Boosting Algorithms for Maximizing the Soft Margin

Gunnar Rätsch

Friedrich Miescher Laboratory of the Max Planck Society, Tübingen, Germany

February 11, 2008

4th Workshop on Ensemble Methods
Protocol of Boosting

- Maintain a distribution $d^t$ on the examples
- At iteration $t = 1, \ldots, T$:
  - Receive a "weak" hypothesis $h_t$
  - Update $d^t$ to $d^{t+1}$, put more weights on "hard" examples
- Output a convex combination of the weak hypotheses

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

[Freund & Schapire, 1995]
Protocol of Boosting

- Maintain a distribution $d^t$ on the examples
- At iteration $t = 1, \ldots, T$:
  1. Receive a “weak” hypothesis $h_t$
  2. Update $d^t$ to $d^{t+1}$, put more weights on “hard” examples
- Output a convex combination of the weak hypotheses

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

[Freund & Schapire, 1995]
Introduction to Boosting

Protocol of Boosting

- Maintain a distribution $d^t$ on the examples
- At iteration $t = 1, \ldots, T$:
  1. Receive a “weak” hypothesis $h_t$
  2. Update $d^t$ to $d^{t+1}$, put more weights on “hard” examples
- Output a convex combination of the weak hypotheses

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

[Freund & Schapire, 1995]
First hypothesis:

- **Error rate:** \( \frac{2}{11} \)

\[
\epsilon_t = \frac{1}{N} \sum_{n=1}^{N} d_t^n \mathbb{1}(h_t(x_n) = y_n)
\]

- **Edge:** \( \frac{9}{22} \)

\[
\gamma_t = \frac{1}{N} \sum_{n=1}^{N} d_t^n y_n h_t(x_n)
= 1 - 2 \epsilon_t
\]
Boosting: 1st Iteration

First hypothesis:

- Error rate: $\frac{2}{11}$

\[
\epsilon_t = \sum_{n=1}^{N} d^t_n \mathbb{1}(h_t(x_n) = y_n)
\]

- Edge: $\frac{9}{22}$

\[
\gamma_t = \sum_{n=1}^{N} d^t_n y_n h_t(x_n) = 1 - 2\epsilon_t
\]
Update Distribution

Misclassified examples \Rightarrow Increased weights

After update:
- Error rate:
  \( \epsilon(h_t, d^{t+1}) = \frac{1}{2} \)
- Edge:
  \( \gamma(h_t, d^{t+1}) = 0 \)
Introduction to Boosting

Update Distribution

Misclassified examples $\Rightarrow$ Increased weights

After update:
- Error rate:
  \[ \epsilon(h_t, d^{t+1}) = \frac{1}{2} \]
- Edge:
  \[ \gamma(h_t, d^{t+1}) = 0 \]
Before 2nd Iteration
Boosting: 2nd Hypothesis

Edge $\gamma > \delta$
Update Distribution

Edge $\gamma = 0$

AdaBoost update sets edge of last hypothesis to 0
Introduction to Boosting

Boosting: 3rd Hypothesis

- Boosting Algorithms for Large Soft Margins
- February 11, 2008
Boosting: 4th Hypothesis
All Hypotheses

- leicht
- schwer
- nicht rot
- sehr rot
Decision: \( f_\alpha(x) = \sum_{t=1}^{T} \alpha_t h_t(x) > 0 \)
Introduction to Boosting

Large Margin and Linear Separation

Input space $\mathcal{X}$

Feature space $\mathcal{F}$

Linear separation in $\mathcal{F}$ is nonlinear separation in $\mathcal{X}$

$\Phi(\mathbf{x}) = \begin{pmatrix} h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \\ \vdots \end{pmatrix}$

$\mathcal{H} = \{ h_1, h_2, \ldots \}$

[O. Mangasarian, 1999; G.R., Mika, Schölkopf & Müller, 2002]
Margin of the combined hypothesis $f_\alpha$ for example $(x_n, y_n)$

$$\rho_n(\alpha) = y(\Phi(x) \cdot \alpha)$$

$$= y_n f_\alpha(x_n)$$

$$= y_n \sum_{t=1}^{T} \alpha_t h_t(x_n) \quad (\alpha \in \mathcal{P}^T)$$

Margin of set of examples is minimum over examples

$$\rho(\alpha) := \min_n \rho_n(\alpha)$$

[Freund, Schapire, Bartlett & Lee, 1998]
**Edge vs. Margin**

**Edge**
- Measurement of “goodness” of a hypothesis w.r.t. a distribution
- Edge of a hypothesis \( h \) for a distribution \( d \) on the examples

\[
\gamma_h(d) = \sum_{n=1}^{N} d_n y_n h(x_n) \quad d \in \mathcal{P}^N
\]

