Optimal and Adaptive Algorithms for Online Boosting

Alina Beygelzimer\textsuperscript{1} \hspace{1cm} \textbf{Satyen Kale}\textsuperscript{1} \hspace{1cm} Haipeng Luo\textsuperscript{2}

\textsuperscript{1}Yahoo! Labs, NYC
\textsuperscript{2}Computer Science Department, Princeton University

December 11, 2015
Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.

Given: a set of training examples.

▶ (“Attn: Beneficiary Contractor Foreign Money Transfer ...”, spam)
▶ (“Let’s meet to discuss QPR –Edo”, not spam)

Obtain a classifier by asking a “weak learning algorithm”:

▶ e.g. contains the word “money” ⇒ spam.

Reweight the examples so that “difficult” ones get more attention.

▶ e.g. spam that doesn’t contain “money”.

Obtain another classifier:

▶ e.g. empty “to address” ⇒ spam.

......

At the end, predict by taking a (weighted) majority vote.
Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.
Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.

- Given: a set of training examples.
  - (“Attn: Beneficiary Contractor Foreign Money Transfer ...”, spam)
  - (“Let’s meet to discuss QPR –Edo”, not spam)
Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.

- Given: a set of training examples.
  - (“Attn: Beneficiary Contractor Foreign Money Transfer ...”, spam)
  - (“Let’s meet to discuss QPR –Edo”, not spam)

- Obtain a classifier by asking a “weak learning algorithm”:
  - e.g. contains the word “money” ⇒ spam.
Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.

- Given: a set of training examples.
  - (“Attn: Beneficiary Contractor Foreign Money Transfer ...”, spam)
  - (“Let’s meet to discuss QPR –Edo”, not spam)

- Obtain a classifier by asking a “weak learning algorithm”:
  - e.g. contains the word “money” ⇒ spam.

- Reweight the examples so that “difficult” ones get more attention.
  - e.g. spam that doesn’t contain “money”.

Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.

- Given: a set of training examples.
  - (“Attn: Beneficiary Contractor Foreign Money Transfer …”, spam)
  - (“Let’s meet to discuss QPR –Edo”, not spam)

- Obtain a classifier by asking a “weak learning algorithm”:
  - e.g. contains the word “money” ⇒ spam.

- Reweight the examples so that “difficult” ones get more attention.
  - e.g. spam that doesn’t contain “money”.

- Obtain another classifier:
  - e.g. empty “to address” ⇒ spam.
Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.

Given: a set of training examples.
  ▶ (“Attn: Beneficiary Contractor Foreign Money Transfer …”, spam)
  ▶ (“Let’s meet to discuss QPR –Edo”, not spam)

Obtain a classifier by asking a “weak learning algorithm”:
  ▶ e.g. contains the word “money” ⇒ spam.

Reweight the examples so that “difficult” ones get more attention.
  ▶ e.g. spam that doesn’t contain “money”.

Obtain another classifier:
  ▶ e.g. empty “to address” ⇒ spam.

......
Boosting: An Example

Idea: combine weak “rules of thumb” to form a highly accurate predictor.

Example: email spam detection.

- **Given:** a set of training examples.
  - ("Attn: Beneficiary Contractor Foreign Money Transfer ...", spam)
  - ("Let’s meet to discuss QPR –Edo”, not spam)

- Obtain a classifier by asking a “weak learning algorithm”:
  - e.g. contains the word “money” ⇒ spam.

- **Reweight** the examples so that “difficult” ones get more attention.
  - e.g. spam that doesn’t contain “money”.

- Obtain another classifier:
  - e.g. empty “to address” ⇒ spam.

- ......

- At the end, predict by taking a (weighted) majority vote.
Online Boosting: Motivation

Boosting is well studied in the **batch setting**, but become **infeasible** when the amount of data is huge.
Online Boosting: Motivation

Boosting is well studied in the **batch setting**, but become **infeasible** when the amount of data is huge.

**Online learning** has proven extremely useful:
- one pass of the data, make prediction on the fly.
Online Boosting: Motivation

Boosting is well studied in the batch setting, but become infeasible when the amount of data is huge.

