Optimal and Adaptive Algorithms for Online Boosting

Alina Beygelzimer
beygel@yahoo-inc.com
Yahoo Research
New York, NY 10036

Satyen Kale
satyen@yahoo-inc.com
Yahoo Research
New York, NY 10036

Haipeng Luo
haipengl@cs.princeton.edu
Princeton University
Princeton, NJ 08540

Abstract

We study online boosting, the task of converting any weak online learner into a strong online learner. Based on a novel and natural definition of weak online learnability, we develop two online boosting algorithms. The first algorithm is an online version of boost-by-majority. By proving a matching lower bound, we show that this algorithm is essentially optimal in terms of the number of weak learners and the sample complexity needed to achieve a specified accuracy. The second algorithm is adaptive and parameter-free, albeit not optimal.

1 Introduction

We study online boosting, the task of boosting the accuracy of any weak online learning algorithm. The theory of boosting in the batch setting has been studied extensively, leading to a huge practical success. See the book by [Schapire and Freund, 2012] for a thorough discussion.

In the batch setting, we typically have a fixed training set S drawn from the underlying distribution. A boosting algorithm assumes access to a weak learner that achieves error at most $\frac{1}{2} - \gamma$ on any distribution over S, for some fixed small "edge" $\gamma > 0$. Each round of boosting typically makes a pass over S, reweighting the examples and running a copy of the weak learner on this specially reweighted dataset.

On the other hand, online learning algorithms receive examples one by one, updating the predictor after seeing each new example. An online booster has to operate in the same fashion, reweighting and passing each example through all copies of the online weak learner before seeing any subsequent examples. In contrast to the batch setting, online learning algorithms typically do not make any stochastic assumptions about the data. They are often much faster, more memory-efficient, and can adapt to the best predictor changing over time.

The success of boosting in batch learning prompted an investigation of whether online learning algorithms can be boosted as well [Oza and Russell, 2001; Grabner and Bischof, 2006; Liu and Yu, 2007; Grabner *et al.*, 2008; Chen *et al.*, 2012; 2014]. From a theoretical viewpoint, the work by [Chen *et al.*, 2012] is perhaps most interesting. The authors proposed an online generalization of the batch weak learning assumption, and made a connection between online boosting and batch boosting that produces smooth distributions over the training examples. The resulting algorithm is guaranteed to achieve an arbitrarily small error rate as long as the number of weak learners and the number of examples are sufficiently large. No assumptions need to be made about how the data is generated, e.g., the data can be generated by an adversary.

We present a new online boosting algorithm, based on the boost-by-majority (BBM) algorithm of [Freund, 1995]. This algorithm, called Online BBM, improves upon the work of [Chen *et al.*, 2012] in several ways:

1. Our assumption on online weak learners is weaker and can be seen as a direct online analogue of the standard batch weak learning assumption.
2. Our algorithm does not require importance weighted online learning, instead using a sampling technique similar to the one used in boosting by filtering in the batch setting [Freund, 1992; Bradley and Schapire, 2008].
3. Our algorithm is optimal in the sense that no online boosting algorithm can achieve the same error rate with fewer weak learners or examples asymptotically.

The following table presents a comparison of the two papers for the setting where the weak learner is derived from an online learning algorithm with an $O(\sqrt{T})$ regret bound. Here N is the number of weak learners and T is the number of examples needed to achieve error rate ϵ, and γ is an online analog of the "edge" of the weak learning oracle.[1]

Algorithm	N	T
Online BBM (Section 3), optimal	$O(\frac{1}{\gamma^2}\ln\frac{1}{\epsilon})$	$\tilde{O}(\frac{1}{\epsilon\gamma^2})$
AdaBoost.OL (Section 4), adaptive	$O(\frac{1}{\epsilon\gamma^2})$	$\tilde{O}(\frac{1}{\epsilon^2\gamma^4})$
OSBoost [Chen *et al.*, 2012]	$O(\frac{1}{\epsilon\gamma^2})$	$\tilde{O}(\frac{1}{\epsilon\gamma^2})$

