

# Training-Time Optimization of a Budgeted Booster

Yi Huang

\*Brian Powers

Lev Reyzin

University of Illinois at Chicago

*{yhuang,bpower6,lreyzin}@math.uic.edu*

July 30, 2015

# Motivation: Making Predictions with a Budget

We must classify a test example but can't afford to know all the facts.

Features may be **costly** to observe

- Time,
- Money,
- Energy,
- Health risk

Motivating scenarios:

- Medical diagnosis,
- Internet applications,
- Mobile devices

# Feature-Efficient Learners

Goal: Supervised Learning Algorithm with:

- Budget  $B > 0$
- Feature costs  $C : [1, \dots, n] \rightarrow \mathbb{R}^+$
- Limited by budget at test time

We call such a learner **feature-efficient**.

# A Sampling of Related Work

- **Sequential analysis:** When to stop sequential clinical trials  
[Wald 47] and [Chernoff '72]
- **PAC learning** with incomplete features  
[Ben-David-Dichterman '93] and [Greiner et al. '02]
- Robust prediction with **missing features**  
[Globerson-Roweis '06]
- Learning **linear functions** by few features  
[Cesa-Bianchi et al. '10]
- Incorporating **feature costs** in CART impurity  
[Xu et al. '12]
- **MDPs** for feature selection [He et al. '13]

# Idea: A Feature-Efficient Boosting Algorithm

An approach using Random Sampling [Reyzin '11]:

- 1 Run AdaBoost to produce an ensemble predictor.
- 2 Sample from ensemble randomly until budget is reached.
- 3 Take importance-weighted average vote of samples.

Performance converges to that of AdaBoost as  $B \rightarrow \infty \dots$

But **is there room for improvement?**

# Budgeted Training

**Yes!**

“Budgeted Training” uses the following principles:

- Use the budget to optimize training.
- Stop training early when budget runs out.
  - The resulting predictor will be feature-efficient.
- Modify base learner selection when costs are non-uniform.

# Algorithm: AdaBoost

AdaBoost ( $S$ ) where:  $S \subset X \times \{-1, +1\}$

- 1: given:  $(x_1, y_1), \dots, (x_m, y_m) \in S$
- 2: initialize  $D_1(i) = \frac{1}{m}$
- 3: **for**  $t = 1, \dots, T$  **do**
- 4:   train base learner using distribution  $D_t$ .
- 5:   get  $h_t \in \mathcal{H} : X \rightarrow \{-1, +1\}$ .
  
- 6:   choose  $\alpha_t = \frac{1}{2} \ln \frac{1+\gamma_t}{1-\gamma_t}$ , where  $\gamma_t = \sum_i D_t(i) y_i h_t(x_i)$ .
- 7:   update  $D_{t+1}(i) = D_t(i) \exp(\alpha_t y_i h_t(x_i)) / Z_t$ ,
- 8: **end for**
- 9: output the final classifier  $H(x) = \text{sign} \left( \sum_{t=1}^T \alpha_t h_t(x) \right)$

# Algorithm: AdaBoost with Budgeted Training

AdaBoostBT( $S, B, C$ ) where:  $S \subset X \times \{-1, +1\}$ ,  $B > 0$ ,  
 $C : [n] \rightarrow \mathbb{R}^+$

- 1: given:  $(x_1, y_1), \dots, (x_m, y_m) \in S$
- 2: initialize  $D_1(i) = \frac{1}{m}$ ,  $B_1 = B$
- 3: **for**  $t = 1, \dots, T$  **do**
- 4:   train base learner using distribution  $D_t$ .
- 5:   get  $h_t \in \mathcal{H} : X \rightarrow \{-1, +1\}$ .
- 6:   **if** the total cost of the unpaid features of  $h_t$  exceeds  $B_t$   
    **then**
- 7:       set  $T = t - 1$  and **end for**
- 8:   **else** set  $B_{t+1}$  as  $B_t$  minus the total cost of the unpaid  
    features of  $h_t$ , marking them as paid
- 9:   choose  $\alpha_t = \frac{1}{2} \ln \frac{1+\gamma_t}{1-\gamma_t}$ , where  $\gamma_t = \sum_i D_t(i) y_i h_t(x_i)$ .
- 10:   update  $D_{t+1}(i) = D_t(i) \exp(\alpha_t y_i h_t(x_i)) / Z_t$ ,
- 11: **end for**
- 12: output the final classifier  $H(x) = \text{sign} \left( \sum_{t=1}^T \alpha_t h_t(x) \right)$