**Margin**
- Measure for “confidence” in prediction for a hypothesis weighting
- Margin of example \( n \) for current hypothesis weighting \( \alpha \)

\[
\rho_n(\alpha) = y_n f_{\alpha}(x_n) = y_n \sum_{t=1}^{T} \alpha_t h_t(x_n) \quad (\alpha \in \mathcal{P}^T)
\]

What is the connection? [Breiman, 1999]
Edge vs. Margin

**Edge**
- Measurement of “goodness” of a hypothesis w.r.t. a distribution
- Edge of a hypothesis $h$ for a distribution $d$ on the examples
  \[
  \gamma_h(d) = \sum_{n=1}^{N} d_n y_n h(x_n) \quad d \in \mathcal{P}^N
  \]

**Margin**
- Measure for “confidence” in prediction for a hypothesis weighting
- Margin of example $n$ for current hypothesis weighting $\alpha$
  \[
  \rho_n(\alpha) = y_n f_\alpha(x_n) = y_n \sum_{t=1}^{T} \alpha_t h_t(x_n) \quad (\alpha \in \mathcal{P}^T)
  \]

What is the connection?  

[Breiman, 1999]
Edge vs. Margin

**Edge**
- Measurement of “goodness” of a hypothesis w.r.t. a distribution
- **Edge of a hypothesis** $h$ for a distribution $d$ on the examples
  \[
  \gamma_h(d) = \sum_{n=1}^{N} d_n y_n h(x_n) \quad d \in \mathcal{P}^N
  \]

**Margin**
- Measure for “confidence” in prediction for a hypothesis weighting
- **Margin of example** $n$ for current hypothesis weighting $\alpha$
  \[
  \rho_n(\alpha) = y_n f_\alpha(x_n) = y_n \sum_{t=1}^{T} \alpha_t h_t(x_n) \quad (\alpha \in \mathcal{P}^T)
  \]

What is the connection? [Breiman, 1999]
von Neumann's Minimax-Theorem

Set of examples \( S = \{(x_1, y_1), \ldots, (x_N, y_N)\} \)
and hypotheses set \( \mathcal{H}^t = \{h_1, \ldots, h_t\} \),

minimum edge: \( \gamma_t^* = \min \max_{d \in \mathcal{P}^N} \max_{h \in \mathcal{H}^t} \gamma_h(d) \)

maximum margin: \( \rho_t^* = \max \min_{\alpha \in \mathcal{P}^t} y_n f_\alpha(x_n) \)

Duality: \( \gamma_t^* = \rho_t^* \)

[von Neumann, 1928]
Duality gap

For any non-optimal \( d \in P^N \) and \( \alpha \in P^t \),

\[
\gamma(d) \geq \gamma_t^* = \rho_t^* \geq \rho(\alpha)
\]
For any non-optimal $d \in \mathcal{P}^N$ and $\alpha \in \mathcal{P}^t$,

$$\gamma(d) \geq \gamma_t^* = \rho_t^* \geq \rho(\alpha)$$
How Large is the Maximal Margin?

Assumptions on Weak learner

For any distribution \( d \) on the examples, the weak learner returns a hypothesis \( h \) with edge \( \gamma_h(d) \) at least \( g \).
Best case: \( g = \rho^* = \gamma^* \).

[Breiman, 1999; Bennett et al.; G.R. et al., 2001; Rudin et al., 2004]
How Large is the Maximal Margin?

Assumptions on Weak learner

For any distribution $d$ on the examples, the weak learner returns a hypothesis $h$ with edge $\gamma_h(d)$ at least $g$.

Best case: $g = \rho^* = \gamma^*$.

Implication from Minimax Theorem

There exists $\alpha \in \mathcal{P}^N$, such that $\rho(\alpha) \geq g$.

[Breiman, 1999; Bennett et al.; G.R. et al., 2001; Rudin et al., 2004]
How Large is the Maximal Margin?

Assumptions on Weak learner

For any distribution $d$ on the examples, the weak learner returns a hypothesis $h$ with edge $\gamma_h(d)$ at least $g$. Best case: $g = \rho^* = \gamma^*$.

Implication from Minimax Theorem

There exists $\alpha \in \mathcal{P}^N$, such that $\rho(\alpha) \geq g$.

Idea to iteratively solve LP: LPBoost

Add “best” hypothesis $h = \arg\max \gamma_h(d^t)$ to $\mathcal{H}^{t+1}$ and resolve

$$d^{t+1} = \arg\min_{d \in \mathcal{P}^N} \max_{h \in \mathcal{H}^{t+1}} \gamma_h(d).$$

[Breiman, 1999; Bennett et al.; G.R. et al., 2001; Rudin et al., 2004]
How Large is the Maximal Margin?

Assumptions on Weak learner

For any distribution \( d \) on the examples, the weak learner returns a hypothesis \( h \) with edge \( \gamma_h(d) \) at least \( g \).

Best case: \( g = \rho^* = \gamma^* \).

What about AdaBoost?

