Optimal and Adaptive Algorithms for Online Boosting

Alina Beygelzimer BEYGEL@YAHOO-INC.COM
Yahoo Labs, New York, NY 10036

Satyen Kale SATYEN@YAHOO-INC.COM
Yahoo Labs, New York, NY 10036

Haipeng Luo HAIPENGL@CS.PRINCETON.EDU
Department of Computer Science, 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 & Freund (2012) for a thorough discussion.

Online learning algorithms receive examples one by one, updating the predictor after seeing each new example. In contrast to the batch setting, online learning algorithms typically don't 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 & Russell, 2001; Grabner & Bischof, 2006; Liu & Yu, 2007; Grabner et al., 2008; Chen et al., 2012; 2014).

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

From a theoretical viewpoint, recent 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. Indeed, the data can even 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 doesn't 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 & 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 (see the lower bounds in Section 3.2).

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]

[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.

Algorithm	N	T
Online BBM (Section 3.1), 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 are actually doing better than the others. The best example of adaptive boosting algorithm is the well-known parameter-free AdaBoost algorithm (Freund & 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 & Gavinsky, 2003; Bradley & 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 & 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 & Freund, 2012). The theory of online loss minimization is the key tool for developing AdaBoost.OL.

Finally, Section 5 presents encouraging experiments on some benchmark datasets.

2. Setup and Assumptions

We describe the formal setup of the task of online classification by boosting. At each time step $t = 1, \dots, T$, 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 $t = 1, 2, \dots, T$ 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 can't 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 $T \gg \frac{S}{\epsilon}$.

Our main theorem is the following:

Theorem 1. *Given a weak online learning algorithm with edge γ and excess loss S and any target error rate $\epsilon > 0$, there is a strong online learning algorithm with error rate ϵ which uses $O(\frac{1}{\gamma^2}\ln(\frac{1}{\epsilon}))$ copies of the weak online learner, and has excess loss $\tilde{O}(\frac{S}{\gamma} + \frac{1}{\gamma^2})$; thus its sample complexity is $\tilde{O}(\frac{1}{\epsilon}(\frac{S}{\gamma} + \frac{1}{\gamma^2}))$. 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.*

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 2. 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 $S \geq \tilde{\Omega}(\frac{1}{\gamma})$.

As the theorem indicates, the strong online learner (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 $\mathrm{WL}^i(\mathbf{x}_t) \in \{-1, 1\}$. 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 $\mathrm{WL}^i(\mathbf{x}_t)$ 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 weighted majority vote of the weak learners' predictions. Specifically, the booster maintains weights $\alpha_t^i \in \mathbb{R}$ for $i = 1, \ldots, N$ corresponding to each weak learner, and its final prediction will then be[2] $\hat{y}_t = \mathrm{sign}(\sum_{i=1}^N \alpha_t^i \mathrm{WL}^i(\mathbf{x}_t))$, where $\mathrm{sign}(\cdot)$ 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[3] $p_t^i = \frac{w_t^i}{\|\mathbf{w}^i\|_\infty}$, where $\mathbf{w}^i = \langle w_1^i, w_2^i, \ldots, w_T^i \rangle$. At the same time the booster updates α_t^i as well, and then it is ready to make a prediction for the next round.

We introduce some more notation to ease the presentation. Let $z_t^i = y_t \mathrm{WL}^i(\mathbf{x}_t)$ and $s_t^i = s_t^{i-1} + \alpha_t^i z_t^i$ with $s_t^0 = 0$. Define $\mathbf{z}^i = \langle z_1^i, z_2^i, \ldots, z_T^i \rangle$. Finally, a martingale concentration bound using (1) yields the following bound. (See Appendix A in the supplementary material for a proof.) The bound can be seen as a weighted version of (1) which is necessary for the rest of the analysis.

