Generalized Boosting Algorithms for Convex Optimization

Alexander Grubb AGRUBB@CMU.EDU
J. Andrew Bagnell DBAGNELL@RI.CMU.EDU
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA

Abstract

Boosting is a popular way to derive powerful learners from simpler hypothesis classes. Following previous work (Mason et al., 1999; Friedman, 2000) on general boosting frameworks, we analyze gradient-based descent algorithms for boosting with respect to any convex objective and introduce a new measure of weak learner performance into this setting which generalizes existing work. We present the first weak to strong learning guarantees for the existing gradient boosting work for smooth convex objectives, and also demonstrate that this work fails for non-smooth objectives. To address this issue, we present new algorithms which extend this boosting approach to arbitrary convex loss functions and give corresponding weak to strong convergence results. In addition, we demonstrate experimental results that support our analysis and demonstrate the need for the new algorithms we present.

1. Introduction

Boosting (Schapire, 2002) is a versatile meta-algorithm for combining together multiple simple hypotheses, or weak learners, to form a single complex hypothesis with superior performance. The power of this meta-algorithm lies in its ability to craft hypotheses which can achieve arbitrary performance on training data using only weak learners that perform marginally better than random. This *weak to strong* learning guarantee is a critical feature of boosting.

To date, much of the work on boosting has focused on optimizing the performance of this meta-algorithm with respect to specific loss functions and problem settings. The AdaBoost algorithm (Freund & Schapire, 1997) is perhaps the most well known and most successful of these. AdaBoost focuses specifically on the task of classification via the minimization of the exponential loss by boosting weak binary classifiers together, and can be shown to be near optimal in this setting. Looking to extend upon the success of AdaBoost, related algorithms have been developed for other domains, such as RankBoost (Freund et al., 2003) and mutliclass extensions to AdaBoost (Mukherjee & Schapire, 2010). Each of these algorithms provides both strong theoretical and experimental results for their specific domain, including corresponding weak to strong learning guarantees, but extending boosting to these and other new settings is non-trivial.

Recent attempts have been successful at generalizing the boosting approach to certain broader classes of problems, but their focus is also relatively restricted. Mukherjee and Schapire (2010) present a general theory of boosting for multiclass classification problems, but their analysis is restricted to the multiclass setting. Zheng et al. (2007) give a boosting method which utilizes the second-order Taylor approximation of the objective to optimize smooth, convex losses. Unfortunately, the corresponding convergence result for their algorithm does not exhibit the typical weak to strong guarantee seen in boosting analyses and their results apply only to weak learners which solve the weighted squared regression problem.

Other previous work on providing general algorithms for boosting has shown that an intuitive link between algorithms like AdaBoost and gradient descent exists (Mason et al., 1999; Friedman, 2000), and that many existing boosting algorithms can be reformulated to fit within this gradient boosting framework. Under this view, boosting algorithms are seen as performing a modified gradient descent through the space of all hypotheses, where the gradient is calculated and then used to find the weak learner which will provide the best descent direction. Unfortunately, these previous attempts failed to make explicit all the theory connect-

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.
Copyright 2011 by the author(s)/owner(s).

ing these two areas and were unable to provide weak to strong convergence results for this general setting.

Our work aims to rigorously define the mathematics underlying this connection and show how standard boosting notions such as that of weak learner performance can be extended to the general case. Using this foundation, we will present the first weak to strong learning results for the existing gradient boosting algorithm (Mason et al., 1999; Friedman, 2000) for the special case of smooth convex objectives.

Furthermore, we will also demonstrate that this existing algorithm can fail to converge on non-smooth objectives. To rectify this issue, we present new algorithms which do have corresponding strong convergence guarantees for all convex objectives, and demonstrate experimentally that these new algorithms often outperform the existing algorithm in practice.

Our analysis is modeled after existing work on gradient descent algorithms for optimizing over vector spaces. For convex problems standard gradient descent algorithms are known to provide good convergence results (Zinkevich, 2003; Boyd & Vandenberghe, 2004; Hazan et al., 2006) and are widely applicable. However, as detailed above, the modified gradient descent procedure which corresponds to boosting does not directly follow the gradient, instead selecting a descent direction from a restricted set of allowable search directions. This *restricted gradient descent* procedure requires new extensions to the previous work on gradient descent optimization algorithms.

A related form of gradient descent with gradient errors has previously been studied in the analysis of budgeted learning (Sutskever, 2009), and general results related to gradient projection errors are given in the literature. While these results apply to the boosting setting, they lack any kind of weak to strong guarantee. Conversely, we are primarily interested in studying what algorithms and assumptions are needed to overcome projection error and achieve strong final performance even in the face of mediocre weak learner performance.