- May “oscillate”
- Does not find maximizing \( \alpha \) (counter examples)
- But there some guarantees:
  - \( \rho(\alpha^t) \geq 0 \) after \( 2 \ln N/g^2 \) iterations
  - \( \rho(\alpha^t) \geq g/2 \) in the limit

[Breiman, 1999; Bennett et al.; G.R. et al., 2001; Rudin et al., 2004]
How to **Maximize** the Margin?

Modify AdaBoost for maximizing margin

- **Arc-GV** asymptotically maximizes the margin
  - quite slow, no converge rates
- **LPBoost** uses a Linear Programming solver
  - Often very fast in practice, but no converge rates
- **AdaBoost** requires \( \frac{2 \log(N)}{\delta^2} \) iterations to get \( \rho^t \in [\rho^* - \delta, \rho^*] \)
  - Slow in practice: just as fast as theory predicts
- **TotalBoost** requires \( \frac{2 \log(N)}{\delta^2} \) iterations to get \( \rho^t \in [\rho^* - \delta, \rho^*] \)
  - Fast in practice
  - Combination of benefits

[Breiman, 1999, G.R. & Warmuth, 2004; Warmuth et al., 2006]
How to **Maximize** the Margin?

Modify AdaBoost for maximizing margin

- **Arc-GV** asymptotically maximizes the margin
  - quite slow, no converge rates
- **LPBoost** uses a Linear Programming solver
  - Often very fast in practice, but **no converge rates**
- **AdaBoost** requires $\frac{2\log(N)}{\delta^2}$ iterations to get $\rho^t \in [\rho^* - \delta, \rho^*]$  
  - Slow in practice: just as fast as theory predicts
- **TotalBoost** requires $\frac{2\log(N)}{\delta^2}$ iterations to get $\rho^t \in [\rho^* - \delta, \rho^*]$  
  - Fast in practice
  - Combination of benefits

[Breiman, 1999, G.R. & Warmuth, 2004; Warmuth et al., 2006]
How to Maximize the Margin?

Modify AdaBoost for maximizing margin

- **Arc-GV** asymptotically maximizes the margin
  - quite slow, no converge rates
- **LPBoost** uses a Linear Programming solver
  - Often very fast in practice, but no converge rates
- **AdaBoost** requires $\frac{2\log(N)}{\delta^2}$ iterations to get $\rho_t \in [\rho^* - \delta, \rho^*]$ but slow in practice: just as fast as theory predicts
- **TotalBoost** requires $\frac{2\log(N)}{\delta^2}$ iterations to get $\rho_t \in [\rho^* - \delta, \rho^*]$ but fast in practice
  - Combination of benefits

[Breiman, 1999, G.R. & Warmuth, 2004; Warmuth et al., 2006]
How to **Maximize** the Margin?

Modify AdaBoost for maximizing margin

- **Arc-GV** asymptotically maximizes the margin
  - quite slow, no converge rates
- **LPBoost** uses a Linear Programming solver
  - Often very fast in practice, but no converge rates
- **AdaBoost\*** requires \( \frac{2 \log(N)}{\delta^2} \) iterations to get \( \rho^t \in [\rho^* - \delta, \rho^*] \)
  - Slow in practice: just as fast as theory predicts
- **TotalBoost** requires \( \frac{2 \log(N)}{\delta^2} \) iterations to get \( \rho^t \in [\rho^* - \delta, \rho^*] \)
  - Fast in practice
  - Combination of benefits

[Breiman, 1999, G.R. & Warmuth, 2004; Warmuth et al., 2006]
Idea: Projections to $\hat{\gamma}_t$ instead of 0

Separation

$$\min_{d \in \mathcal{P}^N} \Delta(d, d^t)$$

s.t.

$$\sum_{n=1}^{N} d_n y_n h_t(x_n) \leq 0 \quad \text{for } r = 1, \ldots, t$$

Large Margin Separation

$$\min_{d \in \mathcal{P}^N} \Delta(d, d^t)$$

s.t.

$$\sum_{n=1}^{N} d_n y_n h_t(x_n) \leq \hat{\gamma}_t \quad \text{for } r = 1, \ldots, t$$

[G.R. & Warmuth, 2004]
Want margin $\geq g - \delta$

Assumption: $\gamma_t \geq g$

Estimate of target: $\hat{\gamma}_t = (\min_{q=1,...,t} \gamma_q) - \delta$
Want margin $\geq g - \delta$

Assumption: $\gamma_t \geq g$

Estimate of target: $\hat{\gamma}_t = (\min_{q=1,\ldots,t} \gamma_q) - \delta$
Corrective vs. Totally Corrective Updates

**Corrective:** Single constraint

\[
\min_{d \in \mathcal{P}^N} \Delta(d, d^t) \quad \text{(AdaBoost*)}
\]

\[
\text{s.t.} \quad \sum_{n=1}^{N} d_n y_n h_t(x_n) \leq \hat{\gamma}_t \quad \text{for } r = 1, \ldots, t
\]