Lemma 1. *There is a constant $\tilde{S} = 2S + \tilde{O}(\frac{1}{\gamma})$ such that for any T, with high probability, for every weak learner WL^i we have*

$$\mathbf{w}^i \cdot \mathbf{z}^i \geq \gamma \|\mathbf{w}^i\|_1 - \tilde{S} \|\mathbf{w}^i\|_\infty.$$

2.1. Handling Importance Weights

Typical online learning algorithms can handle *importance weighted* examples: each example $(\mathbf{x}_t, y_t)$ comes with a weight $p_t \in [0, 1]$, and the loss on that example is scaled by p_t, i.e. the loss for predicting $\hat{y}_t$ is $p_t \mathbf{1}\{\hat{y}_t \neq y_t\}$. Consider the following natural extension to the definition of online weak learners which incorporates importance weighted examples: we now require that for any sequence of weighted examples $(\mathbf{x}_t, y_t)$ with weight $p_t \in [0, 1]$ for $t = 1, 2, \ldots, T$, the online learner generates predictions $\hat{y}_t$ such that with probability at least $1 - \delta$,

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

Having access to such a weak learner makes the boosting algorithm simpler: we now simply pass every example $(\mathbf{x}_t, y_t)$ to every weak learner WL^i using the probability $p_t^i = \frac{w_t^i}{\|\mathbf{w}^i\|_\infty}$ as the importance weight. The advantage is that the bound (2) immediately implies the following inequality for any weak learner WL^i, which can be seen as a (stronger) analogue of Lemma 1.

$$\mathbf{w}^i \cdot \mathbf{z}^i \geq 2\gamma \|\mathbf{w}^i\|_1 - 2S \|\mathbf{w}^i\|_\infty. \quad (3)$$

Since the analysis depends only on the bound in Lemma 1, if we use the importance-weighted version of the boosting algorithm, then we can simply use inequality (3) instead in the analysis, which gives a slightly tighter version of Theorem 1, viz. the excess loss can now be bounded by $O(\frac{S}{\gamma})$.

In the rest of the paper, for simplicity of exposition we assume that the p_t^i's are used as sampling probabilities rather than importance weights, and give the analysis using the bound from Lemma 1. In experiments, however, using the p_t^i's as importance weights rather than sampling probabilities led to better performance.

2.2. 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 & 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.

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

[2] In Section 4 a slightly different final prediction will be used.

[3] In the algorithm we simply use a tight-enough upper bound on $\|\mathbf{w}^i\|_\infty$ (such as the bound from Lemma 4) to compute the values p_t^i; we abuse notation here and use $\|\mathbf{w}^i\|_\infty$ to also denote this upper bound.

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):

Lemma 2. *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 $S = \max_T(R(T) - \gamma T)$.*

Proof. For the given sequence of examples $(\mathbf{x}_t, y_t)$ for $t = 1, 2, \ldots, T$, the richness with edge parameter 2γ and agnostic online learnability assumptions on $\mathcal{H}$ imply that with high probability, the predictions $\hat{y}_t$ generated by the agnostic online learning algorithm for $\mathcal{H}$ satisfy

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

It only remains to show that

$$(\tfrac{1}{2} - 2\gamma)T + R(T) \leq (\tfrac{1}{2} - \gamma)T + S,$$

or equivalently, $R(T) \leq \gamma T + S$, which is true by the definition of S. This concludes the proof. □

Various agnostic online learning algorithms are known that have a regret bound of $O(\sqrt{T \ln(\frac{1}{\delta})})$; for example, a standard experts algorithm on a finite hypothesis space such as Hedge. If we use such an online learning algorithm as a weak online learner, then a simple calculation implies, via Lemma 2, that it has excess loss $\Theta(\frac{\ln(\frac{1}{\delta})}{\gamma})$. Thus, by Theorem 1, we obtain an online boosting algorithm with near-optimal sample complexity.

3. An Optimal Algorithm

In this section, we generalize a family of potential based batch boosting algorithms to the online setting. With a specific potential, an online version of boost-by-majority is developed with optimal number of weak learners and near-optimal sample complexity. Matching lower bounds will be shown at the end of the section.

3.1. A Potential Based Family and Boost-By-Majority

In the batch setting, many boosting algorithms can be understood in a unified framework called drifting games (Schapire, 2001). Here, we generalize the analysis and propose a potential based family of online boosting algorithms.