The rest of the paper is as follows. We first explicitly detail the Hilbert space of functions and various operations within this Hilbert space. Then, we discuss how to quantify the performance of a weak learner in terms of this vector space. Following that, we present theoretical weak to strong learning guarantees for both the existing and our new algorithms. Finally we provide experimental results comparing all algorithms discussed on a variety of tasks.

2. L^2 Function Space

Previous work (Mason et al., 1999; Friedman, 2000) has presented the theory underlying function space gradient descent in a variety of ways, but never in a form which is convenient for convergence analysis. Recently, Ratliff (2009) proposed the L^2 function space as a natural match for this setting. This representation as a vector space is particularly convenient as it dovetails nicely with the analysis of gradient descent based algorithms.

Given a measurable input set $\mathcal{X}$, an output vector space $\mathcal{V}$, and measure μ, the function space $L^2(\mathcal{X}, \mathcal{V}, \mu)$ is the set of all functions $f : \mathcal{X} \to \mathcal{V}$ such that the Lebesgue integral

$$\int_{\mathcal{X}} \|f(x)\|_{\mathcal{V}}^2 \, d\mu \tag{1}$$

is finite. We will specifically consider the special case where μ is a probability measure P with density function $p(x)$, so that (1) is equivalent to $\mathbb{E}_P[\|f(x)\|^2]$.

This Hilbert space has a natural inner product and norm:

$$\begin{aligned} \langle f, g \rangle_P &= \int_{\mathcal{X}} \langle f(x), g(x) \rangle_{\mathcal{V}} \, p(x) \, dx \\ &= \mathbb{E}_P[\langle f(x), g(x) \rangle_{\mathcal{V}}] \\ \|f\|_P^2 &= \langle f, f \rangle_P \\ &= \mathbb{E}_P[\|f(x)\|_{\mathcal{V}}^2]. \end{aligned}$$

We parameterize these operations by P to denote their reliance on the underlying data distribution. In the case of the empirical probability distribution $\hat{P}$ these quantities are simply the corresponding empirical expected value. For example, the inner product becomes

$$\langle f, g \rangle_{\hat{P}} = \frac{1}{N} \sum_{n=1}^{N} \langle f(x_n), g(x_n) \rangle_{\mathcal{V}}$$

In order to perform gradient descent over such a space, we need to compute the gradient of functionals over said space. We will use the notion of a *subgradient* to allow for optimization of non-smooth functions. The standard subgradient definition:

$$\mathcal{R}[f] \geq \mathcal{R}[g] + \langle f - g, \nabla \mathcal{R}[f] \rangle_P$$

where $\nabla \mathcal{R}[f]$ is a (function space) subgradient of the functional $\mathcal{R} : L^2(P) \to \mathbb{R}$ at f, makes these subgradients relatively straightforward to compute for a number of functionals.

For example, for the point-wise loss over a set of training examples,

$$\mathcal{R}_{\text{emp}}[f] = \frac{1}{N}\sum_{n=1}^{N} l(f(x_n), y_n)$$

the subgradients in $L^2(\mathcal{X}, \mathcal{V}, \hat{P})$ are the set:

$$\nabla\mathcal{R}_{\text{emp}}[f] = \{g \mid g(x_n) \in \nabla l(f(x_n), y_n)\}$$

where $\nabla l(f(x_n), y_n)$ is the set of subgradients of the pointwise loss l with respect to $f(x_n)$. For differentiable l, this is just the partial derivative of l with respect to input $f(x_n)$.

Similarly the expected loss,

$$\mathcal{R}[f] = \mathbb{E}_P[\mathbb{E}_{\mathcal{Y}}[l(f(x), y(x))]],$$

has the following subgradients in $L^2(\mathcal{X}, \mathcal{V}, P)$:

$$\nabla\mathcal{R}[f] = \{g \mid g(x) \in \mathbb{E}_{\mathcal{Y}}[\nabla l(f(x), y)]\}.$$

3. Restricted Gradient Descent

We now outline the gradient-based view of boosting (Mason et al., 1999; Friedman, 2000) and how it relates to gradient descent. In contrast to the standard gradient descent algorithm, boosting is equivalent to what we will call the *restricted gradient descent* setting, where the gradient is not followed directly, but is instead replaced by another search direction from a set of allowable descent directions. We will refer to this set of allowable directions as the *restriction set*.