**Totally corrective:** One constraint per past weak hypothesis

\[
\min_{d \in \mathcal{P}^N} \Delta(d, d^1) \quad \text{(TotalBoost)}
\]

\[
\text{s.t.} \quad \sum_{n=1}^{N} d_n y_n h_r(x_n) \leq \hat{\gamma}_t \quad \text{for } r = 1, \ldots, t
\]

[Kivinen & Warmuth, 1999; Warmuth et al., 2006]
TotalBoost_\delta

1. **Input:** $S = \langle (x_1, y_1), \ldots, (x_N, y_N) \rangle$, desired accuracy $\delta$
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots$
   1. Train classifier on $\{S, d^t\}$ and obtain hypothesis $h_t: x \mapsto [-1, 1]$ and let $u_i^t = y_i h_t(x_i)$
   2. Calculate the edge $\gamma_t$ of $h_t$: $\gamma_t = d^t \cdot u^t$
   3. Set $\hat{\gamma}_t = (\min_{q=1,\ldots,t} \gamma_q) - \delta$ and solve
      $d^{t+1} = \text{argmin}_{\{d \in \mathcal{P}^N \mid d \cdot u^q \leq \hat{\gamma}_t, \text{ for } 1 \leq q \leq t\}} \Delta(d, d^1)$
      \{C_t\}
   4. **If** above infeasible or $d^{t+1}$ contains a zero
      **then** $T = t$ and break

4. **Output:** $f_\alpha(x) = \sum_{t=1}^T \alpha_t h_t(x)$, where the coefficients $\alpha_t$ maximize margin over hypotheses set $\{h_1, \ldots, h_T\}$. 
**TotalBoost\(\delta\)**

1. **Input:** \(S = \langle (x_1, y_1), \ldots, (x_N, y_N) \rangle\), desired accuracy \(\delta\)
2. **Initialize:** \(d^1_n = 1/N\) for all \(n = 1 \ldots N\)
3. **Do for** \(t = 1, \ldots\)
   1. Train classifier on \(\{S, d^t\}\) and obtain hypothesis \(h_t: x \mapsto [-1, 1]\) and let \(u^t_i = y_i h_t(x_i)\)
   2. Calculate the edge \(\gamma_t\) of \(h_t\): \(\gamma_t = d^t \cdot u^t\)
   3. Set \(\hat{\gamma}_t = (\min_{q=1, \ldots, t} \gamma_q) - \delta\) and solve
      \[
      d^{t+1} = \underset{d \in \mathcal{P}^N}{\text{argmin}} \Delta(d, d^1) \\
      \{d \in \mathcal{P}^N \mid d \cdot u^q \leq \hat{\gamma}_t, \text{ for } 1 \leq q \leq t\} = \mathcal{C}_t
      \]
   4. If above infeasible or \(d^{t+1}\) contains a zero
      then \(T = t\) and break
4. **Output:** \(f_\alpha(x) = \sum_{t=1}^T \alpha_t h_t(x)\), where the coefficients \(\alpha_t\)
   maximize margin over hypotheses set \(\{h_1, \ldots, h_T\}\).
TotalBoost_\delta

1. **Input:** \( S = \langle (x_1, y_1), \ldots, (x_N, y_N) \rangle \), desired accuracy \( \delta \)

2. **Initialize:** \( d^1_n = 1/N \) for all \( n = 1 \ldots N \)

3. **Do for** \( t = 1, \ldots \)
   1. Train classifier on \( \{S, d^t\} \) and obtain hypothesis \( h_t : x \mapsto [-1, 1] \) and let \( u^t_i = y_i h_t(x_i) \)
   2. Calculate the edge \( \gamma_t \) of \( h_t \): \( \gamma_t = d^t \cdot u^t \)
   3. Set \( \hat{\gamma}_t = (\min_{q=1,\ldots,t} \gamma_q) - \delta \) and solve

   **Optimization Problem**

   \[
   d^{t+1} = \arg\min_{d \in C_t} \Delta(d, d^1)
   \]

   with \( C_t := \{d \in \mathcal{P}^N | d \cdot u^q \leq \hat{\gamma}_t, \text{ for } 1 \leq q \leq t\} \)

4. **Output:** \( f_\alpha(x) = \sum_{t=1}^T \alpha_t h_t(x) \), where the coefficients \( \alpha_t \) maximize margin over hypotheses set \( \{h_1, \ldots, h_T\} \).
Iteration Bound \(\left\lceil \frac{2\ln N}{\delta^2} \right\rceil\)

**Theorem**

Assume the base learner returns hypotheses with edge greater than \(g\), then AdaBoost* and TotalBoost terminate after at most \(\frac{2\ln N}{\delta^2}\) iteration with margin at least \(g - \delta\).