Pick a sequence of $N+1$ non-increasing potential functions $\Phi_i(s)$ such that

$$\begin{aligned} &\Phi_N(s) \geq \mathbf{1}\{s \leq 0\}, \\ &\Phi_{i-1}(s) \geq (\tfrac{1}{2} - \tfrac{\gamma}{2})\Phi_i(s-1) + (\tfrac{1}{2} + \tfrac{\gamma}{2})\Phi_i(s+1). \end{aligned} \tag{4}$$

Then the algorithm is simply to set $\alpha_t^i = 1$ and $w_t^i = \frac{1}{2}(\Phi_i(s_t^{i-1} - 1) - \Phi_i(s_t^{i-1} + 1))$. The following theorem states the error rate bound of this general scheme.

Lemma 3. *For any T and N, with high probability, the number of mistakes made by the algorithm described above is bounded as follows:*

$$\sum_{t=1}^{T} \mathbf{1}\{\hat{y}_t \neq y_t\} \leq \Phi_0(0)T + \tilde{S} \sum_i \|\mathbf{w}^i\|_\infty.$$

Proof. The key property of the algorithm is that for any fixed i and t, one can verify the following:

$$\begin{aligned} &\Phi_i(s_t^i) + w_t^i(z_t^i - \gamma) = \Phi_i(s_t^{i-1} + z_t^i) + w_t^i(z_t^i - \gamma) \\ &= (\tfrac{1}{2} - \tfrac{\gamma}{2})\Phi_i(s_t^{i-1} - 1) + (\tfrac{1}{2} + \tfrac{\gamma}{2})\Phi_i(s_t^{i-1} + 1) \\ &\leq \Phi_{i-1}(s_t^{i-1}) \end{aligned}$$

by plugging the formula of w_t^i, realizing that z_t^i can only be -1 or 1, and using the definition of $\Phi_{i-1}(s)$ from Eq. (4). $t = 1$ to T, we get

$$\sum_{t=1}^{T} \Phi_i(s_t^i) + \mathbf{w}^i \cdot \mathbf{z}^i - \gamma\|\mathbf{w}^i\|_1 \leq \sum_{t=1}^{T} \Phi_{i-1}(s_t^{i-1}).$$

Using Lemma 1, we get

$$\sum_{t=1}^{T} \Phi_i(s_t^i) \leq \sum_{t=1}^{T} \Phi_{i-1}(s_t^{i-1}) + \tilde{S}\|\mathbf{w}^i\|_\infty.$$

which relates the sums of all examples' potential for two successive weak learners. We can therefore apply this inequality iteratively to arrive at:

$$\sum_{t=1}^{T} \Phi_N(s_t^N) \leq \sum_{t=1}^{T} \Phi_0(0) + \tilde{S} \sum_i \|\mathbf{w}^i\|_\infty.$$

Algorithm 1 Online BBM

```
1: for $t = 1$ to $T$ do
2:     Receive example $\mathbf{x}_t$.
3:     Predict $\hat{y}_t = \text{sign}(\sum_{i=1}^N \text{WL}^i(\mathbf{x}_t))$, receive label $y_t$.
4:     Set $s_t^0 = 0$.
5:     for $i = 1$ to $N$ do
6:         Set $s_t^i = s_t^{i-1} + y_t \text{WL}^i(\mathbf{x}_t)$.
7:         Set $k_t^i = \lfloor \frac{N-i-s_t^{i-1}+1}{2} \rfloor$.
8:         Set $w_t^i = \binom{N-i}{k_t^i} \left(\frac{1}{2}+\frac{\gamma}{2}\right)^{k_t^i} \left(\frac{1}{2}-\frac{\gamma}{2}\right)^{N-i-k_t^i}$.
9:         Pass training example $(\mathbf{x}_t, y_t)$ to $\text{WL}^i$
           with probability $p_t^i = \frac{w_t^i}{\|\mathbf{w}^i\|_\infty}$.
10:    end for
11: end for
```