# Selection of Weak Learners

In AdaBoost, weak learners are selected to drive down the training error bound [Freund & Schapire '97]

$$\hat{P}_r[H(x) \neq y] \leq \prod_{t=1}^T \sqrt{1 - \gamma_t^2}.$$

- If costs are uniform ( $T$  is known), choose the weak learner that maximizes  $|\gamma_t|$ .
- If costs are non-uniform:
  - High edges give smaller terms, but
  - Low costs allow for more terms in the product.
  - **How should we trade-off edge vs cost?**

# A Greedy Optimization

To estimate  $T$  we assume **future rounds will be like the current**.

$$\text{So } T = \frac{B}{c(h)}.$$

Then the selection becomes

$$h_t = \operatorname{argmin}_{h \in \mathcal{H}} (1 - \gamma_t(h)^2)^{\frac{1}{c(h)}}. \quad (1)$$

# A Smoother Optimization

Alternate estimate of  $T$  based on milder assumption: **The cost of future rounds will be the average cost so far.**

The resulting selection rule is

$$h_t = \operatorname{argmin}_{h \in \mathcal{H}} (1 - \gamma_t(h)^2)^{\frac{1}{(B - B_t) + c(h)}}. \quad (2)$$

Idea: Using average cost should produce a smoother optimization.

# A Look at SpeedBoost

SpeedBoost [Grubb-Bagnell '12] produces a feature-efficient ensemble in another way.

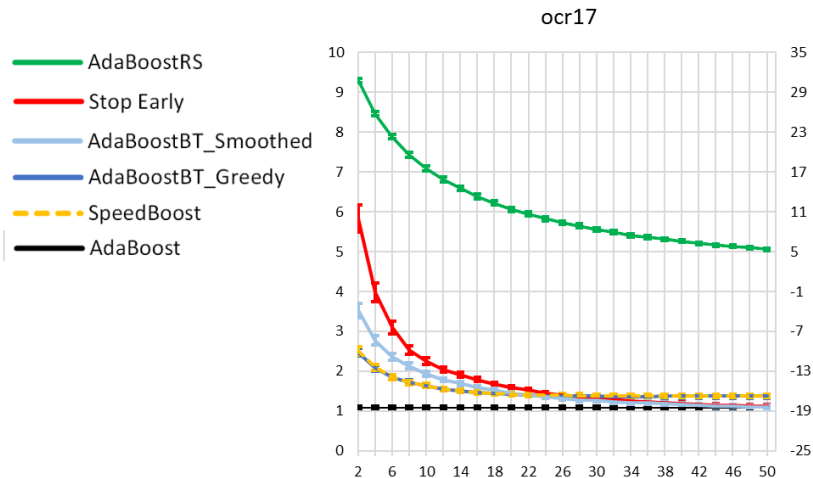
An objective  $R$  is chosen (e.g. a loss function).

While the budget allows:

A Weak learner  $h$  and weight  $\alpha$  are chosen to maximize

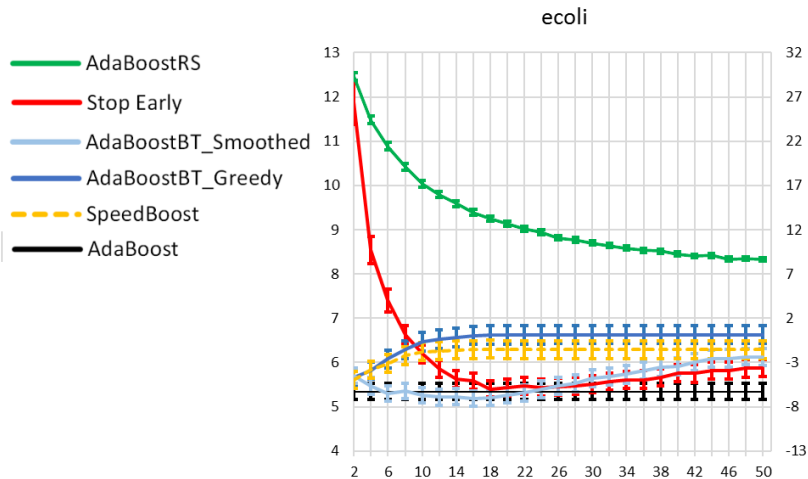
$$\frac{R(f_{i-1}) - R(f_{i-1} + \alpha h)}{c(h)}.$$