**Lemma**

For \(d^t, d^{t+1} \in \mathcal{P}^N\) and \(u \in [-1, 1]^N\), if \(\Delta(d^{t+1}, d^t)\) finite and \(d^{t+1} \cdot u \neq d^t \cdot u\) then

\[
\Delta(d^{t+1}, d^t) > \frac{(d^{t+1} \cdot u - d^t \cdot u)^2}{2}
\]

[Warmuth, Liao & G.R., 2006]
Theorem

Assume the base learner returns hypotheses with edge greater than $g$, then AdaBoost* and TotalBoost terminate after at most $\frac{2\ln N}{\delta^2}$ iteration with margin at least $g - \delta$.

Lemma

For $d^t, d^{t+1} \in P^N$ and $u \in [-1, 1]^N$, if $\Delta(d^{t+1}, d^t)$ finite and $d^{t+1} \cdot u \neq d^t \cdot u$ then

$$\Delta(d^{t+1}, d^t) > \frac{(d^{t+1} \cdot u - d^t \cdot u)^2}{2}$$

[Warmuth, Liao & G.R., 2006]
Generalized Pythagorean Theorem

\[ C_t = \{ d \in \mathcal{P}^N \mid d \cdot u^q \leq \hat{\gamma}_t, \ 1 \leq q \leq t \}, \ C_0 = \mathcal{P}^N, \ C_t \subseteq C_{t-1} \]

\( d^t \) is projection of \( d^1 \) onto \( C_{t-1} \) at iteration \( t - 1 \)

\[ d^t = \arg\min_{d \in C_{t-1}} \Delta(d, d^1) \]

\[ \Delta(d^{t+1}, d^1) \geq \Delta(d^t, d^1) + \Delta(d^{t+1}, d^t) \]

[Herbster & Warmuth, 2001]
Convergence

Sketch of Proof

1: $\Delta(d^2, d^1) - \Delta(d^1, d^1) \geq \Delta(d^2, d^1) > \frac{\delta^2}{2}$

2: $\Delta(d^3, d^2) - \Delta(d^2, d^1) \geq \Delta(d^3, d^2) > \frac{\delta^2}{2}$

\[\vdots\]

\[t: \Delta(d^{t+1}, d^1) - \Delta(d^t, d^1) \geq \Delta(d^{t+1}, d^t) > \frac{\delta^2}{2}\]

\[\vdots\]

\[T - 1: \Delta(d^T, d^1) - \Delta(d^{T-1}, d^1) \geq \Delta(d^T, d^{T-1}) > \frac{\delta^2}{2}\]

Therefore, $T \leq \lceil \frac{2\ln N}{\delta^2} \rceil$

[Warmuth, Liao & G.R., 2006]
Illustrative Experiments

Cox-1 dataset from Telik Inc.
- Relatively small drug-design data set
  - 125 binary labeled examples
  - 3888 binary features
- Compare convergence of margin versus number of iterations
Illustrative Experiments

Cox-1 ($\delta = 0.01$)

- $\text{AdaBoost}_\delta^*$
- LPBoost
- TotalBoost$_\delta$

Results

- Corrective algorithms very slow
- LPBoost & TotalBoost need few iterations
- Initial speed crucially depends on $\delta$
Illustrative Experiments

Cox-1 ($\delta = 0.01$)

- AdaBoost$^*$
- LPBoost
- TotalBoost$^\delta$

Results
- Corrective algorithms very slow
- LPBoost & TotalBoost need few iterations
- Initial speed crucially depends on $\delta$

Gunnar Rätsch (FML, Tübingen)  Boosting Algorithms for Large Soft Margins  February 11, 2008  27 / 38
Illustrative Experiments

Cox-1 ($\delta = 0.01$)

- AdaBoost$_\delta^*$
- LPBoost
- TotalBoost$_\delta$

Results

- Corrective algorithms very slow
- LPBoost & TotalBoost need few iterations
- Initial speed crucially depends on $\delta$
Illustrative Experiments

Cox-1 ($\delta = 0.01$)
- AdaBoost$_\delta$
- LPBoost
- TotalBoost$_\delta$

Results
- Corrective algorithms very slow
- LPBoost & TotalBoost$_\delta$ need few iterations
- Initial speed crucially depends on $\delta$
Illustrative Experiments

Cox-1 ($\delta = 0.01$)
- AdaBoost$_{\delta}^*$
- LPBoost
- TotalBoost$_{\delta}$

Results
- Corrective algorithms very slow
- LPBoost & TotalBoost$_{\delta}$ need few iterations
- Initial speed crucially depends on $\delta$
Illustrative Experiments

Cox-1 ($\delta = 0.01$)
- AdaBoost$_{\delta}$
- LPBoost
- TotalBoost$_{\delta}$

Results
- Corrective algorithms very slow
- LPBoost & TotalBoost$_{\delta}$ need few iterations
- Initial speed crucially depends on $\delta$
LPBoost May Perform Much Worse Than TotalBoost

- Identified cases where LPBoost converges considerably slower than TotalBoost$_\delta$
- Dataset is a series of artificial datasets of 1000 examples with varying number of features created as follows:
  - First generated $N_1$ random $\pm 1$-valued features $x_1, \ldots, x_{N_1}$ and set the label of the examples as $y = \text{sign}(x_1 + x_2 + x_3 + x_4 + x_5)$
  - Then duplicated each features $N_2$ times, perturbed the features by Gaussian noise with $\sigma = 0.1$, and clipped the feature values so that they lie in the interval $[-1,1]$
  - Considered different $N_1, N_2$, the total number of features is $N_2 \times N_1$
LPBoost performs worse for high dimensional data with many redundant features.