Online learning has proven extremely useful:

- one pass of the data, make prediction on the fly.
- works even in an adversarial environment.
  - e.g. spam detection.
Online Boosting: Motivation

Boosting is well studied in the batch setting, but become infeasible when the amount of data is huge.

Online learning has proven extremely useful:

- one pass of the data, make prediction on the fly.
- works even in an adversarial environment.
  - e.g. spam detection.

An natural question: how to extend boosting to the online setting?
Related Work

Several algorithms exist (Oza and Russell, 2001; Grabner and Bischof, 2006; Liu and Yu, 2007; Grabner et al., 2008).

- mimic offline counterparts.
- achieve great success in many real-world applications.
- no theoretical guarantees.
Related Work

Several algorithms exist (Oza and Russell, 2001; Grabner and Bischof, 2006; Liu and Yu, 2007; Grabner et al., 2008).

- mimic offline counterparts.
- achieve great success in many real-world applications.
- no theoretical guarantees.

Chen et al. (2012): first online boosting algorithms with theoretical guarantees.

- online analogue of weak learning assumption.
- connecting online boosting and smooth batch boosting.
Batch Boosting

Given a batch of \( T \) examples, \((x_t, y_t) \in X \times \{-1, 1\}\) for \( t = 1, \ldots, T \). Learner \( A \) predicts \( A(x_t) \in \{-1, 1\}\) for example \( x_t \).
Batch Boosting

Given a batch of $T$ examples, $(x_t, y_t) \in \mathcal{X} \times \{-1, 1\}$ for $t = 1, \ldots, T$. Learner $\mathcal{A}$ predicts $\mathcal{A}(x_t) \in \{-1, 1\}$ for example $x_t$.

Weak learner $\mathcal{A}$ (with edge $\gamma$):

$$\sum_{t=1}^{T} 1\{\mathcal{A}(x_t) \neq y_t\} \leq (\frac{1}{2} - \gamma) T$$
Batch Boosting

Given a batch of $T$ examples, $(x_t, y_t) \in \mathcal{X} \times \{-1, 1\}$ for $t = 1, \ldots, T$. Learner $\mathcal{A}$ predicts $\mathcal{A}(x_t) \in \{-1, 1\}$ for example $x_t$.

Weak learner $\mathcal{A}$ (with edge $\gamma$):

$$\sum_{t=1}^{T} \mathbb{1}\{\mathcal{A}(x_t) \neq y_t\} \leq \left(\frac{1}{2} - \gamma\right)T$$

Strong learner $\mathcal{A}'$ (with any target error rate $\epsilon$):

$$\sum_{t=1}^{T} \mathbb{1}\{\mathcal{A}'(x_t) \neq y_t\} \leq \epsilon T$$
Batch Boosting

Given a batch of $T$ examples, $(x_t, y_t) \in \mathcal{X} \times \{-1, 1\}$ for $t = 1, \ldots, T$. Learner $\mathcal{A}$ predicts $\mathcal{A}(x_t) \in \{-1, 1\}$ for example $x_t$.

Weak learner $\mathcal{A}$ (with edge $\gamma$):

$$\sum_{t=1}^{T} 1\{\mathcal{A}(x_t) \neq y_t\} \leq (\frac{1}{2} - \gamma) T$$

$\Downarrow$ Boosting (Schapire, 1990; Freund, 1995)

Strong learner $\mathcal{A}'$ (with any target error rate $\epsilon$):

$$\sum_{t=1}^{T} 1\{\mathcal{A}'(x_t) \neq y_t\} \leq \epsilon T$$
Online Boosting

Examples \((x_t, y_t) \in X \times \{-1, 1\}\) arrive online, for \(t = 1, \ldots, T\).
Learner \(\mathcal{A}\) observes \(x_t\) and predicts \(\mathcal{A}(x_t) \in \{-1, 1\}\) before seeing \(y_t\).

Weak Online learner \(\mathcal{A}\) (with edge \(\gamma\)):

\[
\sum_{t=1}^{T} 1\{\mathcal{A}(x_t) \neq y_t\} \leq (\frac{1}{2} - \gamma)T
\]

Strong Online learner \(\mathcal{A}'\) (with any target error rate \(\epsilon\)):

\[
\sum_{t=1}^{T} 1\{\mathcal{A}'(x_t) \neq y_t\} \leq \epsilon T
\]
Online Boosting

Examples \((x_t, y_t) \in X \times \{-1, 1\}\) arrive online, for \(t = 1, \ldots, T\).
Learner \(A\) observes \(x_t\) and predicts \(A(x_t) \in \{-1, 1\}\) before seeing \(y_t\).