The proof is completed by noting that

$$\Phi_N(s_t^N) \geq \mathbf{1}\{s_t^N \leq 0\} = \mathbf{1}\{\hat{y}_t \neq y_t\}$$

since $y_t \hat{y}_t = \text{sign}(s_t^N)$ by definition. □

Note that the $\tilde{S}\|\mathbf{w}^i\|_\infty$ term becomes a penalty for the final error rate. Therefore, we naturally want this penalty term to be relatively small. This is not necessarily true for any choice of the potential function. For example, if $\Phi_i(s)$ is the exponential potential that leads to a variant of AdaBoost in the batch setting (see Schapire & Freund, 2012, Chap. 13), then the weight w_t^i could be exponentially large.

Fortunately, there is indeed a set of potential functions that produces small weights, which, in the batch setting, corresponds to an algorithm called boost-by-majority (BBM) (Freund, 1995). All we need to do is to let Eq. (4) hold with equality, and direct calculation shows:

$$\Phi_i(s) = \sum_{k=0}^{\lfloor \frac{N-i-s}{2} \rfloor} \binom{N-i}{k} \left(\frac{1}{2}+\frac{\gamma}{2}\right)^k \left(\frac{1}{2}-\frac{\gamma}{2}\right)^{N-i-k},$$

and

$$w_t^i = \frac{1}{2}\binom{N-i}{k_t^i} \left(\frac{1}{2}+\frac{\gamma}{2}\right)^{k_t^i} \left(\frac{1}{2}-\frac{\gamma}{2}\right)^{N-i-k_t^i} \tag{5}$$

where $k_t^i = \lfloor \frac{N-i-s_t^{i-1}+1}{2} \rfloor$ and $\binom{n}{k}$ is defined to be 0 if $k < 0$ or $k > n$. In other words, imagine flipping a biased coin whose probability of heads is $\frac{1}{2} + \frac{\gamma}{2}$ for $N - i$ times. Then $\Phi_i(s)$ is exactly the probability of seeing at most $(N - i - s)/2$ heads and w_t^i is half of the probability of seeing k_t^i heads. We call this algorithm *Online BBM* (see algorithm 1), and the version that uses importance weights on examples (see Section 2.1), Online BBM.W.

One can see that the weights produced by this algorithm are small since trivially $w_t^i \leq 1/2$. However, the following lemma gives a better estimate of $\|\mathbf{w}^i\|_\infty$.

Lemma 4. *If w_t^i is defined as in Eq. (5), then we have $w_t^i = O(1/\sqrt{N-i})$ for any $i < N$.*

This lemma was essentially proven before by Freund (1993, Lemma 2.3.10). We give an alternative and simpler proof in Appendix B in the supplementary material, by using the Berry-Esseen theorem directly. We are now ready to state the main results of Online BBM.

Theorem 2. *For any T and N, with high probability, the number of mistakes made by the Online BBM algorithm is bounded as follows:*

$$\exp(-\tfrac{1}{2}N\gamma^2)T + \tilde{O}(\sqrt{N}(S + \tfrac{1}{\gamma})). \tag{6}$$

Thus, in order to achieve error rate ϵ, it suffices to use $N = \Theta(\frac{1}{\gamma^2}\ln\frac{1}{\epsilon})$ weak learners, which gives an excess loss bound of $\tilde{\Theta}(\frac{S}{\gamma} + \frac{1}{\gamma^2})$.

Proof. A direct application of Hoeffding's inequality gives $\Phi_0(0) \leq \exp(-N\gamma^2/2)$. With Lemma 4 we have

$$\sum_i \|\mathbf{w}^i\|_\infty = O\left(\sum_{i=1}^{N-1} \frac{1}{\sqrt{N-i}}\right) = O(\sqrt{N}).$$

Applying Lemma 3 proves Eq. (6). Now if we set $N = \frac{2}{\gamma^2}\ln\frac{1}{\epsilon}$, then

$$\sum_{t=1}^T \mathbf{1}\{\hat{y}_t \neq y_t\} \leq \epsilon T + \tilde{O}(\sqrt{N}(S+\tfrac{1}{\gamma})) = \epsilon T + \tilde{O}(\tfrac{S}{\gamma}+\tfrac{1}{\gamma^2}).$$

□

3.2. Matching Lower Bounds

We give lower bounds for the number of weak learners and the sample complexity in this section that show that our Online BBM algorithm is optimal up to logarithmic factors. We only give a proof sketch here; detailed calculations appear in appendix C in the supplementary material.

