}^{(N)})$ for step $t+1$, with

$$\alpha_{(t+1)k}^{(i)} = \alpha_{tk}^{(i)} \cdot e^{-\eta \ell_{tk}^{(i)}} / Z_{(t+1)k}, \tag{6}$$

where $Z_{(t+1)k}$ is the normalization factor and η is the learning rate which we set to δ^3/K. Then we have the following.

Lemma 4. $\Pr\left[\sum_{t,k} L_{tk}(\alpha_{tk}) \leq \mathcal{O}(\delta KT)\right] \geq 1 - 2\delta.$

Proof. Following the standard analysis, one can show that for any $k \in [K]$ and any $\bar{\alpha}_k \in \mathcal{P}_N$,

$$\begin{aligned} \sum_t \langle \ell_{tk}, \alpha_{tk} - \bar{\alpha}_k \rangle &\leq \mathcal{O}((\log N)/\eta) + \eta \sum_t \|\ell_{tk}\|_\infty^2 \\ &\leq \mathcal{O}(\delta KT), \end{aligned} \tag{7}$$

since $\|\ell_{tk}\|_\infty^2 \leq B^2 = K^2/\delta^2$, $\eta = \delta^3/K$ and $N \leq \mathcal{O}(K/\delta^4)$. Now note that for any t and k, $\mathbb{E}[\langle \ell_{tk}, \alpha_{tk} - \bar{\alpha}_k \rangle] = \mathbb{E}[\langle \nabla L_{tk}(\alpha_{tk}), \alpha_{tk} - \bar{\alpha}_k \rangle]$ because given the randomness up to step $t-1$, α_{tk} is fixed and the conditional expectation of ℓ_{tk} equals $\nabla L_{tk}(\alpha_{tk})$. Moreover, as $L_{tk}(\alpha_{tk}) - L_{tk}(\bar{\alpha}_k) \leq \langle \nabla L_{tk}(\alpha_{tk}), \alpha_{tk} - \bar{\alpha}_k \rangle$ for convex L_{tk}, we have

$$\mathbb{E}\left[L_{tk}(\alpha_{tk}) - L_{tk}(\bar{\alpha}_k)\right] \leq \mathbb{E}[\langle \ell_{tk}, \alpha_{tk} - \bar{\alpha}_k \rangle].$$

Then using the bound in (7) and applying a similar martingale analysis as before, one can show that for any $\bar{\alpha}_k \in \mathcal{P}_N$,

$$\Pr\left[\sum_{t,k} \left(L_{tk}(\alpha_{tk}) - L_{tk}(\bar{\alpha}_k)\right) \leq \mathcal{O}(\delta KT)\right] \geq 1 - \delta.$$

Let $\bar{\alpha}_k = (\bar{\alpha}_k^{(1)}, \ldots, \bar{\alpha}_k^{(N)})$, with $\bar{\alpha}_k^{(i)} = \frac{1}{m}$ for $i \leq m$ and $\bar{\alpha}_k^{(i)} = 0$ for $i > m$, so that

$$\begin{aligned} L_{tk}(\bar{\alpha}_k) &= \max\{0, \theta - y_{tk} \bar{f}_{tk}(x_t)\} \\ &\leq (1+\theta)\mathbf{1}[y_{tk} \bar{f}_{tk}(x_t) < \theta]. \end{aligned}$$

Then we know from Lemma 2 that

$$\Pr\left[\sum_{t,k} L_{tk}(\bar{\alpha}_k) \leq (1+\theta) 2\delta KT\right] \geq 1 - \delta.$$

Combining the two probability bounds together, we have the lemma. □

Finally, recall that to predict each y_t, we independently output $H_t(x_t) = \arg\max_{k \in [K]} f_{tk}(x_t)$ with probability $1-\delta$ and a random label with probability δ. Thus, by a Chernoff bound, our algorithm makes at most $|\{t : H_t(x_t) \neq y_t\}| + 2\delta T$ errors with probability $1 - 2^{-\Omega(\delta^2 T)} \geq 1 - \delta$. On the other hand, as

$$\begin{aligned} \mathbf{1}[H_t(x_t) \neq y_t] &\leq \sum_k \mathbf{1}[y_{tk} f_{tk}(x_t) < 0] \\ &\leq \sum_k L_{tk}(\alpha_{tk})/\theta, \end{aligned}$$

Lemma 4 implies that

$$\Pr\left[|\{t : H_t(x_t) \neq y_t\}| \leq \mathcal{O}(\delta KT/\theta)\right] \geq 1 - 2\delta.$$

Consequently, for $\theta = \gamma/(2+\gamma)$, we can conclude that our algorithm makes at most

$$\mathcal{O}(K\delta T/\theta) + 2\delta T \leq \mathcal{O}(K\delta T/\gamma)$$

errors with probability at least $1 - 3\delta$. Therefore, we have the following, which is the main result of our paper.