Weak Online learner \(A\) (with edge \(\gamma\) and excess loss \(S\)):

\[
\sum_{t=1}^{T} 1\{A(x_t) \neq y_t\} \leq \left(\frac{1}{2} - \gamma\right)T + S
\]

Strong Online learner \(A'\) (with any target error rate \(\epsilon\) and excess loss \(S'\))

\[
\sum_{t=1}^{T} 1\{A'(x_t) \neq y_t\} \leq \epsilon T + S'
\]
Online Boosting

Examples \((x_t, y_t) \in X \times \{-1, 1\}\) arrive online, for \(t = 1, \ldots, T\). Learner \(A\) observes \(x_t\) and predicts \(A(x_t) \in \{-1, 1\}\) before seeing \(y_t\).

Weak Online learner \(A\) (with edge \(\gamma\) and excess loss \(S\)):

\[
\sum_{t=1}^{T} \mathbb{1}\{A(x_t) \neq y_t\} \leq (\frac{1}{2} - \gamma) T + S
\]

\(\Downarrow\) Online Boosting (our result)

Strong Online learner \(A'\) (with any target error rate \(\varepsilon\) and excess loss \(S'\))

\[
\sum_{t=1}^{T} \mathbb{1}\{A'(x_t) \neq y_t\} \leq \varepsilon T + S'
\]
Online Boosting

Examples \((x_t, y_t) \in X \times \{-1, 1\}\) arrive online, for \(t = 1, \ldots, T\).
Learner \(\mathcal{A}\) observes \(x_t\) and predicts \(\mathcal{A}(x_t) \in \{-1, 1\}\) before seeing \(y_t\).

Weak Online learner \(\mathcal{A}\) (with edge \(\gamma\) and excess loss \(S\)):

\[
\sum_{t=1}^{T} 1\{\mathcal{A}(x_t) \neq y_t\} \leq (\frac{1}{2} - \gamma) T + S
\]

\(\Downarrow\) Online Boosting (our result)

Strong Online learner \(\mathcal{A}'\) (with any target error rate \(\epsilon\) and excess loss \(S'\))

\[
\sum_{t=1}^{T} 1\{\mathcal{A}'(x_t) \neq y_t\} \leq \epsilon T + S'
\]

this talk: \(S = \frac{1}{\gamma}\) (corresponds to \(\sqrt{T}\) regret)
Main Results

Parameters of interest:
\( N = \) number of weak learners (of edge \( \gamma \)) needed to achieve error rate \( \epsilon \).
\( T_\epsilon = \) minimal number of examples s.t. error rate is \( \epsilon \).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>( N )</th>
<th>( T_\epsilon )</th>
<th>Optimal?</th>
<th>Adaptive?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online BBM</td>
<td>( O(\frac{1}{\gamma^2 \ln \frac{1}{\epsilon}}) )</td>
<td>( \tilde{O}(\frac{1}{\epsilon \gamma^2}) )</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>AdaBoost.OL</td>
<td>( O(\frac{1}{\epsilon \gamma^2}) )</td>
<td>( \tilde{O}(\frac{1}{\epsilon^2 \gamma^4}) )</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Chen et al. (2012)</td>
<td>( O(\frac{1}{\epsilon \gamma^2}) )</td>
<td>( \tilde{O}(\frac{1}{\epsilon \gamma^2}) )</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
Structure of Online Boosting

\[
\begin{aligned}
\text{WL}_1 & \xrightarrow{\mathbf{x}_1} \hat{y}_1 \\
\text{WL}_2 & \xrightarrow{\mathbf{x}_1} \hat{y}_2 \\
\vdots & \\
\text{WL}_N & \xrightarrow{\mathbf{x}_1} \hat{y}_N \\
\end{aligned}
\]
Structure of Online Boosting

\[
\hat{y}_1^1 = WL_1^{\text{predict}}(x_1) \\
\hat{y}_1^2 = WL_2^{\text{predict}}(x_1) \\
\vdots \\
\hat{y}_1^N = WL_N^{\text{predict}}(x_1)
\]

\[
WL_1^{\text{update}}(x_1, y_1) \quad \text{w.p. } p_1 \\
WL_2^{\text{update}}(x_1, y_1) \quad \text{w.p. } p_2 \\
\vdots \\
WL_N^{\text{update}}(x_1, y_1) \quad \text{w.p. } p_N
\]
Structure of Online Boosting

${WL}^1$ predict

${WL}^2$ predict

... 