A clear drawback of both Online BBM and the algorithm in [Chen *et al.*, 2012] is their lack of adaptivity. These algorithms require knowledge of γ as a parameter. More importantly, this also means that the algorithm treats each weak learner equally and ignores the fact that some weak learners

[1] In this paper, we use the $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ notation to suppress dependence on polylogarithmic factors in the natural parameters.

are actually doing better than the others. The best example of adaptive boosting algorithm is the well-known parameter-free AdaBoost algorithm [Freund and Schapire, 1997], where each weak learner is naturally weighted by how accurate it is. In fact, adaptivity is known to be one of the key features that lead to the practical success of AdaBoost, and therefore should also be essential to the performance of online boosting algorithms. In Section 4, we propose AdaBoost.OL, an adaptive and parameter-free online boosting algorithm. As shown in the table, AdaBoost.OL is theoretically suboptimal in terms of N and T. However, empirically it generally outperforms OSBoost and sometimes even beats the optimal Online BBM (see Section 5).

Our techniques are also very different from those of [Chen *et al.*, 2012], which rely on the smooth boosting algorithm of [Servedio, 2003]. As far as we know, all other work on smooth boosting [Bshouty and Gavinsky, 2003; Bradley and Schapire, 2008; Barak *et al.*, 2009] cannot be easily generalized to the online setting, necessitating completely different methods not relying on smooth distributions. Our Online BBM algorithm builds on top of a potential based family that arises naturally in the batch setting as approximate minimax optimal algorithms for so-called drifting games [Schapire, 2001; Luo and Schapire, 2014]. The decomposition of each example in that framework naturally allows us to generalize it to the online setting where example comes one by one. On the other hand, AdaBoost.OL is derived by viewing boosting from a different angle: loss minimization [Mason *et al.*, 2000; Schapire and Freund, 2012]. The theory of online loss minimization is the key tool for developing AdaBoost.OL.

2 Setup and Assumptions

We describe the formal setup of the task of online classification by boosting. At each time step $t = 1, \ldots, T$, an adversary chooses an example $(\mathbf{x}_t, y_t) \in \mathcal{X} \times \{-1, 1\}$, where $\mathcal{X}$ is the domain, and reveals $\mathbf{x}_t$ to the online learner. The learner makes a prediction on its label $\hat{y}_t \in \{-1, 1\}$, and suffers the 0-1 loss $\mathbf{1}\{\hat{y}_t \neq y_t\}$. As is usual with online algorithms, this prediction may be randomized.

For parameters $\gamma \in (0, \frac{1}{2})$, $\delta \in (0, 1)$, and a constant $S > 0$, the learner is said to be a *weak* online learner with edge γ and *excess loss* S if, for any T and for any input sequence of examples $(\mathbf{x}_t, y_t)$ for $t = 1, 2, \ldots, T$ chosen adaptively, it generates predictions $\hat{y}_t$ such that with probability at least $1 - \delta$,

$$\sum_{t=1}^{T} \mathbf{1}\{\hat{y}_t \neq y_t\} \leq (\tfrac{1}{2} - \gamma)T + S. \tag{1}$$

The excess loss requirement is necessary since an online learner cannot be expected to predict with any accuracy with too few examples. Essentially, the excess loss S yields a kind of sample complexity bound: the weak learner starts obtaining a distinct edge of $\Omega(\gamma)$ over random guessing when $T \gg \frac{S}{\gamma}$. Typically, the dependence of the high probability bound on δ is polylogarithmic in $\frac{1}{\delta}$; thus in the following we will avoid explicitly mentioning δ.

For a given parameter $\epsilon > 0$, the learner is said to be a *strong* online learner with error rate ϵ if it satisfies the same conditions as a weak online learner except that its edge is $\frac{1}{2} - \epsilon$, or in other words, the fraction of mistakes made, asymptotically, is ϵ. Just as for the weak learner, the excess loss S yields a sample complexity bound: the fraction of mistakes made by the strong learner becomes $O(\epsilon)$ when $T \gg \frac{S}{\epsilon}$.

