Gradient Boosting on Stochastic Data Streams

Hanzhang Hu **Wen Sun** **Arun Venkatraman** **Martial Hebert** **J. Andrew Bagnell**
School of Computer Science, Carnegie Mellon University
{hanzhang, wensun, arunvenk, hebert, dbagnell}@cs.cmu.edu

Abstract

Boosting is a popular ensemble algorithm that generates more powerful learners by linearly combining base models from a simpler hypothesis class. In this work, we investigate the problem of adapting batch gradient boosting for minimizing convex loss functions to online setting where the loss at each iteration is i.i.d sampled from an unknown distribution. To generalize from batch to online, we first introduce the definition of online weak learning edge with which for strongly convex and smooth loss functions, we present an algorithm, Streaming Gradient Boosting (SGB) with exponential shrinkage guarantees in the number of weak learners. We further present an adaptation of SGB to optimize non-smooth loss functions, for which we derive a $O(\ln N/N)$ convergence rate. We also show that our analysis can extend to adversarial online learning setting under a stronger assumption that the online weak learning edge will hold in adversarial setting. We finally demonstrate experimental results showing that in practice our algorithms can achieve competitive results as classic gradient boosting while using less computation.

1 INTRODUCTION

Boosting (Freund and Schapire, 1995) is a popular method that leverages simple learning models (e.g., decision stumps) to generate powerful learners. Boosting has been used to great effect and trump other learning algorithms in a variety of applications. In computer vision, boosting was made popular by the seminal Viola-Jones Cascade (Viola and Jones, 2001) and is still used to generate state-of-the-art results in pedestrian detection (Nam et al., 2014; Yang et al., 2015; Zhu and Peng, 2016). Boosting has also found success in domains ranging from document relevance ranking (Chapelle et al., 2011) and transportation (Zhang and Haghani, 2015) to medical inference (Atkinson et al., 2012). Finally, boosting yields an anytime property at test time, which allows it to work with varying computation budgets (Grubb and Bagnell, 2012) for use in real-time applications such as controls and robotics.

Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54. Copyright 2017 by the author(s).

The advent of large-scale data-sets has driven the need for adapting boosting from the traditional batch setting, where the optimization is done over the whole dataset, to the online setting where the weak learners (models) can be updated with streaming data. In fact, online boosting has received tremendous attention so far. For classification, (Chen et al., 2012; Oza and Russell, 2001; Beygelzimer et al., 2015b) proposed online boosting algorithms along with theoretical justifications. Recent work by Beygelzimer et al. (2015a), addressed the regression task through the introduction of *Online Gradient Boosting* (OGB). We build upon on the developments in (Beygelzimer et al., 2015a) to devise a new set of algorithms presented below.

In this work, we develop streaming boosting algorithms for regression with strong theoretical guarantees under stochastic setting, where at each round the data are i.i.d sampled from some unknown fixed distribution. In particular, our algorithms are streaming extension to the classic gradient boosting (Friedman, 2001), where weak predictors are trained in a stage-wise fashion to approximate the functional gradient of the loss with respect to the previous ensemble prediction, a procedure that is shown by Mason et al. (2000) to be functional gradient descent of the loss in the space of predictors. Since the weak learners cannot match the gradients of the loss exactly, we measure the error of approximation by redefining of *edge* of online weak learners (Beygelzimer et al., 2015b) for online regression setting.

Assuming a non-trivial edge can be achieved by each deployed weak online learner, we develop algorithms to handle smooth or non-smooth loss functions, and theoretically analyze the convergence rates of our streaming boosting algorithms. Our first algorithm targets strongly convex and smooth loss functions and achieves exponential decay on the average regret with respect to the number of weak learners. We show the ratio of the decay depends on the edge and also the condition number of the loss function. The second algorithm, designed for strongly convex but non-smooth loss functions, extends from the batch residual gradient boosting algorithm from (Grubb and Bagnell, 2011). We show that the algorithm achieves $O(\ln N/N)$ convergence rate with respect to the number of weak learners N, which matches the online gradient descent (OGD)'s no-regret rate for strongly convex loss (Hazan et al., 2007). Both of our algorithms promise that as T (the number of samples) and N go to infinity, the average regret converges to zero. Our analysis leverages Online-to-Batch reduction (Cesa-Bianchi et al., 2004; Hazan and Kale, 2014), hence our results naturally extends to adversarial online learning setting as long as the weak online learning edge holds in adversarial setting, a harsher setting than stochastic setting. We conclude with some proof-of-concept experiments to support our analysis. We demonstrate that our algorithm significantly boosts the performance of weak learners and converges to the performance of classic gradient boosting with less computation.

2 RELATED WORK

Online boosting algorithms have been evolving since their batch counterparts are introduced. Oza and Russell (2001) developed some of the first online boosting algorithm, and their work are applied to online feature selection (Grabner and Bischof, 2006) and online semi-supervised learning (Grabner et al., 2008). Leistner et al. (2009) introduced online gradient boosting for the classification setting albeit without a theoretical analysis. Chen et al. (2012) developed the first convergence guarantees of online boosting for classification. Then Beygelzimer et al. (2015b) presented two online classification boosting algorithms that are proved to be respectively optimal and adaptive.