From a practical standpoint, a projection step is necessary when optimizing over function space because the functions representing the gradient directly are difficult to represent and do not generalize to new inputs well. In terms of the connection to boosting, the restriction set corresponds directly to the set of hypotheses generated by a weak learner.

We are primarily interested in two aspects of this restricted gradient setting: first, appropriate ways to find the best allowable direction of descent, and second, a means of quantifying the performance of a restriction set. Conveniently, the function space view of boosting provides a simple geometric explanation for these concerns.

Given a gradient ∇ and candidate direction h, the closest point h' along h can be found using vector projection:

$$h' = \frac{\langle \nabla, h\rangle}{\|h\|^2} h \tag{2}$$

Algorithm 1 Naive Gradient Projection Algorithm

Given: starting point f_0, step size schedule $\{\eta_t\}_{t=1}^{T}$

for $t = 1, \ldots, T$ **do**
  Compute subgradient $\nabla_t \in \nabla\mathcal{R}[f]$.
  Project ∇_t onto hypothesis space $\mathcal{H}$, finding nearest direction h^*.
  Update f: $f_t \leftarrow f_{t-1} - \eta_t \frac{\langle h^*, \nabla_t\rangle}{\|h^*\|^2} h^*$.
end for

Now, given a set of possible descent directions $\mathcal{H}$ the vector h^* which minimizes the resulting projection error (2) also maximizes the projected length:

$$h^* = \arg\max_{h\in\mathcal{H}} \frac{\langle \nabla, h\rangle}{\|h\|}. \tag{3}$$

This is a generalization of the projection operation in Mason et al. (1999) to functions other than classifiers.

Alternatively, one can find h^* by directly minimizing the distance between ∇ and h^*,

$$h^* = \arg\min_{h\in\mathcal{H}} \|\nabla - h\|^2 \tag{4}$$

thereby reducing the final projected distance found using (2). This projection operation is equivalent to the one given by Friedman (2000).

These two projection methods provide relatively simple ways to search over any restriction set for the 'best' descent direction. The straightforward algorithm (Mason et al., 1999; Friedman, 2000) for peforming restricted gradient descent which uses these projection operations is given in Algorithm 1.

In order to analyze the restricted gradient descent algorithms, we need a way quantify the relative strength of a given restriction set. A guarantee on the performance of each projection step, typically referred to in the traditional boosting literature as the *edge* of a given weak learner is crucial to the convergence analysis of restricted gradient algorithms.

For the projection which maximizes the inner product as in (3), we can use the generalized geometric notion of angle to bound performance by requiring that

$$\langle \nabla, h\rangle \geq \cos\theta \|\nabla\| \|h\|$$

while the equivalent requirement for the norm-based projection in (4) is

$$\|\nabla - h\|^2 \leq (1 - (\cos\theta)^2)\|\nabla\|^2.$$

Parameterizing by $\cos\theta$, we can now concisely define the performance potential of a restricted set of search directions, which will prove useful in later analysis.

Definition 1. *A restriction set $\mathcal{H}$ has* edge γ *if for every projected gradient ∇ there exists a vector $h \in \mathcal{H}$ such that either $\langle \nabla, h \rangle \geq \gamma \|\nabla\| \|h\|$ or $\|\nabla - h\|^2 \leq (1-\gamma^2)\|\nabla\|^2$.*

This definition of edge is parameterized by $\gamma \in [0, 1]$, with larger values of edge corresponding to lower projection error and faster algorithm convergence.

3.1. Relationship to Previous Boosting Work

Though these projection operations apply to any L^2 hypothesis set, they also have convenient interpretations when it comes to specific function classes traditionally used as weak learners in boosting.

For a classification-based weak learner with outputs in $\{-1, +1\}$ and an optimization over single output functions $f : \mathcal{X} \to \mathbb{R}$, projecting as in (3) is equivalent to solving the weighted classification problem over examples $\{x_n, \text{sgn}(\nabla(x_n))\}_{n=1}^N$ and weights $w_n = |\nabla(x_n)|$.

The projection via norm minimization in (4) is directly analogous to solving the regression problem

$$h^* = \arg\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{n=1}^{N} \|\nabla(x_n) - f(x_n)\|^2$$

using the gradient outputs as regression targets.

Similarly, our notion of weak learner performance in Definition 1 can be related to previous work. Like our measure of edge which quantifies performance over the trivial hypothesis $h(x) = 0, \forall x$, previous work has used similar quantities which capture the advantage over baseline hypotheses.