2.1 Discussion of Weak Online Learning Assumption

We now justify our definition of weak online learning, viz. inequality (1). In the standard batch boosting case, the corresponding weak learning assumption (see for example [Schapire and Freund, 2012]) made is that there is an algorithm which, given a training set of examples and an arbitrary distribution on it, generates a hypothesis that has error at most $\frac{1}{2} - \gamma$ on the training data under the given distribution. This statement can be interpreted as making the following two implicit assumptions:

1. (Richness.) Given an edge parameter $\gamma \in (0, \frac{1}{2})$, there is a set of hypotheses, $\mathcal{H}$, such that given any training set (possibly, a multiset) of examples U, there is some hypothesis $h \in \mathcal{H}$ with error at most $\frac{1}{2} - \gamma$, i.e.

$$\sum_{(\mathbf{x},y)\in U} \mathbf{1}\{h(\mathbf{x}) \neq y\} \leq (\tfrac{1}{2} - \gamma)|U|.$$

2. (Agnostic Learnability.) For any $\epsilon \in (0, 1)$, there is an algorithm which, given any training set (possibly, a multiset) of examples U, can compute a nearly optimal hypothesis $h \in \mathcal{H}$, i.e.

$$\sum_{(\mathbf{x},y)\in U} \mathbf{1}\{h(\mathbf{x}) \neq y\} \leq \inf_{h' \in \mathcal{H}} \sum_{(\mathbf{x},y)\in U} \mathbf{1}\{h'(\mathbf{x}) \neq y\} + \epsilon|U|.$$

Our weak online learning assumption can be seen as arising from a direct generalization of the above two assumptions to the online setting. Namely, the richness assumption stays the same, whereas the agnostic learnability of $\mathcal{H}$ assumption is replaced by an agnostic online learnability of $\mathcal{H}$ assumption (c.f. [Ben-David *et al.*, 2009]). I.e., there is an online learning algorithm which, given any sequence of examples, $(\mathbf{x}_t, y_t)$ for $t = 1, 2, \ldots, T$, generates predictions $\hat{y}_t$ such that

$$\sum_{t=1}^{T} \mathbf{1}\{\hat{y}_t \neq y_t\} \leq \inf_{h \in \mathcal{H}} \sum_{t=1}^{T} \mathbf{1}\{h(\mathbf{x}_t) \neq y_t\} + R(T),$$

where $R : \mathbb{N} \to \mathbb{R}_+$ is the regret, a non-decreasing, sublinear function of the number of prediction periods T. Since online learning algorithms are typically randomized, we assume the above bound holds with high probability. The following lemma shows that richness and agnostic online learnability immediately imply our online weak learning assumption (1).[2]

Lemma 1. *Suppose the sequence of examples $(\mathbf{x}_t, y_t)$ is obtained from a data set for which there exists a hypothesis class $\mathcal{H}$ that is both rich for edge parameter 2γ and agnostically online learnable with regret $R(\cdot)$. Then, the agnostic online learning algorithm for $\mathcal{H}$ satisfies the weak learning assumption (1), with edge γ and excess loss $S = \max_T(R(T) - \gamma T)$.*

[2] All proofs can be found in [Beygelzimer *et al.*, 2015].

Algorithm 1 Online BBM

for $t = 1$ **to** T **do**
  Receive example $\mathbf{x}_t$.
  Predict $\hat{y}_t = \text{sign}(\sum_{i=1}^{N} \text{WL}^i(\mathbf{x}_t))$, receive label y_t.
  Set $s_t^0 = 0$.
  for $i = 1$ **to** N **do**
    Set $s_t^i = s_t^{i-1} + y_t \text{WL}^i(\mathbf{x}_t)$.
    Set $k_t^i = \lfloor \frac{N-i-s_t^{i-1}+1}{2} \rfloor$.
    Set $w_t^i = \binom{N-i}{k_t^i} \left(\frac{1}{2} + \frac{\gamma}{2}\right)^{k_t^i} \left(\frac{1}{2} - \frac{\gamma}{2}\right)^{N-i-k_t^i}$.
    Pass training example $(\mathbf{x}_t, y_t)$ to WL^i with probability $p_t^i = \frac{\sqrt{N-i+1}}{4} w_t^i$.
  end for
end for