Our work is most related to (Beygelzimer et al., 2015a), which extends gradient boosting for regression to the online setting under a smooth loss: each weak online learner is trained by minimizing a linear loss, and weak learners are combined using Frank-Wolfe (Frank and Wolfe, 1956) fashioned updates. Their analysis generalizes those of batch boosting for regression (Zhang and Yu, 2005). In particular, these proofs forgo edge assumptions of the weak learners. Though Frank-Wolfe is a nice projection-free algorithm, it has relatively slow convergence and usually is restricted to smooth loss functions. In our work, each weak learner instead minimizes the squared loss between its prediction and the gradient, which allows us to treat weak learners as approximations of the gradients thanks to the weak learner edge assumption. Hence we can mimic classic gradient boosting and use a gradient descent approach to combine the weak learners' predictions. These differences enable our algorithms to handle non-smooth convex losses, such as hinge and L_1-losses, and result in convergence bounds that is more analogous to the bounds of classic batch boosting algorithms. This work also differs from (Beygelzimer et al., 2015a) in that we assume an online weak learner edge exists, a common assumption in the classic boosting literature (Freund and Schapire, 1995, 1999) that is extended to the online boosting for classification by (Chen et al., 2012; Beygelzimer et al., 2015b). With this assumption, we analyze online gradient boosting using techniques from gradient descent for convex losses (Hazan et al., 2007).

3 PRELIMINARIES

In the classic online learning setting, at every time step t, the learner $\mathcal{A}$ first makes a prediction (i.e., picks a predictor $f_t \in \mathcal{F}$, where $\mathcal{F}$ is a pre-defined class of predictors) on the input $x_t \in \mathbb{R}^d$, then receives a loss $\ell_t(f_t(x_t))$. The learner then updates f_t to f_{t+1}. The samples (ℓ_t, x_t) could be generated by an adversary, but this work mainly focuses on the setting where $(\ell_t, x_t) \sim D$ are i.i.d sampled from a distribution D. The regret $R_{\mathcal{A}}(T)$ of the learner is defined as the difference between the total loss from the learner and the total loss from the best hypothesis in hindsight under the sequence of samples $\{(\ell_t, x_t)\}_t$:

$$R_{\mathcal{A}}(T) = \sum_{t=1}^{T} \ell_t(f_t(x_t)) - \min_{f^* \in \mathcal{F}} \sum_{t=1}^{T} \ell_t(f^*(x_t)). \quad (1)$$

We say the online learner is *no-regret* if and only if $R_{\mathcal{A}}(T)$ is $o(T)$. That is, time averaged, the online learner predictor f_t is doing as well as the best hypothesis f^* in hindsight. We define *risk* of a hypothesis f as $\mathbb{E}_{(\ell,x)\sim D}[\ell(f(x))]$. Our analysis of the risk leverages the classic Online-to-Batch reduction (Cesa-Bianchi et al., 2004; Hazan and Kale, 2014). The online-to-batch reduction first analyzes regret without the stochastic assumption on the sequence of loss ℓ, and it then relates regret to risk using concentration of measure.

Throughout the paper we will use the concepts of strong convexity and smoothness. A function $\ell(x)$ is said to be λ-strongly convex and β-smooth with respect to norm $\|\cdot\|$ if and only if for any pair x_1 and x_2:

$$\begin{aligned} &\frac{\lambda}{2}\|x_1 - x_2\|^2 \le \ell(x_1) - \ell(x_2) - \nabla\ell(x_2)(x_1 - x_2) \\ &\le \frac{\beta}{2}\|x_1 - x_2\|^2, \end{aligned} \quad (2)$$

where $\nabla \ell(x)$ denotes the gradient of function ℓ with respect to x.

3.1 Online Boosting Setup

Our online boosting setup is similar to (Beygelzimer et al., 2015b) and (Beygelzimer et al., 2015a). At each time step $t = 1, .., T$, the environment picks loss $\ell_t : \mathbb{R}^m \to \mathbb{R}$. The online boosting learner makes a prediction $y_t \in \mathbb{R}^m$ without knowing ℓ_t. Then the learner suffers loss $\ell_t(y_t)$. Throughout the paper we assume the loss is bounded as $|\ell_t(y)| \leq B, B \in \mathbb{R}^+, \forall t, y$. We also assume that the gradient of the loss $\nabla \ell_t(y)$ is also bounded as $\|\nabla \ell_t(y)\| \leq G, G \in \mathbb{R}^+, \forall t, y$.[1] The online boosting learner maintains a sequence of weak online learning algorithms $\mathcal{A}_1, ..., \mathcal{A}_N$. Each weak learner $\mathcal{A}_i$ can only use hypothesis from a restricted hypothesis class $\mathcal{H}$ to produce its prediction $\hat{y}_t^i = h_t^i(x_t)$ ($h : \mathbb{R}^d \to \mathbb{R}^m, \forall h \in \mathcal{H}$), where $h_t^i \in \mathcal{H}$. To make a prediction y_t at each iteration, each $\mathcal{A}_i$ will first make a prediction $\hat{y}_t^i \in \mathbb{R}^m$ where $\hat{y}_t^i = h_t^i(x_t)$. The online boosting learner combines all the weak learners' predictions to produce the final prediction y_t for sample x_t. The online learner then suffers loss $\ell_t(y_t)$ after the loss ℓ_t is revealed. As we will show later, with the loss ℓ_t, the online learner will pass a square loss to each weak learner. Each weak learner will then use its internal no-regret online update procedure to update its own weak hypothesis from h_t^i to h_{t+1}^i. In stochastic setting where ℓ_t and x_t are i.i.d samples from a fixed distribution, the online boosting learner will output a combination of the hypothesises that were generated by weak learners as the final boosted hypothesis for future testing.

By leveraging linear combination of weak learners, the goal of the online boosting learner is to boost the performance of a single online learner $\mathcal{A}_i$. Additionally, we ideally want the prediction error to decrease exponentially fast in the number N of weak learners, as is the result from classic batch gradient boosting (Grubb and Bagnell, 2011).

4 WEAK ONLINE LEARNING

We specifically consider the setting where each weak learner minimizes a square loss $\|y - h(x)\|^2$, where y is the regression target, and h is in the weak-learner hypothesis class $\mathcal{H}$. At each step t, a weak online learner $\mathcal{A}$ chooses a predictor $h_t \in \mathcal{H}$ to predict $h_t(x_t)$, receives the target y_t[2] and then suffers loss $\|y_t - h_t(x_t)\|^2$. With this, we now introduce the definition of Weak Online Learning Edge.