For weak learners which are binary classifiers, as is the case in AdaBoost (Freund & Schapire, 1997), there is an equivalent notion of edge which refers to the improvement in performance over predicting randomly. We can show that Definition 1 is an equivalent measure:

Theorem 1. *For a weak classifier space $\mathcal{H}$ with outputs in $\{-1, +1\}$, the following statements are equivalent: (1) $\mathcal{H}$ has edge γ for some $\gamma > 0$, and (2) for any non-negative weights w_n over training data x_n, there is a classifier $h \in \mathcal{H}$ which achieves an error of at most $(\frac{1}{2} - \frac{\delta}{2}) \sum_n w_n$ for some $\delta > 0$.*

A similar result can be shown for more recent work on multiclass weak learners (Mukherjee & Schapire, 2010) when optimizing over functions with mutliple outputs $f : \mathcal{X} \to \mathbb{R}^k$:

Theorem 2. *For a weak multiclass classifier space $\mathcal{H}$ with outputs in $\{1, \ldots, K\}$, let the modified hypothesis space $\mathcal{H}'$ contain a hypothesis $h' : \mathcal{X} \to \mathbb{R}^K$ for each $h \in \mathcal{H}$ such that $h'(x)_k = 1$ if $h(x) = k$ and $h'(x) = -\frac{1}{K-1}$ otherwise. Then, the following statements are equivalent: (1) $\mathcal{H}'$ has edge γ for some $\gamma > 0$, and (2) $\mathcal{H}$ satisfies the performance over baseline requirements detailed in Theorem 1 of (Mukherjee & Schapire, 2010).*

Proofs and more details on these equivalences can be found in the extended version of the paper (Grubb & Bagnell, 2011).

4. Convergence Analysis

We now focus on analyzing the behavior of variants of the basic restricted gradient descent algorithm shown in Algorithm 1 on problems of the form:

$$\min_{f \in \mathcal{F}} \mathcal{R}[f],$$

where allowable descent directions are taken from some restriction set $\mathcal{H} \subset \mathcal{F}$.

In line with previous boosting work, we will specifically consider cases where the edge requirement in Definition 1 is met for some γ, and seek convergence results where the empirical objective $\mathcal{R}_{\text{emp}}[f_t]$ approaches the optimal training performance $\min_{f \in \mathcal{F}} \mathcal{R}_{\text{emp}}[f]$.

While we consider L^2 function space specifically, the convergence analysis presented can be extended to optimization over any Hilbert space using restricted gradient descent.

4.1. Smooth Convex Optimization

The convergence analysis of Algorithm 1 relies on two critical properties of the objective functional $\mathcal{R}$.

A functional $\mathcal{R}$ is λ-*strongly convex* if $\forall f, f' \in \mathcal{F}$:

$$\mathcal{R}[f'] \geq \mathcal{R}[f] + \langle \nabla \mathcal{R}[f], f' - f \rangle + \frac{\lambda}{2} \|f' - f\|^2$$

for some $\lambda > 0$, and Λ-*strongly smooth* if

$$\mathcal{R}[f'] \leq \mathcal{R}[f] + \langle \nabla \mathcal{R}[f], f' - f \rangle + \frac{\Lambda}{2} \|f' - f\|^2$$

for some $\Lambda > 0$. Using these two properties, we can now derive a convergence result for unconstrained optimization over smooth functions.

Theorem 3. *Let $\mathcal{R}_{emp}$ be a λ-strongly convex and Λ-strongly smooth functional over $L^2(\mathcal{X}, \hat{P})$ space. Let $\mathcal{H} \subset L^2$ be a restriction set with edge γ. Let $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}_{emp}[f]$. Given a starting point f_0 and step size $\eta_t = \frac{1}{\Lambda}$, after T iterations of Algorithm 1 we have:*

$$\mathcal{R}_{emp}[f_T] - \mathcal{R}_{emp}[f^*] \leq (1 - \frac{\gamma^2 \lambda}{\Lambda})^T (\mathcal{R}_{emp}[f_0] - \mathcal{R}_{emp}[f^*]).$$

The result above holds for the fixed step size $\frac{1}{\Lambda}$ as well for step sizes found using a line search along the descent direction. The analysis uses the strong smoothness requirement to obtain a quadratic upper bound on the function and then makes guaranteed progress by selecting the step size which minimizes this bound, with larger gains made for larger values of γ. A complete proof is provided in the extended version of the paper (Grubb & Bagnell, 2011).