Various agnostic online learning algorithms are known to have a regret bound of $O(\sqrt{T \ln(\frac{1}{\delta})})$, e.g., Hedge [Freund and Schapire, 1995]. If we use such an online learning algorithm as a weak online learner, then a simple calculation implies, via Lemma 1, that it has excess loss $\Theta(\frac{\ln(\frac{1}{\delta})}{\gamma})$.

3 An Optimal Algorithm

The Online BBM algorithm (see Algorithm 1) is an optimal online boosting algorithm, as indicated in Theorem 1 below. The algorithm (hereafter referred to as "booster") works by maintaining N copies of the weak online learner, for some positive integer N to be specified later. Denote the weak online learners WL^i for $i = 1, 2, \ldots, N$. At time step t, the prediction of i-th weak online learner is given by $\text{WL}^i(\mathbf{x}_t) \in \{-1, 1\}$. Note the slight abuse of notation here: WL^i is *not* a function, rather it is an algorithm with an internal state that is updated as it is fed training examples. Thus, the prediction $\text{WL}^i(\mathbf{x}_t)$ depends on the internal state of WL^i, and for notational convenience we avoid reference to the internal state.

In each round t, the booster works by taking a majority vote of the weak learners' predictions. That is, the prediction is $\text{sign}(\sum_{i=1}^{N} \text{WL}^i(\mathbf{x}_t))$, where $\text{sign}(\cdot)$ is 1 if the argument is nonnegative and -1 otherwise. After making the prediction, the true label y_t is revealed by the environment. The booster then updates WL^i by passing the training example $(\mathbf{x}_t, y_t)$ to WL^i with a carefully chosen sampling probability p_t^i (and not passing the example with the remaining probability). The sampling probability p_t^i is obtained by computing a weight w_t^i and setting $p_t^i = \frac{\sqrt{N-i+1}}{4} w_t^i$.

We can prove the following performance guarantee for the Online BBM algorithm:

Theorem 1. *Given a weak online learning algorithm with edge γ and excess loss S and any target error rate $\epsilon > 0$, the Online BBM boosting algorithm constructs a strong online learning algorithm with error rate ϵ using $O(\frac{1}{\gamma^2} \ln(\frac{1}{\epsilon}))$ copies of the weak online learner, and with excess loss $\tilde{O}(\frac{S}{\gamma} + \frac{1}{\gamma^2})$. Its sample complexity is thus $\tilde{O}(\frac{1}{\epsilon}(\frac{S}{\gamma} + \frac{1}{\gamma^2}))$. Furthermore, if $S \geq \tilde{\Omega}(\frac{1}{\gamma})$, then the number of weak online learners is optimal up to constant factors, and the sample complexity is optimal up to polylogarithmic factors.*

Algorithm 2 AdaBoost.OL

Initialize: $\forall i : v_1^i = 1, \alpha_1^i = 0$.
for $t = 1$ **to** T **do**
  Receive example $\mathbf{x}_t$.
  for $i = 1$ **to** N **do**
    Set $\hat{y}_t^i = \text{sign}(\sum_{j=1}^{i} \alpha_t^j \text{WL}^j(\mathbf{x}_t))$.
  end for
  Randomly pick i_t with $\Pr[i_t = i] \propto v_t^i$.
  Predict $\hat{y}_t = \hat{y}_t^{i_t}$, receive label y_t.
  Set $s_t^0 = 0$.
  for $i = 1$ **to** N **do**
    Set $z_t^i = y_t \text{WL}^i(\mathbf{x}_t)$.
    Set $s_t^i = s_t^{i-1} + \alpha_t^i z_t^i$.
    Set $\alpha_{t+1}^i = \Pi\left(\alpha_t^i + \frac{\eta_t z_t^i}{1+\exp(s_t^i)}\right)$ with $\eta_t = 4/\sqrt{t}$.
    Pass example $(\mathbf{x}_t, y_t)$ to WL^i with probability $p_t^i = w_t^i = 1/(1 + \exp(s_t^{i-1}))$.
    Set $v_{t+1}^i = v_t^i \cdot \exp(-\mathbf{1}\{y_t \neq \hat{y}_t^i\})$.
  end for
end for