Definition 4.1. ***(Weak Online Learning Edge)*** *Given a restricted hypothesis class $\mathcal{H}$ and a sequence of square losses $\{\|y_t - h(x_t)\|^2\}_t$, the weak online learner predicts a sequence $\{h_t\}$ that has edge $\gamma \in (0, 1]$, such that with high probability $1 - \delta$:*

$$\sum_{t=1}^{T} \|y_t - h_t(x_t)\|^2 \leq (1 - \gamma) \sum_{t=1}^{T} \|y_t\|^2 + R(T), \quad (3)$$

where $R(T) \in o(T)$ is usually known as the excess loss.

The high probability $1 - \delta$ comes from the possible randomness of the weak online learner and the sequence of examples. Usually the dependence of the high probability bound on δ is poly-logarithmic in $1/\delta$ that is included in the term $R(T)$. We will give a concrete example on this edge definition in next section where we will show what $R(T)$ consists of. Intuitively, a larger edge implies that the hypothesis is able to better explain the variance of the learning targets y. Our online weak learning definition is closely related to the one from (Beygelzimer et al., 2015b) in that our definition is an result of the following two assumptions: (1) the online learning problem is agnostic-learnable (i.e., the weak learner has $\frac{o(T)}{T} \to 0$ time-averaged regret against the best hypothesis $h \in \mathcal{H}$) with high probability:

$$\sum_{t=1}^{T} \|y_t - h_t(x_t)\|^2 \leq \min_{h \in \mathcal{H}} \sum_{t=1}^{T} \|y_t - h(x_t)\|^2 + o(T), \quad (4)$$

and (2) the restricted hypothesis class $\mathcal{H}$ is rich enough such that for any sequence of $\{y_t, x_t\}$ with high probability:

$$\min_{h \in \mathcal{H}} \sum_{t=1}^{T} \|y_t - h(x_t)\|^2 \leq (1 - \gamma) \sum_{t=1}^{T} \|y_t\|^2 + o(T). \quad (5)$$

Our definition of online weak learning directly generalizes the batch weak learning definition in (Grubb and Bagnell, 2011) to the online setting by the additional agnostic learnability assumption as shown in Eqn. 4.

Note that we pick square losses (Eqn. 5) in our weak online learning definition. As we will show later, the goal is to enforce that the weak learners to accurately predict gradients, as was also originally used in the batch gradient boosting algorithm (Friedman, 2001). Least-squares losses are also shown to be important in streaming tasks by (Gao et al., 2016) for their superior computational and theoretical properties.

The above online weak learning edge definition immediately implies the following result, which is used in later proofs:

[1]Throughout the paper, the notation $\|x\|$ for any finite dimension vector x stands for the classic L2 norm.

[2]Abuse of notation: in Sec 4, $y_t \in \mathbb{R}^m$ simply stands for a regression target for the weak learner at step t, not the final prediction of the boosted learner defined in Sec. 3.1.

Lemma 4.2. *Given the sequence of losses $\|y_t - h(x_t)\|^2$, $1 \leq t \leq T$, the online weak learner generates a sequence of predictors $\{h_t\}_t$, such that:*

$$\sum_{t=1}^{T} 2y_t^T h_t(x_t) \geq \gamma \sum_{t=1}^{T} \|y_t\|^2 - R(T), \quad \gamma \in (0,1]. \tag{6}$$

The above lemma can be proved by expanding the square on the LHS of Eqn. 3, cancelling common terms and rearranging terms.

4.1 Why Weak Learner Edge is Reasonable?

We demonstrate here that the weak online learning edge assumption is reasonable. Let us consider the case that the hypothesis class $\mathcal{H}$ is closed under scaling (meaning if $h \in \mathcal{H}$, then for all $\alpha \in \mathbb{R}$, $\alpha h \in \mathcal{H}$) and let us assume $x \sim D$, and $y = f^*(x)$ for some unknown function f^*. We define the inner product $\langle h_1, h_2 \rangle$ of any two functions h_1, h_2 as $\mathbb{E}_{x\sim D}[h_1(x)^T h_2(x)]$ and the squared norm $\|h\|^2$ of any function h as $\langle h, h \rangle$. We assume f^* is bounded in a sense $\|f^*(x)\| \leq F \in \mathbb{R}^+$. The following proposition shows that as long as f^* is not perpendicular to the span of $\mathcal{H}$ ($f^* \not\perp span(\mathcal{H})$), i.e., $\exists h \in span(\mathcal{H})$ such that $\langle h, f^* \rangle \neq 0$, then we can achieve a non-zero edge:

Proposition 4.3. *Consider any sequence of pairs $\{x_t, y_t\}_{t=1}^T$, where x_t is i.i.d sampled from D, $y_t = f^*(x_t)$ and $f^* \not\perp span(\mathcal{H})$. Run any no-regret online algorithm $\mathcal{A}$ on sequence of losses $\{\|y_t - h(x_t)\|^2\}_t$ and output a sequence of predictions $\{h_t\}_t$. With probability at least $1-\delta$, there exists a weak online learning edge $\gamma \in (0,1]$, such that:*

$$\sum_{t=1}^{T} \|h_t(x_t) - y_t\|^2 \leq (1-\gamma) \sum_{t=1}^{T} \|y_t\|^2 + R_{\mathcal{A}}(T) + (2-\gamma) O\left(\sqrt{T \ln(1/\delta)}\right),$$

where $R_{\mathcal{A}}(T)$ is the regret of online algorithm $\mathcal{A}$.