Theorem 3 gives, for smooth objective functionals, a convergence rate of $O((1-\frac{\gamma^2\lambda}{\Lambda})^T)$. This is very similar to the $O((1-4\gamma^2)^{\frac{T}{2}})$ convergence of AdaBoost (Freund & Schapire, 1997), with both requiring $O(\log(\frac{1}{\epsilon}))$ iterations to get performance within ϵ of optimal. While the AdaBoost result generally provides tighter bounds, this relatively naive method of gradient projection is able to obtain reasonably competitive convergence results while being applicable to a much wider range of problems. Additionally, the proposed method derives no benefit from loss-specific optimizations and can use a much broader class of weak learners. This comparison is a common scenario within optimization: while highly specialized algorithms can often perform better on specific problems, general solutions often obtain equally impressive results, albeit less efficiently, while requiring much less effort to implement. One such example is the Frank-Wolfe algorithm (1956) for quadratic programming.

Unfortunately, the naive approach to restricted gradient descent breaks down quickly in more general cases such as non-smooth objectives. Consider the following example objective over two points x_1, x_2: $\mathcal{R}[f] = |f(x_1)| + |f(x_2)|$. If we consider the hypothesis set $h \in \mathcal{H}$ such that $h(x_1) \in \{-1, +1\}$ and $h(x_2) = 0$, clearly $\mathcal{H}$ has positive edge with respect to the possible subgradients of $\mathcal{R}$. But, any combination of hypotheses in $\mathcal{H}$ will always leave the output of x_2 unchanged, leading to arbitrarily poor performance.

An algorithm which only ever attempts to project subgradients of $\mathcal{R}$, such as Algorithm 1, will not be able to obtain strong performance results for cases like these. The algorithms in the next section overcome this obstacle by projecting modified versions of the subgradients of the objective at each iteration.

4.2. General Convex Optimization

For the convergence analysis of general convex functions we now switch to using the no-regret learning framework. We now want to minimize the *regret*:

$$\sum_{t=1}^{T} [\mathcal{R}_t[f_t] - \mathcal{R}_t[f^*]]$$

Algorithm 2 Repeated Gradient Projection Algorithm

Given: starting point f_0, step size schedule $\{\eta_t\}_{t=1}^T$

for $t = 1, \dots, T$ **do**
  Compute subgradient $\nabla_t \in \nabla\mathcal{R}[f]$.
  Let $\nabla' = \nabla_t$, $h^* = 0$.
  for $k = 1, \dots, t$ **do**
    Project ∇' onto hypothesis space $\mathcal{H}$, finding nearest direction h_k^*.
    $h^* \leftarrow h^* + \frac{\langle h_k^*, \nabla' \rangle}{\|h_k^*\|^2} h_k^*$.
    $\nabla' \leftarrow \nabla' - h_k^*$.
  end for
  Update f: $f_t \leftarrow f_{t-1} - \eta_t h^*$.
end for

over a sequence of convex functionals $\mathcal{R}_t$, where $f^* = \arg\min_{f \in \mathcal{F}} \sum_{t=1}^T \mathcal{R}_t[f]$ is the fixed hypothesis which minimizes loss in hindsight.

By showing that the average regret approaches 0 as T grows large, for decreasing step sizes and a fixed objective function $\mathcal{R}_t = \mathcal{R}$ for all timesteps t, it can be shown that the optimality gap $\mathcal{R}[f_t] - \mathcal{R}[f^*]$ also approaches 0.

In what follows, we restrict our analysis to the case when $\mathcal{R}_t = \mathcal{R}$. This is because the true online setting typically involves receiving a new dataset at every time t, and hence a different data distribution $\hat{P}_t$, effectively changing the underlying L^2 function space at every time step, making comparison of quantities at different time steps difficult in the analysis. The convergence analysis for the online case is beyond the scope of this paper and is not presented here.

The convergence results to follow are similar to previous convergence results for the standard gradient descent setting (Zinkevich, 2003; Hazan et al., 2006), but with a number of additional error terms due to the gradient projection step. Sutskever (2009) has previously studied the convergence of gradient descent with gradient projection errors using an algorithm similar to Algorithm 1, but the analysis does not focus on the weak to strong learning guarantee we seek. In order to obtain this guarantee we now present two new algorithms.

Our first general convex solution, shown in Algorithm 2, overcomes this issue by using a meta-boosting strategy. At each iteration t instead of projecting the gradient ∇_t onto a single hypothesis h^*, we use the naive algorithm to construct h^* out of a small num-

ber of restricted steps, optimizing over the distance $\|\nabla_t - h^*\|^2$. By increasing the number of weak learners trained at each iteration over time, we effectively decrease the gradient projection error at each iteration. As the average projection error approaches 0, the performance of the combined hypothesis approaches optimal.