The requirement that $S \geq \tilde{\Omega}(\frac{1}{\gamma})$ in the lower bound is not very stringent; this is precisely the excess loss one obtains when using standard online learning algorithms with regret bound $O(\sqrt{T})$, as is explained in the discussion following Lemma 1. Furthermore, since we require the bound (1) to hold with high probability, typical analyses of online learning algorithms will have an $\tilde{O}(\sqrt{T})$ deviation term, which also leads to $S \geq \tilde{\Omega}(\frac{1}{\gamma})$.

4 An Adaptive Algorithm

Although the Online BBM algorithm is optimal, it is unfortunately not adaptive since it requires the knowledge of γ as a parameter, which is unknown ahead of time. As discussed in the introduction, adaptivity is essential to the practical performance of boosting algorithms such as AdaBoost. We now design an adaptive online boosting algorithms using the theory of online loss minimization.

We conceptually define N different "experts" giving advice on what to predict on the current example $\mathbf{x}_t$. In round t, expert i predicts by combining the first i weak learners using a *weighted majority* rule with weights α_t^j for WL^j, as follows:

$$\hat{y}_t^i = \text{sign}(\textstyle\sum_{j=1}^{i} \alpha_t^j \text{WL}^j(\mathbf{x}_t)).$$

Now as in the batch boosting algorithm AdaBoost.L [Schapire and Freund, 2012, Chapter 7], the weight w_t^i for WL^i is obtained by computing the logistic loss of the prediction of expert $i-1$, i.e. $\ell(s_t^{i-1})$, and then setting w_t^i to be the negative derivative of the loss:

$$w_t^i = -\ell'(s_t^{i-1}) = \frac{1}{1 + \exp(s_t^{i-1})} \in [0, 1].$$

Table 1: Performance of various online boosting algorithms on various datasets. The lowest loss attained for each dataset is bolded. The baseline is the loss obtained by running the weak learner, VW, on the data.

Dataset	VW baseline	Online BBM.W	AdaBoost.OL.W	AdaBoost.OL	OSBoost.OCP	OSBoost
20news	0.0812	**0.0775**	0.0777	0.0777	0.0791	0.0801
a9a	0.1509	**0.1495**	0.1497	0.1497	0.1509	0.1505
activity	0.0133	**0.0114**	0.0128	0.0127	0.0130	0.0133
adult	0.1543	**0.1526**	0.1536	0.1536	0.1539	0.1544
bio	0.0035	**0.0031**	0.0032	0.0032	0.0033	0.0034
census	0.0471	**0.0469**	**0.0469**	**0.0469**	**0.0469**	0.0470
covtype	0.2563	**0.2347**	0.2495	0.2450	0.2470	0.2521
letter	0.2295	**0.1923**	0.2078	0.2078	0.2148	0.2150
maptaskcoref	0.1091	**0.1077**	0.1083	0.1083	0.1093	0.1091
nomao	0.0641	**0.0627**	0.0635	0.0635	**0.0627**	0.0633
poker	0.4555	**0.4312**	0.4555	0.4555	0.4555	0.4555
rcv1	0.0487	0.0485	**0.0484**	**0.0484**	0.0488	0.0488
vehv2binary	0.0292	0.0286	0.0291	0.0291	**0.0284**	0.0286