Theorem 1. *Suppose Assumption 1 holds and* $T \geq c_0(K^2/\delta^4)\log(K/\delta)$ *for a large enough constant* c_0*. Then our bandit algorithm uses* $\mathcal{O}(K/(\delta^2\gamma^2))$ *weak learners and makes* $\mathcal{O}(K\delta T/\gamma)$ *errors with probability* $1-3\delta$*.*

Note that the error rate of our algorithm is $\mathcal{O}(K\delta/\gamma)$, which can be made to any ε by setting $\delta = \mathcal{O}(\varepsilon\gamma/K)$, with the requirement on T and the number of weak learners adjusted accordingly. We remark that we did not attempt to optimize our bounds (which we believe can be improved) as our focus was on establishing the possibility of boosting in the bandit setting. Moreover, it does not seem appropriate to compare our error bound with the regret bounds of existing bandit algorithms. This is because existing algorithms are usually based on linear classifiers, which may have large error rates even though their regrets are small. On the other hand, our boosting algorithm works for any type of classifiers and achieves a small error rate as long as we have weak learners which satisfy Assumption 1.

5. Experiments

In this section, we validate the empirical performance of the proposed algorithm on several real-world data sets. We compare with two representative algorithms. The first one is **Banditron** (Kakade et al., 2008), which is one of the first proposed algorithms for the bandit setting. It is modified from a multiclass variant of the well-known Perceptron algorithm (Rosenblatt, 1962) using the so-called Kesler's construction (Duda & Hart, 1973). By doing some random exploration, it can accurately construct the estimation of the update step for the full-information multiclass Perceptron. The algorithm has good theoretical guarantee, especially when the data is linearly separable. The algorithm can be viewed as a direct modification of a full-information learner (Perceptron) for the bandit setting, without combining the learners for boosting.

The second one is **Conservative OVA** (C-OVA) (Chen et al., 2009), which uses the one-versus-all multiclass to binary decomposition similar to our algorithm. But unlike most of the bandit algorithms, it does not do random exploration at all. Instead, it conservatively updates using whatever it gets from the partial feedback, and hence the name. Note that although it embeds an online binary learning algorithm as its "base learner", it does not perform boosting by combining several base learners like our algorithm does. Also, C-OVA performs a margin-based decoding of the binary classification results, and hence may not work well with non-margin-based base learners.

To demonstrate the boosting ability of our proposed algorithm, we choose two completely different types of online binary classifiers as our weak learners. The first one is Perceptron, a standard margin-based linear classifier. Note that in (Chen et al., 2009) they use a similar but more complex *Online Passive-Aggressive Algorithm* (PA) (Crammer et al., 2006) as its internal learner. Since we found little difference in performance on the data sets we tested between the PA algorithm and the Perceptron algorithm, we only report the results using the simpler and more famous Perceptron algorithm to compare fairly with Banditron. The second weak learner we use is *Naive Bayes*, a simple statistical classifier that estimates the posterior probability for each class using Bayes theorem and the assumption of conditional independence between features.

Table 1. The data sets used in our experiments.