${WL}^N$ predict

$x_1$ $\hat{y}_1$ $y_1$

$x_1$ $\hat{y}_1$

$x_1$ $\hat{y}_1$

$x_1$ $\hat{y}_1$

$x_1$ $\hat{y}_1$
Structure of Online Boosting

\[ WL^1 \]
\[ WL^2 \]
\[ WL^N \]

Predict

update

\[ x_1 \]
\[ \hat{y}_1 \]
\[ y_1 \]

\[ w.p. \ p_1 \]
\[ (x_1, y_1) \]

\[ \cdots \]

\[ \cdots \]
Structure of Online Boosting

\[
\begin{align*}
WL^1 & \quad \text{predict} \quad \hat{y}_2^1 \\
WL^2 & \quad \text{predict} \quad \hat{y}_2^2 \\
\ldots & \quad \text{predict} \quad \hat{y}_2^N \\
WL^N & \quad \text{update} \quad (x_2, y_2) \\
WL^1 & \quad \text{update} \quad (x_2, y_2) \\
WL^2 & \quad \text{update} \quad (x_2, y_2)
\end{align*}
\]
Structure of Online Boosting

\[ x_t, \hat{y}_t, y_t \]

\[ WL^1 \]
\[ predict \]
\[ \hat{y}_t^1 \] \[ w.p. \ p_t^1 (x_t, y_t) \]
\[ WL^1 \]
\[ update \]

\[ WL^2 \]
\[ predict \]
\[ \hat{y}_t^2 \] \[ w.p. \ p_t^2 (x_t, y_t) \]
\[ WL^2 \]
\[ update \]

\[ \ldots \]

\[ WL^N \]
\[ predict \]
\[ \hat{y}_t^N \] \[ w.p. \ p_t^N (x_t, y_t) \]
\[ WL^N \]
\[ update \]
Batch boosting can be analyzed using drifting game.
Batch boosting can be analyzed using drifting game.

**Online version:** sequence of potentials $\Phi_i(s)$ s.t.

- $\Phi_N(s) \geq 1\{s \leq 0\}$,
- $\Phi_{i-1}(s) \geq (\frac{1}{2} - \frac{\gamma}{2})\Phi_i(s - 1) + (\frac{1}{2} + \frac{\gamma}{2})\Phi_i(s + 1)$.
Batch boosting can be analyzed using drifting game.

**Online version:** sequence of potentials $\Phi_i(s)$ s.t.

\[
\begin{align*}
\Phi_N(s) & \geq 1\{s \leq 0\}, \\
\Phi_{i-1}(s) & \geq \left(\frac{1}{2} - \frac{\gamma}{2}\right)\Phi_i(s - 1) + \left(\frac{1}{2} + \frac{\gamma}{2}\right)\Phi_i(s + 1).
\end{align*}
\]

Online boosting algorithm using $\Phi_i$:

- **prediction:** majority vote.
Batch boosting can be analyzed using drifting game.

**Online version:** sequence of potentials $\Phi_i(s)$ s.t.

\[
\begin{align*}
\Phi_N(s) &\geq 1\{s \leq 0\}, \\
\Phi_{i-1}(s) &\geq \left(\frac{1}{2} - \frac{\gamma}{2}\right)\Phi_i(s - 1) + \left(\frac{1}{2} + \frac{\gamma}{2}\right)\Phi_i(s + 1).
\end{align*}
\]

Online boosting algorithm using $\Phi_i$:

- **prediction:** majority vote.
- **update:** $p_t^i = \Pr[(x_t, y_t) \text{ sent to } i\text{th weak learner}] \propto w_t^i$ where $w_t^i = \text{difference in potentials if example is misclassified or not.}$
Mistake Bound

Generalized drifting games analysis implies

\[ \sum_{t=1}^{T} 1\{\mathcal{A}'(x_t) \neq y_t\} \leq \Phi_0(0) T + (S + \frac{1}{\gamma}) \sum_i \|w^i\|_\infty. \]

So we want small \( \|w^i\|_\infty \).