# Experimental Results: $C \sim Unif(0, 2)$



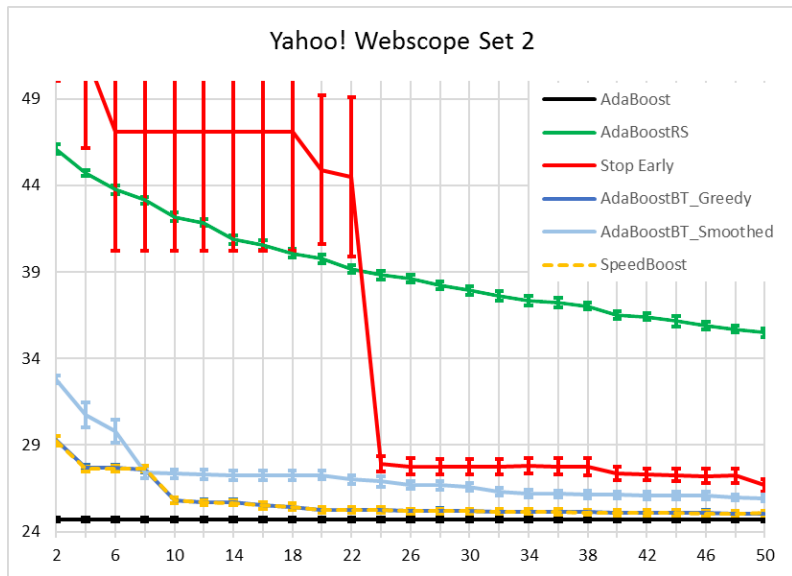
Budget on horizontal axis, test error rate on vertical (AdaBoostRS error on right). AdaBoost at T=400 as a benchmark.

# Experimental Results: $C \sim Unif(0, 2)$



Budget on horizontal axis, test error rate on vertical (AdaBoostRS error on right). AdaBoost at T=400 as a benchmark.

# Experimental Results: Real World Data



# Observations

- Budgeted training improves significantly on AdaBoostRS.
- Modifying with Greedy and Smoothed optimizations tend to yield additional improvements:
  - Greedy tends to win for small budgets.
  - Smoothed tends to win for larger budgets.
- SpeedBoost and our Greedy Budgeted Training perform almost identically.
  - There is an explanation using a Taylor series expansion.



# Observations

- Too many cheap features can kill Greedy Optimization.
- Smoothed optimization avoids this trap, since cost becomes less important as  $t \rightarrow \infty$ .
- Both Greedy and Smoothed optimizations run a higher risk of over-fitting than simply stopping early.

# Future Work

- Improve optimization for cost distributions with few cheap features.
- Consider adversarial cost models.
- Refine optimizations by considering the complexity term in AdaBoost's generalization error bound.
- Study making other machine learning algorithms feature-efficient through budgeted training.

# Thank you

# Visit my poster at Panel 4

# Thank you!

**Training-Time Optimization of a Budgeted Booster**

**UIC**  
UNIVERSITY OF ILLINOIS  
CHICAGO

Yi Huang, Brian Powers, Leo Hayes  
[yhuang, bpowers, leohay]@research.illinois.edu

**Problem Setting**  
Standard supervised learning with Bayesian cost items:  
• Training examples  $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$   
• Feature cost function  $c(x) = \sum_{i=1}^n c_i(x_i)$   
• Test time budget  $B > 0$   
Challenge:  
Provide an example optimal budget

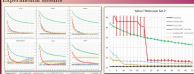
**Random Sampling**  
achieved by Exercise 11:  
1. Train  $n$ -feature using  $\text{AdaBoost}$   
2. Randomly sample from candidate features  
3. Pay for each input feature until budget is reached  
4. The optimized cost of candidate features

**Budgeted Training**  
• Consider costs during training  
• Train training set once or budget is reached  
• Randomly choose until stay budget  
• The only truly useful statement for budgeted training


**Cost Threshold Equations**  
Step: Add new feature  
• Check  $c$  with maximum  $\gamma$   
• Does not probe beyond budget  
Mathematical Formulation:  
• Goal: change hypothesis to follow better training set  
• Cost function value  $\gamma$  is unknown  
• Estimate  $\gamma$  by assuming feature results will have same cost as input  
• Step feature is chosen to minimize:  
$$N_i \cdot \exp\left(-\frac{\gamma_i^2}{2\sigma_i^2}\right) \quad (1)$$
  
• Package an approximate approximation?  
Mathematical Formulation:  
• Estimate  $\gamma$  by assuming feature results will have average cost  
• Step feature is chosen to minimize:  
$$N_i \cdot \exp\left(-\frac{1}{2} \cdot \frac{\gamma_i^2}{\sigma_i^2 + \epsilon}\right) \quad (2)$$
  
• While approximations should smooth optimization

**A Margin Bound Justification**  
Doesn't verify the "specificity" of each feature's "quality" but is a guarantee that was a guarantee until "margin" length  $\ell$  is equal to  $\ell$ . The margin bound is:  
$$\text{Pr}(\text{Error} \leq \epsilon) \leq \text{Pr}(\text{Error} \leq \epsilon) + \epsilon \cdot \left(\frac{1}{\sqrt{\pi}}\right)$$
  
where  $\text{Pr}(\text{Error} \leq \epsilon) = \sum_{i=1}^n \text{Pr}(\text{Error}_i \leq \epsilon)$ . The first term can be bounded by:  
$$\text{Pr}(\text{Error} \leq \epsilon) \leq \sum_{i=1}^n \text{Pr}(\text{Error}_i \leq \epsilon) = \sum_{i=1}^n \text{Pr}(\text{Error}_i \leq \epsilon)$$
  