Theorem 3. *For any $\gamma \in (0, \frac{1}{4})$, $S \geq \frac{\ln(\frac{1}{\delta})}{\gamma}$, $\delta \in (0, 1)$ and $\epsilon \in (0, 1)$, there is a weak online learning algorithm with edge γ and excess loss S satisfying (1) with probability at least $1 - \delta$, such that to achieve error rate ϵ, an online boosting algorithm needs at least $\Omega(\frac{1}{\gamma^2}\ln\frac{1}{\epsilon})$ weak learners and a sample complexity of $\Omega(\frac{S}{\epsilon\gamma}) = \Omega(\frac{1}{\epsilon}(\frac{S}{\gamma} + \frac{1}{\gamma^2}))$.*

Proof. (Sketch.) The proof of both lower bounds use a similar construction. In either case, all examples' labels are generated uniformly at random from $\{-1, 1\}$, and in time period t, each weak learner outputs the correct label y_t independently of all other weak learners and other examples with a certain probability p_t to be specified later. For the lower bound on the number of weak learners, we set $p_t = \frac{1}{2} + 2\gamma$, and then the Azuma-Hoeffding inequality implies that the inequality (1) guarantee holds. In this case, the Bayes optimal output of a booster using N weak learners is to simply take a majority vote of all the weak learners (see for instance Schapire & Freund, 2012, Chap. 13.2.6), and the probability that the majority vote is incorrect is $\Theta(\exp(-8N\gamma^2))$. Setting this error to ϵ and solving for N gives the desired lower bound.

Now we turn to the lower bound on the sample complexity. We divide the whole process into two phases: for $t \le T_0 = \frac{S}{4\gamma}$, we set $p_t = \frac{1}{2}$, and for $t > T_0$, we set $p_t = \frac{1}{2} + 2\gamma$. Again the Azuma-Hoeffding inequality implies that the inequality (1) guarantee holds. However, in the first phase (i.e. $t \le T_0$), since the predictions of the weak learners are uncorrelated with the true labels, it is clear that no matter what the booster does, it makes a mistake with probability $\frac{1}{2}$. Thus, it will make $\Omega(T_0)$ mistakes with high probability in the first phase, and thus to achieve ϵ error rate, it needs at least $\Omega(T_0/\epsilon) = \Omega(\frac{S}{\epsilon\gamma})$ examples. □

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. Boosting can be viewed as trying to find a linear combination of weak hypotheses to minimize the total loss of the training examples, usually using functional gradient descent (see for details Schapire & Freund, 2012, Chap. 7). AdaBoost, for instance, minimizes the exponential loss. Here, as discussed before, we intuitively want to avoid using exponential loss since it could lead to large weights. Instead, we will consider logistic loss $\ell(s) = \ln(1 + \exp(-s))$, which results in an algorithm called AdaBoost.L in the batch setting (Schapire & Freund, 2012, Chap. 7).

In the online setting, 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: $\hat{y}_t^i = \text{sign}(\sum_{j=1}^{i} \alpha_t^j \text{WL}^j(\mathbf{x}_t))$. Now as in AdaBoost.L, 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:

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

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, we can almost achieve that using online learning algorithm which allow us perform almost as well as the best fixed choice (α^i) in hindsight.