Theorem 4. *Let $\mathcal{R}_{emp}$ be a λ-strongly convex functional over $\mathcal{F}$. Let $\mathcal{H} \subset \mathcal{F}$ be a restriction set with edge γ. Let $\|\nabla \mathcal{R}[f]\|_{\hat{P}} \leq G$. Let $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}_{emp}[f]$. Given a starting point f_0 and step size $\eta_t = \frac{2}{\lambda t}$, after T iterations of Algorithm 2 we have:*

$$\frac{1}{T}\sum_{t=1}^{T}[\mathcal{R}_{emp}[f_t] - \mathcal{R}_{emp}[f^*]] \leq \frac{G^2}{\lambda T}(1 + \ln T + \frac{1-\gamma^2}{\gamma^2}).$$

The proof (Grubb & Bagnell, 2011) relies on the fact that as the number of iterations increases, our gradient projection error approaches 0 at the rate given in Theorem 3, causing the behavior of Algorithm 2 to approach the standard gradient descent algorithm. The additional error term in the result is a bound on the geometric series describing the errors introduced at each time step.

Theorem 5. *Let $\mathcal{R}_{emp}$ be a convex functional over $\mathcal{F}$. Let $\mathcal{H} \subset \mathcal{F}$ be a restriction set with edge γ. Let $\|\nabla \mathcal{R}[f]\|_{\hat{P}} \leq G$ and $\|f\|_{\hat{P}} \leq F$ for all $f \in \mathcal{F}$. Let $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}_{emp}[f]$. Given a starting point f_0 and step size $\eta_t = \frac{1}{\sqrt{t}}$, after T iterations of Algorithm 2 we have:*

$$\frac{1}{T}\sum_{t=1}^{T}[\mathcal{R}_{emp}[f_t] - \mathcal{R}_{emp}[f^*]] \leq \frac{F^2}{2\sqrt{T}} + \frac{G^2}{\sqrt{T}} + 2FG\frac{1-\gamma^2}{\gamma^2}.$$

Again, the result is similar to the standard gradient descent result, with an added error term dependent on the edge γ.

An alternative version of the repeated projection algorithm allows for a variable number of weak learners to be trained at each iteration. An accuracy threshold for each gradient projection can be derived given a desired accuracy for the final hypothesis, and this threshold can be used to train weak learners at each iteration until the desired accuracy is reached.

Algorithm 3 gives a second method for optimizing over convex objectives. Like the previous approach, the projection error at each time step is used again in projection, but a new step is not taken immediately to decrease the projection error. Instead, this approach keeps track of the residual error left over after projection and includes this error in the next projection step.

Algorithm 3 Residual Gradient Projection Algorithm

Given: starting point f_0, step size schedule $\{\eta_t\}_{t=1}^T$

Let $\Delta = 0$.
for $t = 1, \ldots, T$ **do**
  Compute subgradient $\nabla_t \in \nabla \mathcal{R}[f]$. $\Delta \leftarrow \Delta + \nabla_t$.
  Project Δ onto hypothesis space $\mathcal{H}$, finding nearest direction h^*.
  Update f: $f_t \leftarrow f_{t-1} - \eta_t \frac{\langle h^*, \Delta \rangle}{\|h^*\|^2} h^*$.
  Update residual: $\Delta \leftarrow \Delta - \frac{\langle h^*, \Delta \rangle}{\|h^*\|^2} h^*$
end for

This forces the projection steps to eventually account for past errors, preventing the possibility of systematic error being adversarially introduced through the weak learner set.

As with Algorithm 2, we can derive similar convergence results for strongly-convex and general convex functionals for this new residual-based algorithm.

Theorem 6. *Let $\mathcal{R}_{emp}$ be a λ-strongly convex functional over $\mathcal{F}$. Let $\mathcal{H} \subset \mathcal{F}$ be a restriction set with edge γ. Let $\|\nabla \mathcal{R}[f]\|_{\hat{P}} \leq G$. Let $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}_{emp}[f]$. Let $c = \frac{2}{\gamma^2}$. Given a starting point f_0 and step size $\eta_t = \frac{1}{\lambda t}$, after T iterations of Algorithm 3 we have:*

$$\frac{1}{T}\sum_{t=1}^{T}[\mathcal{R}[f_t] - \mathcal{R}_{emp}[f^*]] \leq \frac{2c^2G^2}{\lambda T}(1 + \ln T + \frac{2}{T}).$$