LPBoost vs. TotalBoost_δ on two 100,000 dimensional datasets: [left] many redundant features (N_1 = 1,000, N_2 = 100) and [right] independent features (N_1 = 100,000, N_2 = 1). Show margin vs. number of iterations.
Do these algorithms work better in practice?

Do they generalize better?

- **Usually not!** Algorithms just overfit quicker.
- Only slight improvements in the noise free case.

Soft Margins for AdaBoost

1. Limit the *influence* of examples ("AdaBoost$_{reg}$")
   - Heuristic algorithm, no convergence result . . .
   - . . . but works very well in practice
2. Soft margins à la SVMs ("$\nu$-Arc", "SoftBoost")
   - Change optimization problem to include slacks
   - Convergence proofs did not work $\Rightarrow$ only asymptotic results

Do These Algorithms Work Better in Practice?

Do they generalize better?

- **Usually not!** Algorithms just overfit quicker.
- Only slight improvements in the noise free case.

Soft Margins for AdaBoost

1. Limit the *influence* of examples ("AdaBoost\,_{\text{reg}}")
   - Heuristic algorithm, no convergence result . . .
   - . . . but works very well in practice

2. Soft margins à la SVMs ("$\nu$-Arc", "SoftBoost")
   - Change optimization problem to include slacks
   - Convergence proofs did not work $\Rightarrow$ only asymptotic results
   - New: Can prove fast convergence with soft margin (NIPS'07)

Do These Algorithms Work Better in Practice?

**Do they generalize better?**

- **Usually not!** Algorithms just overfit quicker.
- Only slight improvements in the noise free case.

**Soft Margins for AdaBoost**

1. Limit the *influence* of examples ("AdaBoost$_{\text{reg}}$")
   - Heuristic algorithm, no convergence result . . .
   - . . . but works very well in practice

2. Soft margins à la SVMs ("$\nu$-Arc", "SoftBoost")
   - Change optimization problem to include slacks
   - Convergence proofs did not work $\Rightarrow$ only asymptotic results
   - **New:** Can prove fast convergence with soft margin (NIPS’07)

Do These Algorithms Work Better in Practice?

Do they generalize better?

- **Usually not!** Algorithms just overfit quicker.
- Only slight improvements in the noise free case.

**Soft Margins for AdaBoost**

1. **Limit the influence** of examples ("AdaBoost\textsubscript{reg}")
   - Heuristic algorithm, no convergence result . . .
   - . . . but works very well in practice

2. **Soft margins à la SVMs** ("\(\nu\)-Arc", "SoftBoost")
   - Change optimization problem to include slacks
   - Convergence proofs did not work \(\Rightarrow\) only asymptotic results
   - **New:** Can prove fast convergence with soft margin (NIPS’07)

SoftBoost = TotalBoost with $\nu$-Trick

Hard-margin separation

\[
\max_{\rho, \xi, \alpha} \rho - \frac{1}{\nu} \sum_{n=1}^{N} \xi_n \\
\text{s.t. } \rho \geq 0, \xi \in \mathbb{R}_+, \alpha \in \mathbb{R}^{|\mathcal{H}|} \\
\text{with } y_n \sum_{j=1}^{\mathcal{H}} \alpha_j h_j(x_n) \geq \rho - \xi_n \\
\text{for } n = 1, \ldots, N.
\]
**SoftBoost = TotalBoost with \( \nu \)-Trick**

\[
\begin{align*}
\max_{\rho, \xi, \alpha} \quad & \rho - \frac{1}{\nu} \sum_{n=1}^{N} \xi_n \\
\text{s.t.} \quad & \rho \geq 0, \xi \in \mathbb{R}_+, \alpha \in \mathbb{R}^{|\mathcal{H}|} \\
\text{with} \quad & y_n \sum_{j=1}^{\mathcal{H}} \alpha_j h_j(x_n) \geq \rho - \xi_n \\
& \text{for } n = 1, \ldots, N.
\end{align*}
\]