Specifically, it turns out that it suffices to restrict α to the feasible set $[-2, 2]$. Then consider the following simple one dimensional online learning problem: on each round t, algorithm predicts α_t^i from a feasible set $[-2,2]$; the environment then reveals loss function $f_t(\alpha) = \ell(s_t^{i-1} + \alpha z_t^i)$ and the algorithm suffers loss $f_t(\alpha_t^i)$. There are many so-called "low-regret" algorithms in the literature (see the survey by Shalev-Shwartz (2011)) for this problem ensuring

$$\sum_{t=1}^{T} f_t(\alpha_t^i) - \min_{\alpha\in[-2,2]} \sum_{t=1}^{T} f_t(\alpha) \le R_T^i,$$

where R_T^i is sublinear in T so that on average it goes to 0 when T is large and the algorithm is thus doing almost as well as the best constant choice α^i. The simplest low-regret algorithm in this case is perhaps *online gradient descent* (Zinkevich, 2003):

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

where η_t is a time-varying learning rate and Π represents projection onto the set $[-2, 2]$, i.e., $\Pi(\cdot) = \max\{-2, \min\{2, \cdot\}\}$. Since the loss function is actually 1-Lipschitz ($|f_t'(\alpha)| \le 1$), if we set η_t to be $4/\sqrt{t}$, then standard analysis shows $R_T^i = 4\sqrt{T}$.

Finally, it remains to specify the algorithm's final prediction $\hat{y}_t$. In Online BBM, we simply used the advice of expert N. Unfortunately the algorithm described in this section cannot guarantee that expert N will always make highly accurate predictions. However, as we will show in the proof of Theorem 4, the algorithm does ensure that at least *one of the* N *experts* will have high accuracy. Therefore, what we really need to do is to decide which expert to follow on each round, and try to predict almost as well as the best fixed expert in the hindsight. This is again another classic online learning problem (called expert or hedge problem), and can be solved, for instance, by the well-known Hedge algorithm (Littlestone & Warmuth, 1994; Freund & Schapire, 1997). The idea is to pick an expert on each round randomly with different importance weights according to their previous performance.

We call the final resulting algorithm AdaBoost.OL,[4] and summarize it in Algorithm 2. The version of this algorithm using importance weights on examples (see Sec-

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

Algorithm 2 AdaBoost.OL

```
1: **Initialize:** $\forall i: v_1^i = 1, \alpha_1^i = 0$.
2: **for** $t = 1$ **to** $T$ **do**
3:     Receive example $\mathbf{x}_t$.
4:     **for** $i = 1$ **to** $N$ **do**
5:         Set $\hat{y}_t^i = \text{sign}(\sum_{j=1}^i \alpha_t^j \text{WL}^j(\mathbf{x}_t))$.
6:     **end for**
7:     Randomly pick $i_t$ with $\Pr[i_t = i] \propto v_t^i$.
8:     Predict $\hat{y}_t = \hat{y}_t^{i_t}$, receive label $y_t$.
9:     Set $s_t^0 = 0$.
10:    **for** $i = 1$ **to** $N$ **do**
11:        Set $z_t^i = y_t \text{WL}^i(\mathbf{x}_t)$.
12:        Set $s_t^i = s_t^{i-1} + \alpha_t^i z_t^i$.
13:        Set $\alpha_{t+1}^i = \Pi\left(\alpha_t^i + \frac{\eta_t z_t^i}{1+\exp(s_t^i)}\right)$ with $\eta_t = 4/\sqrt{t}$.
14:        Pass example $(\mathbf{x}_t, y_t)$ to $\text{WL}^i$ with probability[5]
           $p_t^i = w_t^i = 1/(1+\exp(s_t^{i-1}))$.
15:        Set $v_{t+1}^i = v_t^i \cdot \exp(-\mathbf{1}\{y_t \neq \hat{y}_t^i\})$.
16:    **end for**
17: **end for**
```

tion 2.1) is called AdaBoost.OL.W. 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 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 4. *For any T and N, with high probability, the number of mistakes made by AdaBoost.OL is bounded by*

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

Proof. Let the number of mistakes made by expert i be $M_i \triangleq \sum_{t=1}^T \mathbf{1}\{y_t \neq \hat{y}_t^i\}$, also define $M_0 = T$ for convenience. Note that AdaBoost.OL is using a variant of the Hedge algorithm with $\mathbf{1}\{y_t \neq \hat{y}_t^i\}$ being the loss of expert i on round t (Line 7 and 15). So by standard analysis (see e.g. Cesa-Bianchi & Lugosi, 2006, Corollary 2.3), and the Azuma-Hoeffding inequality, we have with high probability

$$\sum_{t=1}^T \mathbf{1}\{y_t \neq \hat{y}_t\} \leq 2\min_i M_i + 2\ln(N) + \tilde{O}(\sqrt{T}). \quad (7)$$

[5]Note that we are using the bound $\|\mathbf{w}^i\|_\infty \leq 1$ here.