Exponential potential (corresponding to AdaBoost) does not work. Boost-by-Majority (Freund, 1995) potential works well!
Mistake Bound

Generalized drifting games analysis implies

$$\sum_{t=1}^{T} 1\{A'(x_t) \neq y_t\} \leq \phi_0(0) T + (S + \frac{1}{\gamma}) \sum_i \|w^i\|_{\infty}.$$ 

So we want small $\|w^i\|_{\infty}$.

- exponential potential (corresponding to AdaBoost) does not work.
Mistake Bound

Generalized drifting games analysis implies

$$\sum_{t=1}^{T} \mathbf{1}\{A'(x_t) \neq y_t\} \leq \Phi_0(0) T + (S + \frac{1}{\gamma}) \sum_i ||w^i||_\infty.$$ 

So we want small $||w^i||_\infty$.

- exponential potential (corresponding to AdaBoost) does not work.
- Boost-by-Majority (Freund, 1995) potential works well!
Mistake Bound

Generalized drifting games analysis implies

$$\sum_{t=1}^{T} \mathbf{1}\{A'(x_t) \neq y_t\} \leq \Phi_0(0) T + (S + \frac{1}{\gamma}) \sum_i \|w^i\|_\infty.$$ 

So we want small $\|w^i\|_\infty$.

- exponential potential (corresponding to AdaBoost) does not work.
- Boost-by-Majority (Freund, 1995) potential works well!
  - $w_t^i = \text{Pr}[k_t^i \text{ heads in } N - i \text{ flips of a } \frac{\gamma}{2}\text{-biased coin}]$
Generalized drifting games analysis implies

\[
\sum_{t=1}^{T} 1\{A'(x_t) \neq y_t\} \leq \Phi_0(0) T + (S + \frac{1}{\gamma}) \sum_i \|w^i\|_{\infty}.
\]

So we want small \(\|w^i\|_{\infty}\).

- exponential potential (corresponding to AdaBoost) does not work.
- Boost-by-Majority (Freund, 1995) potential works well!
  
  \[w_t^i = \Pr[k^i_t \text{ heads in } N - i \text{ flips of a } \frac{\gamma}{2}-\text{biased coin}] \leq \frac{4}{\sqrt{N-i}}\]
Mistake Bound

Generalized drifting games analysis implies

\[
\sum_{t=1}^{T} 1\{A'(x_t) \neq y_t\} \leq \Phi_0(0) T + \left( S + \frac{1}{\gamma} \right) \sum_i \|w^i\|_\infty.
\]

So we want small \(\|w^i\|_\infty\).

- exponential potential (corresponding to AdaBoost) does not work.
- Boost-by-Majority (Freund, 1995) potential works well!
  
  \[w_t^i = \Pr[k_t^i \text{ heads in } N - i \text{ flips of a } \frac{\gamma}{2}\text{-biased coin}] \leq \frac{4}{\sqrt{N-i}}\]

Online BBM: to get \(\epsilon\) error rate, needs
\[N = O\left( \frac{1}{\gamma^2 \ln(\frac{1}{\epsilon})} \right) \text{ weak learners and } T_\epsilon = O\left( \frac{1}{\epsilon \gamma^2} \right) \text{ examples.} \text{ (Optimal)}\]
Drawback of Online BBM

The draw back of BBM (or Chen et al. (2012)) is the lack of adaptivity.

- requires $\gamma$ as a parameter.
Drawback of Online BBM

The draw back of BBM (or Chen et al. (2012)) is the lack of adaptivity.

- requires $\gamma$ as a parameter.
- treats each weak learner equally: predicts via simple majority vote.
Drawback of Online BBM

The draw back of BBM (or Chen et al. (2012)) is the lack of adaptivity.

- requires $\gamma$ as a parameter.
- treats each weak learner equally: predicts via simple majority vote.

Adaptivity is the key advantage of AdaBoost!