The proof of the above proposition can be found in Appendix. Matching to Eq. 3, we have $R(T) = R_{\mathcal{A}}(T) + (2-\gamma)O\left(\sqrt{T\ln(1/\delta)}\right) \in o(T)$. In addition, the contrapositive of the proposition implies that without a positive edge, $span(\mathcal{H})$ is orthogonal to f^* so that no linear boosted ensemble can approximate f^*. Hence having a positive online weak learner edge is necessary for online boosted algorithms.

5 ALGORITHM

5.1 Smooth Loss Functions

We first present Streaming Gradient Boosting (SGB), an algorithm (Alg. 1) that is designed for loss functions $\{\ell_t(y)\}$ that are λ-strongly convex and β-smooth. Alg. 1 is the online version of the classic batch gradient boosting algorithms (Friedman, 2001; Grubb and Bagnell, 2011). Alg. 1 maintains N weak learners. At each time step t, given example x_t, the algorithm predicts y_t by linearly combining the weak learners' predictions (Line 5). Then after receiving loss ℓ_t, for each weak learner, the algorithm computes the gradient of ℓ_t with respect to y evaluated at the *partial* sum y_t^{i-1} (Line 11) and feeds the square loss $l_t(h)$ with the computed gradient as the regression target to weak learner $\mathcal{A}^i$ (Line 12). The weak learner $\mathcal{A}^i$ then performs its own no-regret online update to compute h_i^{t+1} (Line 13).

Algorithm 1 Streaming Gradient Boosting (SGB)

```
1: Input: A restricted class $\mathcal{H}$. $N$ online weak learners $\{\mathcal{A}_i\}_{i=1}^N$. Learning rate $\eta$.
2: Each weak learner initlizes a hypothesis $h_i^1 \in \mathcal{H}, \forall 1 \leq i \leq N$.
3: for t = 1 to T do
4:     Receive $x_t$ and initialize $y_t^0 = y_0$ (e.g., $y_0 = 0$).
5:     for i = 1 to N do
6:         Set the partial sum $y_t^i = y_t^{i-1} - \eta h_i^t(x_t)$.
7:     end for
8:     Predict $y_t = y_t^N$.
9:     $\ell_t$ is revealed and learner suffers loss $\ell_t(y_t)$.
10:    for i = 1 to N do
11:        Compute gradient w.r.t partial sum: $\nabla_i^t = \nabla \ell_t(y_t^{i-1})$.
12:        Feed loss $\|\nabla_i^t - h_i^t(x_t)\|^2$ to $\mathcal{A}^i$.
13:        Weak learner $\mathcal{A}^i$ computes $h_i^{t+1}$ using its no-regret update procedure.
14:    end for
15: end for
16: Set $\bar{h}_i = \frac{1}{T}\sum_{t=1}^T h_i^t, \forall 1 \leq i \leq N$.
17: Return: $\{\bar{h}_1, ..., \bar{h}_N\}$.
```

Line 16 and 17 are needed for stochastic setting. We compute the average $\bar{h}_i$ for every weak learner $\mathcal{A}_i$ in Line 16. In testing time, given $x \sim D$, we predict y as:

$$y = y_0 - \eta \sum_{i=1}^{N} \bar{h}_i(x). \tag{7}$$

Since we penalize the weak learners by the squared deviation of its own prediction and the gradient from the previous partial sum, we essentially force weak learners to produce predictions that are close to the gradients (in a no-regret perspective). With this perspective, SGB can be understood as using the weak learners' predictions as N gradient descent steps where the gradient of each step i is approximated by a weak learner's prediction (Line 5). Let us define $\Delta_0 = \sum_{t=1}^T (\ell_t(y_t^0) - \ell_t(f^*(x_t)))$, for any $f^* \in \mathcal{F}$.

Namely Δ_0 measures the performance of the initialization $\{y_t^0\}_t$. Under our assumption that the loss is bounded, $|\ell_t(x)| \leq B, \forall t, x$, we can simply upper bound Δ_0 as $\Delta_0 \leq 2BT$. Alg. 1 has the following performance guarantee:

Theorem 5.1. *Assume weak learner $\mathcal{A}_i, \forall i$ has weak online learning edge $\gamma \in (0,1]$. Let $f^* = \arg\min_{f\in\mathcal{F}} \sum_t \ell_t(f(x_t))$. There exists a $\eta = \frac{\gamma}{\beta(8-4\gamma)}$, for λ-strongly convex and β-smooth loss functions, ℓ_t, such that when $T \to \infty$, Alg. 1 generates a sequence of predictions $\{y_t\}_t$ where:*

$$\frac{1}{T}[\sum_{t=1}^{T} \ell_t(y_t) - \sum_{t=1}^{T} \ell_t(f^*(x_t))] \leq 2B(1-\frac{\gamma^2\lambda}{16\beta})^N. \quad (8)$$

For stochastic setting where $(x_t, \ell_t) \sim D$ independently, we have when $T \to \infty$:

$$\mathbb{E}\big[\ell\big(y_0 - \eta\sum_{i=1}^{N}\bar{h}_i(x)\big) - \ell(f^*(x))\big] \leq 2B(1-\frac{\gamma^2\lambda}{16\beta})^N. \quad (9)$$

The expectation in Eqn. 9 of the above theorem is taken over the randomness of the sequence of pairs of loss and samples $\{\ell_t, x_t\}_{t=1}^T$ (note that $\bar{h}_i$ is dependent on $\ell_1, x_1, ..., \ell_T, x_T$) and ℓ, x. Theorem 5.1 shows that with infinite amount samples the average regret decreases exponentially as we increase the number of weak learners. This performance guarantee is very similar to classic batch boosting algorithms (Schapire and Freund, 2012; Grubb and Bagnell, 2011), where the empirical risk decreases exponentially with the number of algorithm iterations, i.e., the number of weak learners. Theorem 5.1 mirrors that of Theorem 1 in (Beygelzimer et al., 2015a), which bounds the regret of the Frank-Wolfe-based Online Gradient Boosting algorithm. Our results utilize the additional assumptions that the losses ℓ_t are strongly convex and that the weak learners have edge, allowing us to shrink the average regret exponentially with respect to N, while the average regret in (Beygelzimer et al., 2015a) shrinks in the order of $1/N$ (though this dependency on N is optimal under their setting).