Now, whenever expert $i-1$ makes a mistake (i.e. $s_t^{i-1} \leq 0$), we have $w_t^i = 1/(1+\exp(s_t^{i-1})) \geq 1/2$ and therefore

$$\|\mathbf{w}^i\|_1 \geq M_{i-1}/2. \quad (8)$$

Note that Eq. (8) holds even for $i = 1$ by the definition of M_0. We now bound the difference between the logistic loss of two successive experts, $\Delta_i \triangleq \sum_{t=1}^T \left(\ell(s_t^i) - \ell(s_t^{i-1})\right)$. Online gradient descent (Line 13) ensures that

$$\sum_{t=1}^T \ell(s_t^i) \leq \min_{\alpha\in[-2,2]} \sum_{t=1}^T \ell(s_t^{i-1} + \alpha z_t^i) + 4\sqrt{T}, \quad (9)$$

as discussed previously. On the other hand, direct calculation shows $\ell(s_t^{i-1} + \alpha z_t^i) - \ell(s_t^{i-1}) = \ln\left(1 + w_t^i(e^{-\alpha z_t^i} - 1)\right) \leq w_t^i(e^{-\alpha z_t^i} - 1)$. With $\sigma_i \triangleq \sum_{t=1}^T \frac{w_t^i}{\|\mathbf{w}^i\|_1} \mathbf{1}\{z_t^i = 1\} = \frac{1}{2} + \gamma_i$, we thus have

$$\begin{aligned}
&\min_{\alpha\in[-2,2]} \sum_{t=1}^T \left(\ell(s_t^{i-1} + \alpha z_t^i) - \ell(s_t^{i-1})\right) \\
&\leq \min_{\alpha\in[-2,2]} \|\mathbf{w}^i\|_1(\sigma_i e^{-\alpha} + (1-\sigma_i)e^{\alpha} - 1) \\
&\leq -\tfrac{1}{2}\|\mathbf{w}^i\|_1(2\sigma_i - 1)^2 \quad (10)\\
&= -2\gamma_i^2\|\mathbf{w}^i\|_1 \quad (11)\\
&\leq -\gamma_i^2 M_{i-1}. \quad (12)
\end{aligned}$$

Here, inequality (10) follows from Lemma 5 and inequality (12) from inequality (8). The above inequality and inequality (9) imply that

$$\Delta_i \leq -\gamma_i^2 M_{i-1} + 4\sqrt{T}.$$

Summing over $i = 1, \ldots, N$ and rearranging gives

$$\sum_{i=1}^N \gamma_i^2 M_{i-1} + \sum_{t=1}^T \ell(s_t^N) \leq \sum_{t=1}^T \ell(0) + 4N\sqrt{T}$$

which implies that

$$\min_i M_i \leq \min_i M_{i-1} \leq \frac{\ln(2)}{\sum_i \gamma_i^2} T + \frac{4N}{\sum_i \gamma_i^2}\sqrt{T}$$

since $M_i \leq M_0$ for all i, $\ell(s_t^N) \geq 0$ for all t and $\ell(0) = \ln(2)$. Using this bound in inequality (7), we get

$$\begin{aligned}
\sum_{t=1}^T \mathbf{1}\{y_t \neq \hat{y}_t\} &\leq \frac{2\ln(2)T}{\sum_i \gamma_i^2} + \tilde{O}\left(\frac{N\sqrt{T}}{\sum_i \gamma_i^2} + \ln(N)\right) \\
&\leq \frac{2T}{\sum_i \gamma_i^2} + \tilde{O}\left(\frac{N^2}{\sum_i \gamma_i^2}\right),
\end{aligned}$$