- different weak learners weighted differently based on their performance.
- Adapts to easy data
Batch boosting finds a combination of weak learners to minimize some loss function using coordinate descent. (Breiman, 1999)
Adaptivity via Online Loss Minimization

Batch boosting finds a combination of weak learners to minimize some loss function using coordinate descent. (Breiman, 1999)

- **AdaBoost**: exponential loss
- **AdaBoost.L**: logistic loss
Adaptivity via Online Loss Minimization

Batch boosting finds a combination of weak learners to minimize some loss function using coordinate descent. (Breiman, 1999)

- **AdaBoost**: exponential loss
- **AdaBoost.L**: logistic loss

We generalize this to the online setting:

- replace line search with online gradient descent.
Batch boosting finds a combination of weak learners to minimize some loss function using coordinate descent. (Breiman, 1999)

- AdaBoost: exponential loss
- AdaBoost.L: logistic loss

We generalize this to the online setting:

- replace line search with online gradient descent.
- exponential loss does not work again, use logistic loss to get adaptive online boosting algorithm AdaBoost.OL.
Intuition and main ideas

- Classifier $f$ with real-valued output $f(x)$: predict $\text{sign}(f(x))$
- Logistic loss $\ln(1 + \exp(-f(x)y))$: surrogate for $1\{\text{sign}(f(x)) \neq y\}$
Intuition and main ideas

- Classifier $f$ with real-valued output $f(x)$: predict $\text{sign}(f(x))$
- Logistic loss $\ln(1 + \exp(-f(x)y))$: surrogate for $1\{\text{sign}(f(x)) \neq y\}$
- In batch setting (AdaBoost.L):
  - for each $i$, add output of weak learner with step-size $\alpha$ found by line search to minimize logistic loss

In online setting (AdaBoost.OL):
- for each $i$, search for step-size $\alpha$ using online gradient descent over sequence of $T$ data points
- for each data point, final prediction is weighted majority with weights given by current $\alpha$'s
Intuition and main ideas

- Classifier $f$ with real-valued output $f(x)$: predict $\text{sign}(f(x))$
- Logistic loss $\ln(1 + \exp(-f(x)y))$: surrogate for $1\{\text{sign}(f(x)) \neq y\}$
- In batch setting (*AdaBoost.L*):
  - for each $i$, add output of weak learner with step-size $\alpha$ found by line search to minimize logistic loss
  - final prediction is weighted majority with weights $\alpha$

In online setting (*AdaBoost.OL*):
- for each $i$, search for step-size $\alpha$ using online gradient descent over sequence of $T$ data points
- for each data point, final prediction is weighted majority with weights $\alpha$
Intuition and main ideas

- Classifier \( f \) with real-valued output \( f(x) \): predict \( \text{sign}(f(x)) \)
- Logistic loss \( \ln(1 + \exp(-f(x)y)) \): surrogate for \( 1\{\text{sign}(f(x)) \neq y\} \)
- In batch setting (AdaBoost.L):
  - for each \( i \), add output of weak learner with step-size \( \alpha \) found by line search to minimize logistic loss
  - final prediction is weighted majority with weights \( \alpha \)
- In online setting (AdaBoost.OL):
- for each \( i \), search for step-size \( \alpha \) using online gradient descent over sequence of \( T \) data points
Intuition and main ideas

- Classifier $f$ with real-valued output $f(x)$: predict $\text{sign}(f(x))$
- Logistic loss $\ln(1 + \exp(-f(x)y))$: surrogate for $1\{\text{sign}(f(x)) \neq y\}$
- In batch setting (AdaBoost.L):
  - for each $i$, add output of weak learner with step-size $\alpha$ found by line search to minimize logistic loss
  - final prediction is weighted majority with weights $\alpha$
- In online setting (AdaBoost.OL):
  - for each $i$, search for step-size $\alpha$ using online gradient descent over sequence of $T$ data points
  - for each data point, final prediction is weighted majority with weights given by current $\alpha$'s
Mistake Bound

If WL^i has edge γ_i, then

$$\sum_{t=1}^{T} 1\{A'(x_t) \neq y_t\} \leq \frac{2T}{\sum_i \gamma_i^2} + \tilde{O}\left(\frac{N^2}{\sum_i \gamma_i^2}\right)$$
Mistake Bound