Proof of Theorem 5.1, detailed in Appendix B, weaves our additional assumptions into the proof framework of gradient descent on smooth losses. In particular, using weak learner edge assumption, we derive Lemma 4.2 and the Lemma B.1 to relate parts of the strong smoothness expansion of the losses to the norm-squared of the gradients $\|\nabla\ell_t(y_t^i)\|^2$, which is an upper bound of $2\lambda(\ell_t(y_t^i) - \ell_t(f^*(x_t)))$ due to strong convexity. Using this observation, we can relate the total regret of the ensemble of the first i learners, $\Delta_i = \sum_{t=1}^T(\ell_t(y_t^i) - \ell_t(f^*(x_t)))$, with the regret from using $i+1$ learners, Δ_{i+1}, and show that Δ_{i+1} shrinks Δ_i by a constant fraction while only adding a small term $O(R(T)) \in o(T)$. Solving the recursion on the sequence of Δ_i, we arrive at the final exponentially decaying regret bound in the number of learners.

Remark Due to the weak online learning edge assumption, the regret bound shown in Eqn. 8 and the risk bound shown in Eqn. 9 are stronger than typical bounds in classic online learning, in a sense that we are competing against f^* that could potentially be much more powerful than any hypothesis from $\mathcal{H}$. For instance when the loss function is square loss $\ell(f(x)) = \|f(x) - z\|^2$, Theorem 5.1 essentially shows that the risk of the boosted hypothesis $\mathbb{E}[\|y_0 - \eta\sum_{i=1}^N \bar{h}_i(x) - z\|^2]$ approaches to zero as N approaches to infinity, under the assumption that $\mathcal{A}_i, \forall i$ have no-zero weak learning edge (e.g.,$f^* \in span(\mathcal{H})$). Note that this is analogous to the results of classification based batch boosting (Freund and Schapire, 1995; Grubb and Bagnell, 2011) and online boosting (Beygelzimer et al., 2015b): as number of weak learners increase, the average number of prediction mistakes approaches to zero. In other words, with the corresponding edge assumptions, these batch/online boosting classification algorithms can compete against any arbitrarily powerful classifier that always makes zero mistakes on any given training data.

5.2 Non-smooth Loss Functions

The regret bound shown in Theorem 5.1 only applies for strongly convex and smooth loss functions. In fact, one can show that Alg. 1 will fail for general non-smooth loss functions. We can construct a sequence of non-smooth loss functions and a special weak hypothesis class $\mathcal{H}$, which together show that the regret of Alg. 1 grows linearly in the number of samples, regardless of the number of weak learners. We refer readers to Appendix D for more details.

Our next algorithm, Alg. 2, extends SGB (Alg. 1) to handle strongly convex but non-smooth losses. Instead of training each weak learner to fit the subgradients of non-smooth loss with respect to current prediction, we instead keep track of a residual Δ_i[3] that accumulates the difference between the subgradients, ∇_k, and the fitted prediction $h_k(x_t)$, from $k = 1$ up to $i-1$. Instead of fitting the predictor h_{i+1} to match the subgradient ∇_{i+1}, we fit it to match the sum of the subgradient and the residuals, $\nabla_{i+1} + \Delta_i$. More specifically, in Line 13 of Alg. 2, for each weak learner $\mathcal{A}^i$, we feed a

[3]Note the abusive notation. For the non-smooth loss setting (Alg. 2), Δ_i does not refer to the regret of the ensemble's regret with the i-th as used in the analysis of Alg. 1

Algorithm 2 Streaming Gradient Boosting (SGB) for non-smooth loss (Residual Projection)

```
1: Input: A restricted class $\mathcal{H}$. $N$ online weak learners $\{\mathcal{A}_i\}_{i=1}^N$. Learning rate schedule $\{\eta_i\}_{i=1}^N$.
2: $\forall i, \mathcal{A}_i$ initializes a hypothesis $h_i^1 \in \mathcal{H}$.
3: for t = 1 to T do
4:     Receive $x_t$ and initialize $y_t^0 = y_0$ (e.g., $y_0 = 0$).
5:     for i = 1 to N do
6:         Set the projected partial sum $y_t^i = \Pi_{\mathcal{Y}}(y_t^{i-1} - \eta_i h_i^t(x_t))$.
7:     end for
8:     Predict $y_t = \frac{1}{N}\sum_{i=0}^N y_t^i$
9:     The loss $\ell_t$ is revealed and compute loss $\ell_t(y_t)$.
10:    Set initial residual $\Delta_0^t = 0$.
11:    for i = 1 to N do
12:        Compute subgradient w.r.t. partial sum: $\nabla_i^t = \nabla \ell_t(y_t^{i-1})$.
13:        Feed loss $\left\|(\Delta_{i-1}^t + \nabla_i^t) - h(x)\right\|^2$ to $\mathcal{A}^i$.
14:        Update residual: $\Delta_i^t = \Delta_{i-1}^t + \nabla_i^t - h_i^t(x_t)$.
15:        Weak learner $\mathcal{A}^i$ computes $h_i^{t+1}$ using its no-regret update procedure.
16:    end for
17: end for
18: Return: $h_t^i, 1 \le i \le N, 1 \le t \le T$.
```

Algorithm 3 SGB (Residual Projection) for testing

```
1: Input: Test sample $x$ and $h_t^i, 1 \le i \le N, 1 \le t \le T$ from the output of Alg. 2.
2: for t = 1 to T do
3:     for i = 1 to N do
4:         $y_t^i = \Pi_{\mathcal{Y}}(y_t^{i-1} - \eta_i h_i^t(x))$.
5:     end for
6:     $y_t = \frac{1}{N}\sum_{i=0}^N y_t^i$.
7: end for
8: Predict: $y = \mathcal{T}(x) = \frac{1}{T}\sum_{t=1}^T y_t$.
```

square loss with the sum of residual and the gradient as the regression target. Then Line 14 sets the new the residual Δ_i^t as the difference between the target $(\Delta_{i-1}^t + \nabla_i^t)$ and the weak learner $\mathcal{A}^i$'s prediction $h_i^t(x_t)$.