where the last inequality follows from the bound $\frac{cN\sqrt{T}}{\sum_i \gamma_i^2} \leq \frac{T}{2\sum_i \gamma_i^2} + \frac{c^2N^2}{2\sum_i \gamma_i^2}$, where c is the hidden $\tilde{O}(1)$ factor in the $\tilde{O}(\frac{N\sqrt{T}}{\sum_i \gamma_i^2})$ term, using the arithmetic mean-geometric mean inequality. □

Figure 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

For the case when the weak learners do satisfy the bound (1), we get the following bound on the number of errors:

Theorem 5. *If the weak learners satisfy (1), then for any T and N, with high probability, the number of mistakes made by AdaBoost.OL 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, which gives an excess loss bound of $\tilde{O}(\frac{S}{\gamma} + \frac{1}{\epsilon\gamma^4})$.

Proof. The proof is on the same lines as that of Theorem 4. The only change is that in inequality (11), we use the bound $\gamma_i^2 \geq \frac{\gamma^2}{4} - \frac{\gamma\tilde{S}}{2\|\mathbf{w}^i\|_1}$ which follows from Lemma 1 using the fact that $a \geq b - c$ implies $a^2 \geq b^2 - 2bc$ for non-negative a, b and c, and the fact that $\|\mathbf{w}^i\|_\infty \leq 1$. This leads to the following change in inequality (12):

$$\min_{\alpha\in[-2,2]} \sum_{t=1}^{T} \left(\ell(s_t^{i-1} + \alpha z_t^i) - \ell(s_t^{i-1})\right) \leq -\frac{\gamma^2}{4} M_{i-1} + \gamma\tilde{S}.$$

Continuing using this bound in the proof and simplifying, we get the stated bound on the number of errors. □

The following lemma is a simple calculation and the proof appears in Appendix D in the supplementary material.

Lemma 5. *For any $\sigma \in [0, 1]$,*

$$\min_{\alpha\in[-2,2]} \sigma e^{-\alpha} + (1-\sigma)e^{\alpha} \leq 1 - \tfrac{1}{2}(2\sigma - 1)^2.$$

Although the number of weak learners and excess loss for Adaboost.OL are suboptimal, the adaptivity of AdaBoost.OL is an appealing feature and leads to good performance in experiments. The possibility of obtaining an algorithm that is both adaptive and optimal is left as an open question.

5. Experiments

While the focus of this paper is a theoretical investigation of online boosting, we have also performed experiments to evaluate our algorithms.

We extended the Vowpal Wabbit open source machine learning system (VW) to include the algorithms studied in this paper. We used VW's default base learning algorithm as our weak learner, tuning only the learning rate. The boosting algorithms implemented were Online BBM.W, AdaBoost.OL.W, OSBoost (using uniform weighting on the weak learners) and OSBoost.OCP from (Chen et al., 2012), all using importance weighted examples in VW, as in Section 2.1. We also implemented AdaBoost.OL from Algorithm 2, which samples examples sent to VW.

All experiments were done on a diverse collection of 13 publically available datasets, described in Appendix E in the supplementary material. 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 is the 0-1 loss on the test set.

It should be noted that the VW baseline is already a strong learner. The results obtained are given in Figure 4. As can be seen, 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%.

References

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

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

Blum, Avrim, Kalai, Adam, and Langford, John. 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, pp. 203–208, 1999.

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

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

Cesa-Bianchi, Nicolò and Lugosi, Gábor. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

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

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

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

Freund, Yoav. *Data Filtering and Distribution Modeling Algorithms for Machine Learning*. PhD thesis, University of California at Santa Cruz, 1993.

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

Freund, Yoav and Schapire, Robert E. 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.

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

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

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

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

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

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

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

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

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

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

Shalev-Shwartz, Shai. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

VW. URL `https://github.com/JohnLangford/vowpal_wabbit/`.

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