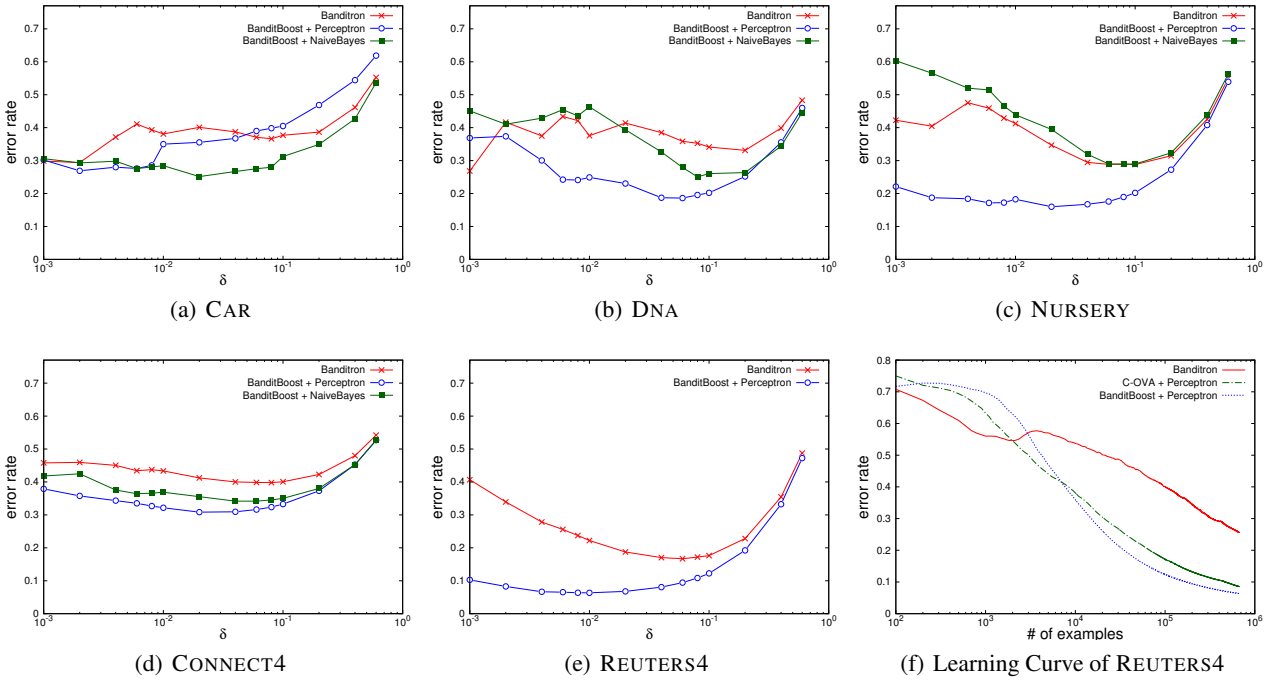
Data set	Car	DNA	Nursery	Connect4	Reuters4
#classes	4	3	5	3	4
#features	6	180	8	126	346,810
#examples	1,728	3,186	12,960	67,557	673,768

5.1. Results

We test our algorithm on 5 public real-world data sets from various domains with different sizes: CAR, NURSERY, and CONNECT4 from the UCI machine learning repository (Frank & Asuncion, 2010); DNA from the Statlog project (Michie et al., 1994); REUTERS4 from the paper of Banditron (Kakade et al., 2008). Basic information of these data sets are summarized in Table 1. As described previously, each example is first used for prediction before the disclosure of its label, and the error rate is the number of prediction errors divided by the total number of examples. All the experiments are repeated 10 times with different random orderings of the examples.

For fairness of comparison to Banditron, we do not tune the parameters other than the exploration rate δ. We fix the number of weak learners to be 100 and the assumed weak learner advantage γ to be 0.1 as in the full-information online boosting algorithm (Chen et al., 2012). For the exploration rate δ, we test a wide range of values to see the effect of random exploration. The results are shown in Figure 1. Note that C-OVA is not included in this figure since C-OVA does not perform random exploration at all and is parameter-free. One can see that for reasonable range of values of δ (around 0.01 to 0.1), the performance of our algorithm is quite strong and relatively stable, while setting it too high or too low results in worse performance as expected. Table 2 summarizes the average error rate and the standard deviation when the best choices of δ are used in Banditron and in our algorithm.

Let us first focus on the case when Perceptron is used as the weak learner. Here, the categorical features are transformed into numerical ones by decomposition into binary vectors. We can see that the proposed bandit boosting algorithm consistently outperforms Banditron on all the data sets, and is also comparable to C-OVA, especially on larger



(a) CAR (b) DNA (c) NURSERY

(d) CONNECT4 (e) REUTERS4 (f) Learning Curve of REUTERS4

Figure 1. (a)-(d): Error rate using different values of exploration rate δ. (f): learning curve of REUTERS4 using the best δ

Table 2. Average (over 10 trials) error rate (%) and standard deviation comparison