If $WL^i$ has edge $\gamma_i$, then

$$\sum_{t=1}^{T} 1\{A'(x_t) \neq y_t\} \leq \frac{2T}{\sum_i \gamma_i^2} + \tilde{O}\left(\frac{N^2}{\sum_i \gamma_i^2}\right)$$

Suppose $\gamma_i \geq \gamma$, then to get $\epsilon$ error rate AdaBoost.OL needs $N = O\left(\frac{1}{\epsilon \gamma^2}\right)$ weak learners and $T_\epsilon = O\left(\frac{1}{\epsilon^2 \gamma^4}\right)$ examples.
Mistake Bound

If \( \text{WL}^i \) has edge \( \gamma_i \), then

\[
\sum_{t=1}^{T} \mathbf{1}\{A'(x_t) \neq y_t\} \leq \frac{2T}{\sum_i \gamma_i^2} + \tilde{O}\left(\frac{N^2}{\sum_i \gamma_i^2}\right)
\]

Suppose \( \gamma_i \geq \gamma \), then to get \( \epsilon \) error rate AdaBoost.OL needs \( N = O\left(\frac{1}{\epsilon \gamma^2}\right) \) weak learners and \( T_\epsilon = O\left(\frac{1}{\epsilon^2 \gamma^4}\right) \) examples.

Not optimal but adaptive.
Results
Available in **Vowpal Wabbit 8.0**.

- command line option: `--boosting`.
- **VW** as the default “weak” learner (a rather strong one!)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>VW baseline</th>
<th>Online BBM</th>
<th>AdaBoost.OL</th>
<th>Chen et al. 12</th>
</tr>
</thead>
<tbody>
<tr>
<td>20news</td>
<td>0.0812</td>
<td>0.0775</td>
<td>0.0777</td>
<td>0.0791</td>
</tr>
<tr>
<td>a9a</td>
<td>0.1509</td>
<td>0.1495</td>
<td>0.1497</td>
<td>0.1509</td>
</tr>
<tr>
<td>activity</td>
<td>0.0133</td>
<td>0.0114</td>
<td>0.0128</td>
<td>0.0130</td>
</tr>
<tr>
<td>adult</td>
<td>0.1543</td>
<td>0.1526</td>
<td>0.1536</td>
<td>0.1539</td>
</tr>
<tr>
<td>bio</td>
<td>0.0035</td>
<td>0.0031</td>
<td>0.0032</td>
<td>0.0033</td>
</tr>
<tr>
<td>census</td>
<td>0.0471</td>
<td>0.0469</td>
<td>0.0469</td>
<td>0.0469</td>
</tr>
<tr>
<td>covtype</td>
<td>0.2563</td>
<td>0.2347</td>
<td>0.2495</td>
<td>0.2470</td>
</tr>
<tr>
<td>letter</td>
<td>0.2295</td>
<td>0.1923</td>
<td>0.2078</td>
<td>0.2148</td>
</tr>
<tr>
<td>maptaskcoref</td>
<td>0.1091</td>
<td>0.1077</td>
<td>0.1083</td>
<td>0.1093</td>
</tr>
<tr>
<td>nomao</td>
<td>0.0641</td>
<td>0.0627</td>
<td>0.0635</td>
<td>0.0627</td>
</tr>
<tr>
<td>poker</td>
<td>0.4555</td>
<td>0.4312</td>
<td>0.4555</td>
<td>0.4555</td>
</tr>
<tr>
<td>rcv1</td>
<td>0.0487</td>
<td>0.0485</td>
<td>0.0484</td>
<td>0.0488</td>
</tr>
<tr>
<td>vehv2binary</td>
<td>0.0292</td>
<td>0.0286</td>
<td>0.0291</td>
<td>0.0284</td>
</tr>
</tbody>
</table>
Conclusions

We propose

- A natural framework of online boosting.
- An optimal algorithm Online BBM.
- An adaptive algorithm AdaBoost.OL.
Conclusions

We propose

- A natural framework of online boosting.
- An optimal algorithm Online BBM.
- An adaptive algorithm AdaBoost.OL.

Open problem: optimal and adaptive algorithm?