The last line of Alg. 2 is needed for stochastic setting where $(\ell_t, x_t) \sim D$ i.i.d. In test, given sample $x \sim D$, we predict y using $h_t^i, \forall i, t$ in procedure shown in Alg. 3. For notation simplicity, we denote the testing procedure shown in Alg. 3 as $\mathcal{T}(x)$, which $\mathcal{T}$ explicitly depends on the returns $h_t^i, 1 \le i \le N, 1 \le t \le T$ from SGB (Residual Projection). Since it's impractical to store and apply all TN models, we follow a common stochastic learning technique which uses the final predictor at time T for testing (e.g., Johnson and Zhang (2013)) in the experiment section (i.e., simply set $t = T$ in Line 3 in Alg. 3). In practice, if the learners converge and T is large, the average and final predictions are close.

Intuitively, this approach prevents the weak learners from consistently failing to match a certain direction of the subgradient as the net error in the direction is stored in residual. By the assumption of weak learner edge, the directions will be approximated. We also note that if we assume the subgradients are bounded, then the residual magnitudes increase at most linearly in the number of weak learners. Simultaneously, each weak learner shrinks the residual by at least a constant factor due to the assumption of edge. Hence, we expect the residual to shrink exponentially in the number of learners. Utilizing this observation, we arrive at the following performance guarantee:

Theorem 5.2. *Assume the loss ℓ_t is λ-strongly convex for all t with bounded gradients, $\|\nabla \ell_t(y)\| \le G$ for all y, and each weak learner $\mathcal{A}_i$ has edge $\gamma \in (0, 1]$. Let $\mathcal{F}$ be a function space, and $\mathcal{H} \subset \mathcal{F}$ be a restriction of $\mathcal{F}$ Let $f^* = \arg\min_{f\in\mathcal{F}} \frac{1}{T}\sum_{t=1}^T \ell_t(f(x_t))$ be the optimal predictor in $\mathcal{F}$ in hindsight. Let $c = \frac{2}{\gamma} - 1$. Let step size be $\eta_i = \frac{1}{\lambda i}$. When $T \to \infty$, we have:*

$$\frac{1}{T}\sum_{t=1}^T (\ell_t(y_t) - \ell_t(f^*(x_t))) \le \frac{4c^2G^2}{\lambda N}(1 + \ln N + \frac{1}{8N}). \tag{10}$$

For stochastic setting where $(x_t, \ell_t) \sim D$ independently, when $T \to \infty$ we have:

$$\mathbb{E}\big[\ell(\mathcal{T}(x)) - \ell(f^*(x))\big] \le \frac{4c^2G^2}{\lambda N}(1 + \ln N + \frac{1}{8N}).$$

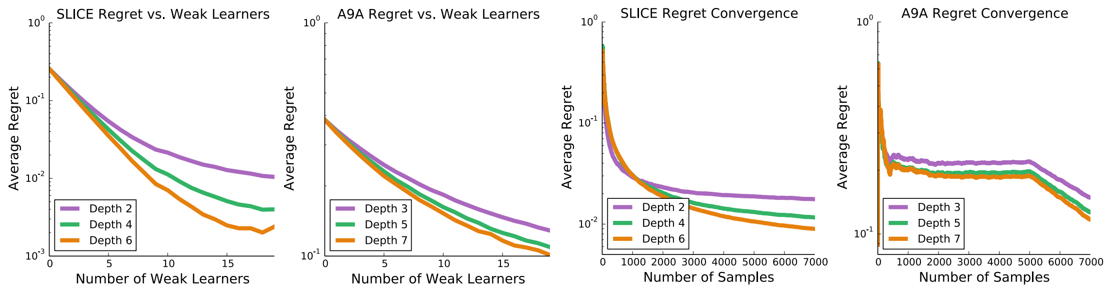
The above theorem shows that the average regret of Alg. 2 is $O(\ln N/N)$ with respect to the number N of weak learners, which matches the regret bounds of Online Gradient Descent for strongly convex loss. The key idea for proving Theorem 5.2 is to combine our online weak learning edge definition with the proof framework of Online Gradient Descent for strongly convex loss functions from (Hazan et al., 2007). The detailed proof can be found in Appendix C.

6 EXPERIMENTS

We demonstrate the performance of our Streaming Gradient Boosting using the following UCI datasets (Lichman, 2013): YEAR, ABALONE, SLICE, and A9A (Kohavi and Becker) as well as the MNIST (LeCun et al., 1998) dataset. If available, we use the given train-test split of each data-set. Otherwise, we create a random 90%-10% train-test split.

6.1 Experimental Analysis of Regret Bounds

We first demonstrate the relationships between the regret bounds shown in Eqn. 8 and the parameters



(a) Regret versus number of weak learners (b) Regret versus number of samples seen

Figure 1: Average regret of SGB with regression trees with various depths on SLICE and A9A datasets.

including the number of weak learners, the number of samples and edge γ. We compute the regret of SGB with respect to a deep regression tree (depth$\geq$ 15), which plays the f^* in Eqn. 8. We use regression trees as the weak learners. We assume that deeper trees have higher edges γ because they empirically fit training data better. We show how the regret relates to the trees' depth, the number of weak learners N (Fig. 1a) and the number of samples T (Fig. 1b).