In terms of the weight of WL^i, i.e. α_t^i, ideally we wish to mimic AdaBoost.L and use a fixed α^i for all t such that the total logistic loss is minimized: $\alpha^i = \arg\min_\alpha \sum_{t=1}^T \ell(s_t^{i-1} + \alpha z_t^i)$. Of course this is not possible because α^i depends on the future unknown examples. Nevertheless, it suffices to restrict the α_t^i's to the set $[-2, 2]$ and tune them using *online gradient descent* [Zinkevich, 2003], which allows us perform almost as well as the best fixed choice (α^i) in hindsight. This is implemented in step 13 of the algorithm:

$$\alpha_{t+1}^i = \Pi\left(\alpha_t^i - \eta_t f_t'(\alpha_t^i)\right) = \Pi\left(\alpha_t^i + \frac{\eta_t z_t^i}{1+\exp(s_t^i)}\right),$$

where η_t is a time-varying learning rate and Π represents projection onto the set $[-2, 2]$, i.e., $\Pi(\cdot) = \max\{-2, \min\{2, \cdot\}\}$.

Finally, it remains to specify the algorithm's final prediction $\hat{y}_t$. It turns out that at least *one of the N experts* will have high accuracy. Using the well-known Hedge algorithm [Littlestone and Warmuth, 1994; Freund and Schapire, 1997], we can (almost) achieve the performance of the best expert. This is implemented in steps 7 and 15 of the algorithm.

We call the final resulting algorithm AdaBoost.OL,[3] and summarize it in Algorithm 2. Note that as promised, AdaBoost.OL is an adaptive online boosting algorithm and does not require knowing γ in advance. In fact, in the analysis we do not even have to assume that the weak learners satisfy the bound (1). Instead, define the quantities $\gamma_i \triangleq \frac{\mathbf{w}^i \cdot \mathbf{z}^i}{2\|\mathbf{w}^i\|_1}$ for each weak learner WL^i. This can be interpreted as the (weighted) edge over random guessing that WL^i obtains. Note that γ_i may even be negative, which means flipping the sign of WL^i's predictions performs better than random guessing. Nevertheless, the algorithm can still make accurate predictions even with negative γ_i since it will end up choosing negative weights α_t^i in that case. The performance of AdaBoost.OL is provided below.

Theorem 2. *For any T and N, with high probability, the number of mistakes made by AdaBoost.OL is bounded by*

$$\frac{2T}{\sum_i \gamma_i^2} + \tilde{O}\left(\frac{N^2}{\sum_i \gamma_i^2}\right).$$

Moreover, if the weak learners satisfy (1), then the number of mistakes is bounded by

$$\frac{8T}{\gamma^2 N} + \tilde{O}\left(\frac{N}{\gamma^2} + \frac{S}{\gamma}\right).$$

Thus, in order to achieve error rate ϵ, it suffices to use $N \geq \frac{8}{\epsilon\gamma^2}$ weak learners, giving an excess loss $\tilde{O}(\frac{S}{\gamma} + \frac{1}{\epsilon\gamma^4})$.

5 Experiments

While the focus of this paper is a theoretical investigation of online boosting, we also performed an experimental evaluation. We extended the Vowpal Wabbit open source machine learning system [4] to include the algorithms studied in this paper. Since VW can handle importance weights, we implemented versions of our boosting algorithms (Online BBM.W and AdaBoost.OL.W) that use the probabilities p_t^i as importance weights, as well as OSBoost (using uniform weighting on the weak learners) and OSBoost.OCP from [Chen *et al.*, 2012]. To compare importance weighting versus sampling, we also implemented AdaBoost.OL from Algorithm 2, which samples examples sent to VW with the p_t^i probabilities.

All experiments were done on a diverse collection of 13 publicly available datasets. For each dataset, we performed a random split with 80% of the data used for single-pass training and the remaining 20% for testing. We tuned the learning rate, the number of weak learners, and the edge parameter γ (for all but the two versions of AdaBoost.OL) using progressive validation 0-1 loss on the training set. Progressive validation is a standard online validation technique, where each training example is used for testing before it is used for updating the model [Blum *et al.*, 1999]. Reported in Table 1 is the 0-1 loss on the test set.