The cost of this trade is:  
$$\prod_{i=1}^n \frac{1}{\sqrt{\pi}} \cdot \sqrt{\sigma_i^2 + \epsilon}$$

**Algorithm: AdaBoost with Budgeted Training**  
AdaBoost( $\mathcal{D}, B, \ell$ ), where  $\mathcal{D} = \{x_i, y_i\}_{i=1}^n, B = \sum_{i=1}^n c_i(x_i), \ell = \frac{B}{\sigma}$   
1.  $\text{Pr}(\text{Error} \leq \epsilon) \leq \sum_{i=1}^n \text{Pr}(\text{Error}_i \leq \epsilon)$   
2.  $\text{Pr}(\text{Error} \leq \epsilon) \leq \sum_{i=1}^n \text{Pr}(\text{Error}_i \leq \epsilon)$   
3. For  $i = 1, \dots, \ell$ :  
•  $\sigma_i$  = width base learner using distribution  $D_i$ , get  $h_i, \sigma_i, \gamma_i = \frac{1}{2} \cdot \frac{\sigma_i^2}{\sigma_i^2 + \epsilon}$   
• If the total cost of the input features of  $h_i$ , exceeds  $B$ , then  
•  $\sigma_i^2 = \frac{1}{2} \cdot \frac{\sigma_i^2}{\sigma_i^2 + \epsilon}$  and end for  
• else set  $D_{i+1}$  on  $\mathcal{D}$ , using the total cost of the input features of  $h_i$ , mark them to cost  
• set  $\sigma_{i+1} = \frac{1}{2} \cdot \frac{\sigma_i^2}{\sigma_i^2 + \epsilon}$ , where  $\sigma_i = \frac{1}{2} \cdot \frac{\sigma_i^2}{\sigma_i^2 + \epsilon}$   
• update  $\text{Pr}(\text{Error} \leq \epsilon) = \text{Pr}(\text{Error} \leq \epsilon) + \epsilon \cdot \left(\frac{1}{\sqrt{\pi}}\right)$ , in the next iteration  
• end for  
• output the best classifier  $H(\text{Error} \leq \epsilon) = \sum_{i=1}^{\ell} h_i(x_i)$

**Experimental Results**  
  
Figure 1: Experimental results with 500 random forests built comparing our approach to AdaBoost and Random Forest. The size is calculated as budget percentage of  $B$ . The feature cost is uniformly distributed in the interval  $[0, 1]$  (not an easy target). Random forest budgeted cost is more than AdaBoost over time and the high level cost can be used this way.

**Observations**  
Comparison to AdaBoost  
• Budgeted Training requires significantly less AdaBoost  
• Clearly and bounded optimization lead to high AdaBoost performance  
• Clearly trade to this for small budget  
• Bounded trade to this for larger budget  
• Both use higher level of searching than AdaBoost  
Impact of Budget Size  
• Training time for budgeted training is less than AdaBoost  
• More many cheap features can still clearly optimization (more  $\ell$ )  
• Bounded trade the step cost between low budgeted value  $\epsilon = \epsilon$   
The Cheap Feature Step  
• Too many cheap features can still clearly optimization (more  $\ell$ )  
• Bounded trade the step cost between low budgeted value  $\epsilon = \epsilon$   
Value "Welfare" Data  
• The higher production bonus with cost of  $B$   
• Clearly effective between AdaBoost and the optimization  
• Clearly and bounded results possible for budgeted training

**Discussion: Trade**  
•  $\ell$  (FFT) Decision tree, an efficient solution, but to cover complete generalization error  
  
Figure 2: Error Rate of Decision tree. The between cost is a representation of cost. The optimal trade point is the point where the AdaBoost cost will be less than the cost of the Decision tree.

**References**  
[1] The Boosting Book of Mathematics, Chapter 10: Boosting with Feature Costs, pp. 101-110, 2011.  
[2] Robert A. Holte, "Fast Decision Tree Methods with Feature Costs," pp. 101-110, 2011.  
[3] Robert A. Holte, "Fast Decision Tree Methods with Feature Costs," pp. 101-110, 2011.  
[4] Robert A. Holte, "Fast Decision Tree Methods with Feature Costs," pp. 101-110, 2011.  
[5] Robert A. Holte, "Fast Decision Tree Methods with Feature Costs," pp. 101-110, 2011.