For the experimental results shown in Fig. 1, we used smooth loss functions with L_2 regularization (see Appendix E for more details). We use logistic loss and square loss for binary classification (A9A) and regression task (SLICE), respectively. For each regression tree weak learner, Follow The Regularized Leader (FTRL) (Shalev-Shwartz, 2011) was used as the no-regret online update algorithm with regularization posed as the depth of the tree. Fig. 1a shows the relationship between the number of weak learners and the average regret given a fixed total number of samples. The average regret decreases as we increase the number of weak learners. We note that the curves are close to linear at the beginning, matching our theoretical analysis that the average regret decays exponentially (note the y-axis is log scale) with respect to the number of weak learners. This shows that SGB can significantly boost the performance of a single weak learner.

To investigate the effect of the edge parameter γ, we additionally compute the average regret in Fig. 1 as the depth of the regression tree is increased. The tree depth increases the model complexity of the base learner and should relate to a larger γ edge parameter. From this experiment, we see that the average regret shrinks as the depth of the trees increases.

Finally, Fig. 1b shows the convergence of the average regret with respect to the number of samples. We see that more powerful weak learners (deeper regression trees) results in faster convergence of our algorithm. We ran Alg. 2 on A9A with hinge loss and SLICE with $L1$ (least absolute deviation) loss and observed very similar results as shown in Fig. 1.

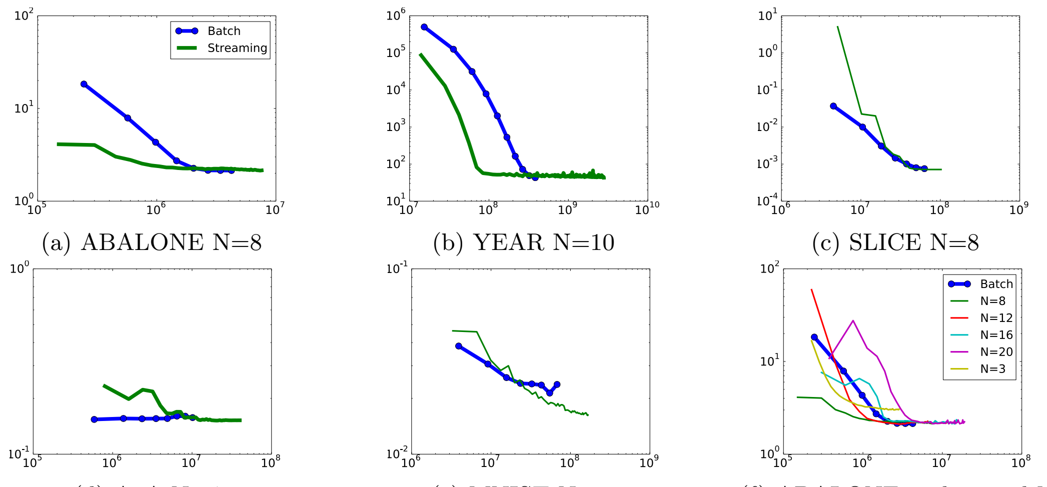
6.2 Batch Boosting vs. Streaming Boosting

We next compare batch boosting to SGB using two-layer neural networks as weak learners[4] and see that SGB reaches similar final performance as the batch boosting algorithm albeit with less training computation. As stated in Sec 5.2, we report h_T^i instead of $\bar{h}_i$ for SGB, since at convergence the average prediction is close to the final prediction, and the latter is impractical to compute. We implement our baseline, the classic batch gradient boosting (**GB**) (Friedman, 2001), by optimizing each weak learner until convergence in order. In both GB and SGB, we train weak learners using ADAM (Kingma and Ba, 2015) optimization and use the default random parameter initialization for NN.

We analyze the complexity of training SGB and GB. We define the prediction complexity of one weak learner as the *unit cost*, since the training run-time complexity almost equates the total complexity of weak learner predictions and updates. Our choice of weak learner and update method (two-layer networks and ADAM) determines that updating a weak learner is about two units cost. In training using SGB, each of the T data samples triggers predictions and updates with all N of the weak learners. This results in a training computational complexity of $3TN = O(TN)$. For GB, let T_B be the samples needed for each weak learner to converge. Then the complexity of training GB is $T_B \sum_{i=1}^N i + 2T_B N \simeq \frac{1}{2} T_B N^2 = O(T_B N^2)$, because when training weak learner i, all previous $i-1$ weak learners must also predict for each data point[5]. Hence, SGB and GB will have the same training complexity if $T_B \simeq \frac{6T}{N} = \Theta(\frac{T}{N})$. In our experiments we observe weak learners typically converge less than $\frac{T}{N}$ samples, but our following experiment shows that SGB still can converge faster overall.

[4]The number of hidden units by data-set: ABALONE, A9A: 1; YEAR, SLICE: 10; MNIST: 5x5 convolution with stride of 2 and 5 output channels. Sigmoid is used as the activation for all except SLICE, which uses leaky ReLU.

[5]Saving previous predictions is disallowed, because data may not be revisited in an actual streaming setting.



(a) ABALONE N=8 (b) YEAR N=10 (c) SLICE N=8

(d) A9A N=4 (e) MNIST N=10 (f) ABALONE with varied N

Figure 2: Log-log plots of test-time loss vs. computation complexity on various data-sets. The x-axis represents computation complexity measured by number of weak leaner predictions; the y-axis measures square loss for regression tasks (ABALONE, SLICE and YEAR), and classification error for A9A and MNIST.