**Hard-margin separation with outlier**
SoftBoost = TotalBoost with $\nu$-Trick

Soft-margin separation with outlier

Chose margin such that $\nu$ examples are within the margin area

$$\max_{\rho, \xi, \alpha} \quad \rho - \frac{1}{\nu} \sum_{n=1}^{N} \xi_n$$

s.t. $\rho \geq 0, \xi \in \mathbb{R}_+, \alpha \in \mathbb{R}^{\mathcal{H}}$

with $y_n \sum_{j=1}^{\mathcal{H}} \alpha_j h_j(x_n) \geq \rho - \xi_n$ for $n = 1, \ldots, N.$
**SoftBoost = TotalBoost with \( \nu \)-Trick**

**Soft-margin separation with outlier**

Chose margin such that \( \nu \) examples are within the margin area

\[
\begin{align*}
\max_{\rho, \xi, \alpha} & \quad \rho - \frac{1}{\nu} \sum_{n=1}^{N} \xi_n \\
\text{s.t.} & \quad \rho \geq 0, \xi \in \mathbb{R}_+, \alpha \in \mathbb{R}^{\vert\mathcal{H}\vert} \\
\text{with} & \quad y_n \sum_{j=1}^{\vert\mathcal{H}\vert} \alpha_j h_j(x_n) \geq \rho - \xi_n \\
& \quad \text{for } n = 1, \ldots, N.
\end{align*}
\]
Softboost$_{\delta,\nu}$

1. **Input:** $S = \langle (x_1, y_1), \ldots , (x_N, y_N) \rangle$, desired accuracy $\delta$
2. **Initialize:** $d^1_n = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots$
   1. Train classifier on $\{S, d^t\}$ and obtain hypothesis $h_t : x \mapsto [-1, 1]$ and let $u^t_i = y_i h_t(x_i)$
   2. Calculate the edge $\gamma_t$ of $h_t$: $\gamma_t = d^t \cdot u^t$
   3. Set $\hat{\gamma}_t = (\min_{q=1,\ldots,t} \gamma_q) - \delta$ and solve
      $$ d^{t+1} = \arg \min_{\{d \in P^N | d \leq 1/\nu; d \cdot u^q \leq \hat{\gamma}_t, \text{for } 1 \leq q \leq t\}} \Delta(d, d^1) $$
      with $C_t$
   4. **If** above infeasible or $d^{t+1}$ contains a zero
      **then** $T = t$ and break

4. **Output:** $f_\alpha(x) = \sum_{t=1}^T \alpha_t h_t(x)$, where the coefficients $\alpha_t$
maximize margin over hypotheses set $\{h_1, \ldots , h_T\}$.
Softboost$_{\delta,\nu}$

1. **Input:** $S = \langle (x_1, y_1), \ldots, (x_N, y_N) \rangle$, desired accuracy $\delta$
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots$
   1. Train classifier on $\{S, d^t\}$ and obtain hypothesis $h_t: x \mapsto [-1, 1]$ and let $u^t_i = y_i h_t(x_i)$
   2. Calculate the edge $\gamma_t$ of $h_t$: $\gamma_t = d^t \cdot u^t$
   3. Set $\hat{\gamma}_t = \left( \min_{q=1, \ldots, t} \gamma_q \right) - \delta$ and solve
      $$d^{t+1} = \text{argmin}_{\{d \in \mathcal{P}^N \mid d \leq 1/\nu; d \cdot u^q \leq \hat{\gamma}_t, \text{ for } 1 \leq q \leq t\}} \Delta(d, d^1)$$
      with $C_t$
   4. If above infeasible or $d^{t+1}$ contains a zero then $T = t$ and break
4. **Output:** $f_\alpha(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$, where the coefficients $\alpha_t$ maximize margin over hypotheses set $\{h_1, \ldots, h_T\}$.
Softboost $\delta, \nu$

1. **Input:** $S = \langle (x_1, y_1), \ldots, (x_N, y_N) \rangle$, desired accuracy $\delta$
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots$
   1. Train classifier on $\{S, d^t\}$ and obtain hypothesis $h_t : x \mapsto [-1, 1]$ and let $u_i^t = y_i h_t(x_i)$
   2. Calculate the edge $\gamma_t$ of $h_t$: $\gamma_t = d^t \cdot u^t$
   3. Set $\hat{\gamma}_t = (\min_{q=1, \ldots, t} \gamma_q) - \delta$ and solve

**Optimization Problem**

$$d^{t+1} = \arg\min_{d \in C_t} \Delta(d, d^1)$$

with $C_t := \{d \in \mathcal{P}^N \mid d \leq 1/\nu, \ d \cdot u^q \leq \hat{\gamma}_t, \ \text{for} \ 1 \leq q \leq t\}$

4. **Output:** $f_{\alpha}(x) = \sum_{t=1}^T \alpha_t h_t(x)$, where the coefficients $\alpha_t$ maximize margin over hypotheses set $\{h_1, \ldots, h_T\}$. 
Iteration Bound

Theorem

Assume the base learner returns hypotheses with edge greater than $g$, then SoftBoost terminates after at most \[ \frac{2 \ln(N/\nu)}{\delta^2} \] iteration with margin at least $g - \delta$.