It should be noted that the VW baseline is already a strong learner. For most datasets, Online BBM.W had the best performance. The average improvement of Online BBM.W over the baseline was 5.14%. For AdaBoost.OL.W, it was 2.57%. Using sampling in AdaBoost.OL boosts the average to 2.67%. The average improvement for OSBoost.OCP was 1.98%, followed by OSBoost with 1.13%.

[3] O stands for online and L stands for logistic loss.

[4] https://github.com/JohnLangford/vowpal_wabbit/wiki

References

[Barak *et al.*, 2009] Boaz Barak, Moritz Hardt, and Satyen Kale. The uniform hardcore lemma via approximate bregman projections. In *The twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1193–1200, 2009.

[Ben-David *et al.*, 2009] Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic Online Learning. In *COLT 2009*, 2009.

[Beygelzimer *et al.*, 2015] Alina Beygelzimer, Satyen Kale, and Haipeng Luo. Optimal and adaptive algorithms for online boosting. In *Proceedings of the 32st International Conference on Machine Learning*, 2015.

[Blum *et al.*, 1999] Avrim Blum, Adam Kalai, and John Langford. Beating the hold-out: Bounds for k-fold and progressive cross-validation. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, COLT '99, pages 203–208, 1999.

[Bradley and Schapire, 2008] Joseph K. Bradley and Robert E. Schapire. FilterBoost: Regression and classification on large datasets. In *Advances in Neural Information Processing Systems 20*, 2008.

[Bshouty and Gavinsky, 2003] Nader H Bshouty and Dmitry Gavinsky. On boosting with polynomially bounded distributions. *The Journal of Machine Learning Research*, 3:483–506, 2003.

[Chen *et al.*, 2012] Shang-Tse Chen, Hsuan-Tien Lin, and Chi-Jen Lu. An Online Boosting Algorithm with Theoretical Justifications. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[Chen *et al.*, 2014] Shang-Tse Chen, Hsuan-Tien Lin, and Chi-Jen Lu. Boosting with Online Binary Learners for the Multiclass Bandit Problem. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

[Freund and Schapire, 1995] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory: Second European Conference, EuroCOLT '95*, pages 23–37. Springer-Verlag, 1995.

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

[Freund, 1992] Yoav Freund. An improved boosting algorithm and its implications on learning complexity. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pages 391–398, July 1992.

[Freund, 1995] Yoav Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

[Grabner and Bischof, 2006] Helmut Grabner and Horst Bischof. On-line boosting and vision. In *CVPR*, volume 1, pages 260–267, 2006.

[Grabner *et al.*, 2008] Helmut Grabner, Christian Leistner, and Horst Bischof. Semi-supervised on-line boosting for robust tracking. In *ECCV*, pages 234–247, 2008.

[Littlestone and Warmuth, 1994] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

[Liu and Yu, 2007] Xiaoming Liu and Ting Yu. Gradient feature selection for online boosting. In *ICCV*, pages 1–8, 2007.

[Luo and Schapire, 2014] Haipeng Luo and Robert E. Schapire. A Drifting-Games Analysis for Online Learning and Applications to Boosting. In *Advances in Neural Information Processing Systems 27*, 2014.

[Mason *et al.*, 2000] Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. Functional gradient techniques for combining hypotheses. In *Advances in Large Margin Classifiers*. MIT Press, 2000.

[Oza and Russell, 2001] Nikunj C. Oza and Stuart Russell. Online bagging and boosting. In *Eighth International Workshop on Artificial Intelligence and Statistics*, pages 105–112, 2001.

[Schapire and Freund, 2012] Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms*. MIT Press, 2012.

[Schapire, 2001] Robert E. Schapire. Drifting games. *Machine Learning*, 43(3):265–291, June 2001.

[Servedio, 2003] Rocco A. Servedio. Smooth boosting and learning with malicious noise. *Journal of Machine Learning Research*, 4:633–648, 2003.

[Zinkevich, 2003] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on Machine Learning*, 2003.