Fig. 2 plots the test-time loss versus training computation, measured by the unit cost. Blue dots highlights when the weak learners are added in GB. We first note that SGB successfully converges to the results of GB in all cases, supporting that SGB is a truly a streaming conversion of GB. As it takes many weak learners to achieve good performance on ABALONE and YEAR, we observe that SGB converges with less computation than GB. On A9A, however, GB is more computationally efficient than SGB, because the first weak learner in GB already performs well and learning a single weak learner for GB is faster than simultaneously optimizing all $N = 8$ weak learners with SGB. This suggests that if we initially set N too big, SGB could be less computationally efficient. In fact Fig. 2f shows that very larger N causes slower convergence to the same final error plateau. On the other hand, small N ($N = 3$) results in worse performance. We specify the chosen N for SGB in Fig. 2, and they are around the number of weak learners that GB requires to converge and achieve good performance. We also note that SGB has slower initial progress compared to GB on SLICE in Fig. 2c and MNIST in Fig. 2e. This is an understandable result as SGB has a much larger pool of parameters to optimize. Despite this initial disadvantage, SGB surpasses GB and converges faster overall, suggesting the advantage of updating all the weak learners together. In practice, if we do not have a good guess of N, we can still use SGB to add multiple weak learners at a time in GB to speed up convergence. Table 1 records the test error (square error for regression and error ratio for classification) of the neural network base learner, GB, and SGB. We observe that SGB achieves test errors that are competitive with GB in all cases.

	Base	**GB**	**SGB**
ABALONE (regression)	8.2848	2.1411	2.1532
YEAR (regression)	4.99×10^5	42.8976	43.0573
SLICE (regression)	0.036045	0.000755	0.000713
A9A (classification)	0.1547	0.1579	0.1523
MNIST (classification)	0.163280	0.019320	0.016320

Table 1: Average test-time loss: square error for regression, and error rate for classification.

7 CONCLUSION

In this paper, we present SGB for online convex programming. By introducing an online weak learning edge definition that naturally extends the edge definition from batch boosting to the online setting and by using square loss, we are able to boost the predictions from weak learners in a gradient descent fashion. Our SGB algorithm guarantees exponential regret shrinkage in the number N of weak learners for strongly convex and smooth loss functions. We additionally extend SGB for optimizing non-smooth loss function, which achieves $O(\ln N/N)$ no-regret rate. Finally, experimental results support the theoretical analysis.

Though our SGB algorithm currently utilizes the procedure of gradient descent to combine the weak learners predictions, our online weak learning definition and the design of square loss for weak learners leave open the possibility to leverage other gradient-based update procedures such as accelerated gradient descent, mirror descent, and adaptive gradient descent for combining the weak learners' predictions.

Acknowledgements

This material is based upon work supported in part by: Echo's Grant name, DARPA ALIAS contract number HR0011-15-C-0027, and National Science Foundation Graduate Research Fellowship Grant No. DGE-1252522.

References

E. J. Atkinson, T. M. Therneau, L. J. Melton, J. J. Camp, S. J. Achenbach, S. Amin, and S. Khosla. Assessing fracture risk using gradient boosting machine (gbm) models. *Journal of Bone and Mineral Research*, 2012.

A. Beygelzimer, E. Hazan, S. Kale, and H. Luo. Online gradient boosting. In *NIPS*, pages 2449–2457, 2015a.

A. Beygelzimer, S. Kale, and H. Luo. Optimal and adaptive algorithms for online boosting. In *ICML*, pages 2323–2331, 2015b.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

O. Chapelle, Y. Chang, and T. Liu, editors. *Proceedings of the Yahoo! Learning to Rank Challenge, held at ICML 2010*, volume 14 of *JMLR Proceedings*, 2011.

S.-T. Chen, H.-T. Lin, and C.-J. Lu. An online boosting algorithm with theoretical justifications. In *ICML*, 2012.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

Y. Freund and R. E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23–37. Springer, 1995.

Y. Freund and R. E. Schapire. A short introduction to boosting. In *Journal of Japanese Society for Artificial Intelligence*, 1999.

J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

W. Gao, L. Wang, R. Jin, S. Zhu, and Z.-H. Zhou. One-pass auc optimization. In *Artificial Intelligence Journal*, volume 236, pages 1–29, 2016.

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

H. Grabner, C. Leistner, and H. Bischof. Semisupervised on-line boosting for robust tracking. In *ECCV*, page 234 247, 2008.

A. Grubb and D. Bagnell. Generalized boosting algorithms for convex optimization. In *ICML*, 2011.

A. Grubb and D. Bagnell. Speedboost: Anytime prediction with uniform near-optimality. In *AISTATS*, pages 458–466, 2012.

E. Hazan and S. Kale. Beyond the regret minimization barrier: Optimal algorithms for stochastic strongly-convex optimization. *Journal of Machine Learning Research*, 15: 2489–2512, 2014.

E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192, 2007.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26*, 2013.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR, arXiv:1412.6980*, 2015.

R. Kohavi and B. Becker. Adult data set. UCI Machine Learning Repository.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

C. Leistner, A. Saffari, P. M. Roth, and H. Bischof. On robustness of on-line boosting - a competitive study. In *ICCV Workshop on On-line Learning for Computer Vision*, 2009.

M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithms as gradient descent. In *NIPS*, 2000.

W. Nam, P. Dollár, and J. H. Han. Local decorrelation for improved pedestrian detection. In *NIPS*, pages 424–432, 2014.

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

R. E. Schapire and Y. Freund. *Boosting: Foundations and algorithms*. MIT press, 2012.

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

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, volume 1. IEEE, 2001.

B. Yang, J. Yan, Z. Lei, and S. Z. Li. Convolutional channel features. In *ICCV*, pages 82–90, 2015.

T. Zhang and B. Yu. Boosting with early stopping: Convergence and consistency. 33:15381579, 2005.

Y. Zhang and A. Haghani. A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies*, 58:308–324, 2015.

C. Zhu and Y. Peng. Group cost-sensitive boosting for multi-resolution pedestrian detection. In *AAAI*, 2016.