[Warmuth, Glocer & G.R., 2007]
Generalization performance of SoftBoost (solid) and LPBoost (dotted) on a synthetic data set with 10% label-noise for different $\delta$. 

Gunnar Rätsch (FML, Tübingen)
Convergence Speed of Different Algorithms

Soft margin objective vs. the number of iterations for LPBoost, SoftBoost, BrownBoost and SmoothBoost.
Generalization Errors on IDA Benchmarks

- RBF networks as base learner
- 5-fold cross-validation on 100 splits
- Average error ± standard deviation
- LPBoost very similar to SoftBoost (not shown)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>AdaBoost</th>
<th>BrownBoost</th>
<th>AdaBoost reg</th>
<th>SoftBoost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banana</td>
<td>13.3 ± 0.7</td>
<td>12.9 ± 0.7</td>
<td>11.3 ± 0.6</td>
<td><strong>11.1 ± 0.5</strong></td>
</tr>
<tr>
<td>B.Cancer</td>
<td>32.1 ± 3.8</td>
<td>30.2 ± 3.9</td>
<td><strong>27.3 ± 4.3</strong></td>
<td>28.0 ± 4.5</td>
</tr>
<tr>
<td>Diabetes</td>
<td>27.9 ± 1.5</td>
<td>27.2 ± 1.6</td>
<td>24.5 ± 1.7</td>
<td><strong>24.4 ± 1.7</strong></td>
</tr>
<tr>
<td>German</td>
<td>26.9 ± 1.9</td>
<td>24.8 ± 1.9</td>
<td>25.0 ± 2.2</td>
<td><strong>24.7 ± 2.1</strong></td>
</tr>
<tr>
<td>Heart</td>
<td>20.1 ± 2.7</td>
<td>20.0 ± 2.8</td>
<td><strong>17.6 ± 3.0</strong></td>
<td>18.2 ± 2.7</td>
</tr>
<tr>
<td>Ringnorm</td>
<td>1.9 ± 0.3*</td>
<td>1.9 ± 0.2</td>
<td><strong>1.7 ± 0.2</strong></td>
<td>1.8 ± 0.2</td>
</tr>
<tr>
<td>F.Solar</td>
<td>36.1 ± 1.5</td>
<td>36.1 ± 1.4</td>
<td><strong>34.4 ± 1.7</strong></td>
<td>35.5 ± 1.4</td>
</tr>
<tr>
<td>Thyroid</td>
<td><strong>4.4 ± 1.9</strong></td>
<td>4.6 ± 2.1</td>
<td>4.9 ± 2.0</td>
<td>4.9 ± 1.9</td>
</tr>
<tr>
<td>Titanic</td>
<td>22.8 ± 1.0</td>
<td>22.8 ± 0.8</td>
<td><strong>22.7 ± 1.0</strong></td>
<td>23.0 ± 0.8</td>
</tr>
<tr>
<td>Waveform</td>
<td>10.5 ± 0.4</td>
<td>10.4 ± 0.4</td>
<td>10.4 ± 0.7</td>
<td><strong>9.8 ± 0.5</strong></td>
</tr>
</tbody>
</table>
Summarizing Boosting

- AdaBoost can be viewed as entropy projection
- TotalBoost projects based on all previous hypotheses
- Provably maximizes the margin
  - Theory: as fast as AdaBoost$^*$
  - Practice: much faster ($\approx$ LPBoost)
- Experiments corroborate our theory
  - Few iterations (good for feature selection)
  - LPBoost may have problems of maximizing the margin
- Soft margin extension with provable convergence
- First time to combine
  - Theoretical convergence guarantee
  - Empirical fast convergence
  - Good generalization performance on noisy data
Slides with references will be available at http://www.fml.mpg.de/raetsch/lectures/

Collaborators for this work:
Kristin Bennett, Ayhan Demiriz, Karen Glocer, Jun Liao, Sebastian Mika, Klaus-Robert Müller, Takashi Onoda, Bernhard Schölkopf, Sören Sonnenburg, Manfred Warmuth

Thank You!
C. Cortes and V.N. Vapnik.  
Support vector networks.  

A. Demiriz, K.P. Bennett, and J. Shawe-Taylor.  
Linear programming boosting via column generation.  

Y. Freund and R.E. Schapire.  
A decision-theoretic generalization of on-line learning and an application to boosting.  

R. Hettich and K.O. Kortanek.  
Semi-infinite programming: Theory, methods and applications.  

R. Meir and G. Rätsch.  
An introduction to boosting and leveraging.

G. Rätsch.

*Robust Boosting via Convex Optimization.*

G. Rätsch, A. Demiriz, and K. Bennett.
Sparse regression ensembles in infinite and finite hypothesis spaces.

G. Rätsch and M.K. Warmuth.
Efficient margin maximization with boosting.

Totally corrective boosting algorithms that maximize the margin.