Theorem 7. *Let $\mathcal{R}_{emp}$ be a convex functional over $\mathcal{F}$. Let $\mathcal{H} \subset \mathcal{F}$ be a restriction set with edge γ. Let $\|\nabla \mathcal{R}[f]\|_{\hat{P}} \leq G$ and $\|f\|_{\hat{P}} \leq F$ for all $f \in \mathcal{F}$. Let $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}_{emp}[f]$. Let $c = \frac{2}{\gamma^2}$. Given a starting point f_0 and step size $\eta_t = \frac{1}{\sqrt{t}}$, after T iterations of Algorithm 3 we have:*

$$\frac{1}{T}\sum_{t=1}^{T}[\mathcal{R}_{emp}[f_t] - \mathcal{R}_{emp}[f^*]] \leq \frac{F^2}{2\sqrt{T}} + \frac{c^2G^2}{\sqrt{T}} + \frac{c^2G^2}{2T^{\frac{3}{2}}}.$$

Again, the results are similar bounds to those from the non-restricted case. Like the previous proof, the extra terms in the bound come from the penalty paid in projection errors at each time step, but here the residual serves as a mechanism for pushing the error back to later projections. The analysis relies on a bound on the norm of the residual Δ, derived by observing that it is increased by at most the norm of the gradient and then multiplicatively decreased in projection due to the edge requirement. This bound on the size of the residual presents itself in the c term present in the

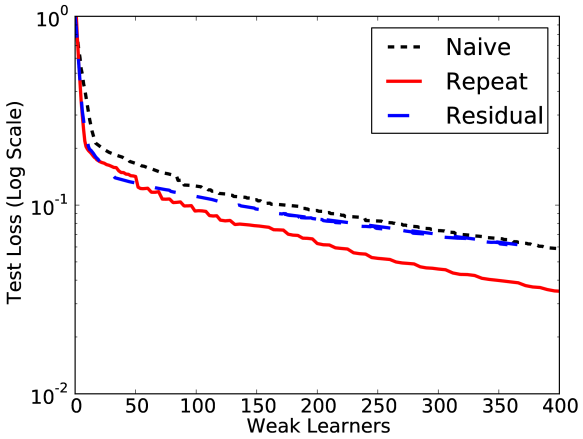

Figure 1. Test set loss vs number of weak learners used for a maximum margin structured imitation learning problem for all three restricted gradient algorithms.

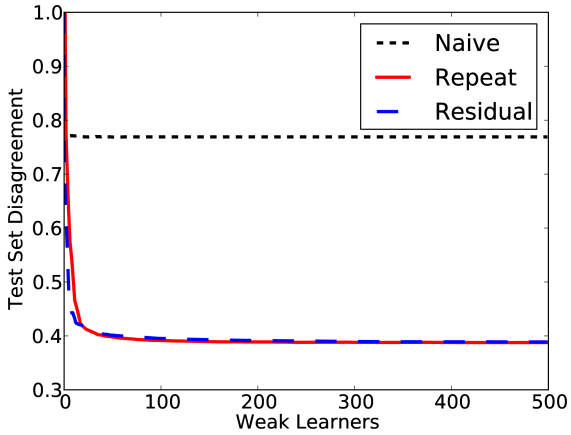

Figure 2. Test set disagreement (fraction of violated constraints) vs number of weak learners used for the MSLR-WEB10K ranking dataset for all three restricted gradient algorithms.

bound. Complete proofs are presented in the extended version of the paper (Grubb & Bagnell, 2011).

In terms of efficiency, these two algorithms are similarly matched. For the strongly convex case, the repeated projection algorithm takes $O(T^2)$ projected steps to obtain an average regret $O(\frac{\ln T}{T} + \frac{1}{\gamma^2 T})$, while the residual algorithm uses $O(T)$ projected steps and has average regret $O(\frac{\ln T}{\gamma^4 T})$. The major difference lies in frequency of the gradient evaluation, where the repeated projection algorithm evaluates the gradient much less often than the than the residual algorithm.

5. Experimental Results

We present preliminary experimental results for these new algorithms on three tasks, an imitation learning problem, a ranking problem and a set of sample classification tasks.

The first experimental setup is an imitation learning problem based on structured prediction called Maximum Margin Planning (Ratliff et al., 2009). In this setting, a demonstrated policy is provided as example behavior and the goal is to learn a cost function over features of the environment which produce policies with similar behavior. This is done by optimizing over a convex, non-smooth loss function which minimizes the difference in costs between the current and demonstrated behavior. Previous attempts in the literature have been made to adapt boosting to this setting (Ratliff et al., 2009; Bradley, 2009), similar to the naive algorithm presented here, but no convergence results for this settings are known.