Data set	Banditron	C-OVA (Perceptron)	BanditBoost (Perceptron)	C-OVA (Naive Bayes)	BanditBoost (Naive Bayes)
CAR	29.4 ± 0.9	22.8 ± 1.1	26.9 ± 2.4	30.0 ± 0.1	25.1 ± 1.8
DNA	26.8 ± 9.0	13.5 ± 0.5	18.6 ± 0.6	42.9 ± 8.3	25.1 ± 2.6
NURSERY	28.8 ± 1.4	17.9 ± 3.0	16.0 ± 1.1	59.3 ± 8.2	28.9 ± 2.1
CONNECT4	39.8 ± 0.4	28.1 ± 0.2	30.8 ± 0.2	34.9 ± 2.4	34.2 ± 0.4
REUTERS4	16.7 ± 0.5	8.5 ± 4.2	6.3 ± 0.1	N/A	N/A

data sets. To take a closer look at the performance of these algorithms, we plot the learning curve for the largest data set (REUTERS4) in Figure 1 (f). One can see that our algorithm begins to outperform the other algorithms when the number of examples is sufficiently large. This is due to the more complex model we use and the need for random exploration as opposed to the deterministic C-OVA algorithm. Note that it is in accordance to our analysis in Theorem 1, as the error bound only holds when the number of rounds T is large.

Next, let us see the situation when the weak learner is switched to Naive Bayes. Note that here we did not test on the REUTERS4 data set due to the slow inference of Naive Bayes for high dimensional data. It can be seen that our algorithm consistently reaches the best on all the data sets. Moreover, we see a large difference between C-OVA and our algorithm, especially in DNA and NURSERY data sets. The superiority echoes the earlier conjecture that C-OVA may not work well with non-margin-based base learners. On the other hand, the proposed bandit boosting algorithm enjoys a stronger theoretical guarantee and works well with various types of weak learners.

6. Conclusion

We propose a boosting algorithm to efficiently generate strong multiclass bandit learners by exploiting the abundance of existing online binary learners. The proposed algorithm can be viewed as a careful combination of the online boosting algorithm for binary classification (Chen et al., 2012) and some key estimation techniques in the bandit algorithms. While the proposed algorithm is simple, we show some non-trivial theoretical analysis that leads to sound theoretical guarantee. To the best of our knowledge, our proposed boosting algorithm is the first one that comes with such theoretical guarantee. In addition, experimental results on real-world data sets show that the proposed bandit boosting algorithm can be easily coupled with different weak binary learners to reach promising performance.

References

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. The non-stochastic multi-armed bandit problem. *SIAM Journal of Computing*, 32:48–77, 2002.

Chen, G., Chen, G., Zhang, J., Chen, S., and Zhang, C. Beyond banditron: A conservative and efficient reduction for online multiclass prediction with bandit setting model. In *Proceedings of ICDM*, pp. 71–80, 2009.

Chen, S.-T., Lin, H.-T., and Lu, C.-J. An online boosting algorithm with theoretical justifications. In *Proceedings of ICML*, pp. 1007–1014, July 2012.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585, December 2006.

Duda, R. O. and Hart, P. E. *Pattern Classification and Scene Analysis*. Wiley, 1973.

Flaxman, A. D., Kalai, A. T., and McMahan, H. B. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of SODA*, pp. 385–394, Philadelphia, PA, USA, 2005.

Frank, A. and Asuncion, A. UCI machine learning repository, 2010. URL http://archive.ics.uci.edu/ml.

Freund, Y. and Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1): 119–139, 1997.

Hazan, E. and Kale, S. Newtron: an efficient bandit algorithm for online multiclass prediction. In *Proceedings of NIPS*, pp. 891–899, 2011.

Kakade, S. M., Shalev-Shwartz, S., and Tewari, A. Efficient bandit algorithms for online multiclass prediction. In *Proceedings of ICML*, pp. 440–447, New York, NY, USA, 2008.

Li, L., Chu, W., Langford, J., and Schapire, R. E. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of WWW*, pp. 661–670, New York, NY, USA, 2010. ACM.

Michie, D., Spiegelhalter, D. J., and Taylor, C. C. Machine learning, neural and statistical classification, 1994.

Oza, N. C. and Russell, S. Online bagging and boosting. In *Proceedings of AISTATS*, pp. 105–112, 2001.

Rosenblatt, F. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan, 1962.

Schapire, R. E. The strength of weak learnability. *Mach. Learn.*, 5(2):197–227, July 1990.

Schapire, R. E. and Singer, Y. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, December 1999.

Servedio, R. A. Smooth boosting and learning with malicious noise. *JMLR*, 4:473–489, 2003.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of ICML*, pp. 928–936, 2003.