Figure 1 shows the results of running all three of the algorithms presented here on a sample planning dataset from this domain. The weak learners used were neural networks with 5 hidden units each.

The second experimental setting is a ranking task from the Microsoft Learning to Rank Datasets, specifically MSLR-WEB10K, using the ranking version of the hinge loss and decision stumps as weak learners. Figure 2 shows the test set disagreement (the percentage of violated ranking constraints) plotted against the number of weak learners.

As a final test, we ran our boosting algorithms on several multiclass classification tasks from the UCI Machine Learning Repository (Frank & Asuncion, 2010), using the 'connect4', 'letter', 'pendigits' and 'satimage' datasets. All experiments used the multiclass extension to the hinge loss (Crammer & Singer, 2002), along with multiclass decision stumps for the weak learners.

Of particular interest are the experiments where the naive approach to restricted gradient descent clearly fails to converge. In line with the presented convergence results, both non-smooth algorithms approach optimal training performance at relatively similar rates, while the naive approach cannot overcome this particular condition and fails to achieve strong performance.

Acknowledgements

We would like to thank Kevin Waugh, Daniel Munoz and the ICML reviewers for their helpful feedback.

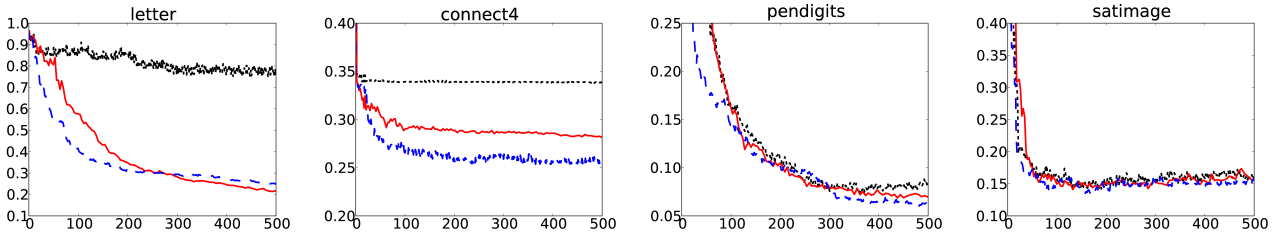

Figure 3. Performance on multiclass classification experiments over the UCI 'connect4', 'letter', 'pendigits' and 'satimage' datasets. The algorithms shown are the naive projection (black dashed line), repeated projection steps (red solid line), and the residual projection algorithm (blue long dashed line).

This work was conducted through collaborative participation in the Robotics Consortium sponsored by the U.S Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement W911NF-10-2-0016.

References

Boyd, S. and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

Bradley, D. M. *Learning in Modular Systems*. PhD thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, 2009.

Crammer, K. and Singer, Y. On the algorithmic implementation of multiclass kernel-based vector machines. *J. Mach. Learn. Res.*, 2:265–292, March 2002.

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

Frank, M. and Wolfe, P. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:95–110, 1956.

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

Freund, Y., Iyer, R., Schapire, R. E., and Singer, Y. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969, 2003.

Friedman, J. H. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29: 1189–1232, 2000.

Grubb, A. and Bagnell, J. A. Generalized boosting algorithms for convex optimization (with proofs). *arXiv*, arXiv:1105.2054 [cs.LG], 2011.

Hazan, E., Kalai, A., Kale, S., and Agarwal, A. Logarithmic regret algorithms for online convex optimization. In *Proceedings of the 19th Annual Conference on Learning Theory*, pp. 499–513, 2006.

Mason, L., Baxter, J., Bartlett, P. L., and Frean, M. Functional gradient techniques for combining hypotheses. In *Advances in Large Margin Classifiers*. MIT Press, 1999.

Mukherjee, I. and Schapire, R. E. A theory of multiclass boosting. In *Advances in Neural Information Processing Systems 22*, Cambridge, MA, 2010. MIT Press.

Ratliff, N. *Learning to Search: Structured Prediction Techniques for Imitation Learning*. PhD thesis, The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, 2009.

Ratliff, N., Silver, D., and Bagnell, J. A. Learning to search: Functional gradient techniques for imitation learning. *Autonomous Robots*, 27(1):25–53, July 2009.

Schapire, R. E. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

Sutskever, I. A simpler unified analysis of budget perceptrons. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 985–992, New York, NY, USA, 2009. ACM.

Zheng, Z., Zha, H., Zhang, T., Chapelle, O., Chen, K., and Sun, G. A general boosting method and its application to learning ranking functions for web search. In *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, 2003.