Learning Prediction Suffix Trees with Winnow

Nikos Karampatziakis Dexter Kozen

Cornell University

June 15, 2009

Introduction

Sequential prediction

Introduction

Sequential prediction

Algorithm that learns small and accurate Prediction Suffix Trees (PSTs)

Introduction

Sequential prediction

Algorithm that learns small and accurate Prediction Suffix Trees (PSTs)

Our task: Is a program behaving normally?

Introduction

Sequential prediction

Algorithm that learns small and accurate Prediction Suffix Trees (PSTs)

Our task: Is a program behaving normally?

- Monitor a program and take appropriate actions based on the actual system calls and the predicted ones

Introduction

Sequential prediction

Algorithm that learns small and accurate Prediction Suffix Trees (PSTs)

Our task: Is a program behaving normally?

- Monitor a program and take appropriate actions based on the actual system calls and the predicted ones
- Deviations may signify a bug, a security problem, etc.

Sequential Prediction

Known alphabet e.g. {A, C, G, T} or $\{-1, +1\}$ or {open(), read(), …} or …

Sequential Prediction

Known alphabet e.g. {A, C, G, T} or $\{-1, +1\}$ or {open(), read(), ...} or ...

Given $y_1, y_2, \ldots, y_{t-2}, y_{t-1}$ predict y_t

Sequential Prediction

Known alphabet e.g. {A, C, G, T} or $\{-1, +1\}$ or {open(), read(), ...} or ...

Given $y_1, y_2, \ldots, y_{t-2}, y_{t-1}$ predict y_t

PSTs (aka Context Trees) are popular models for this task [Pereira & Singer, 1999, Ron et al., 1996, Willems et al., 1995]

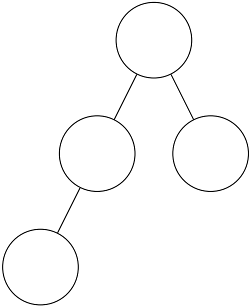
Prediction Suffix Trees

Assume $y_t \in \{-1, +1\}$

Prediction Suffix Trees

Assume $y_t \in \{-1, +1\}$

Now, a PST is a binary tree

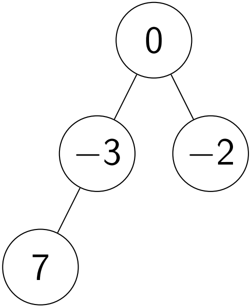


Prediction Suffix Trees

Assume $y_t \in \{-1, +1\}$

Now, a PST is a binary tree

Each node has a value



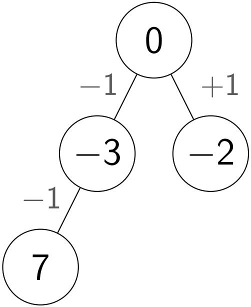
Prediction Suffix Trees

Assume $y_t \in \{-1, +1\}$

Now, a PST is a binary tree

Each node has a value

Left links labeled -1, right $+1$



Prediction Suffix Trees

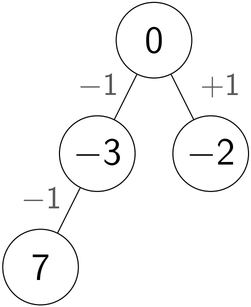
Assume $y_t \in \{-1, +1\}$

Now, a PST is a binary tree

Each node has a value

Left links labeled -1, right $+1$

To predict y_t follow the path labeled $y_{t-1}, y_{t-2}, \ldots$



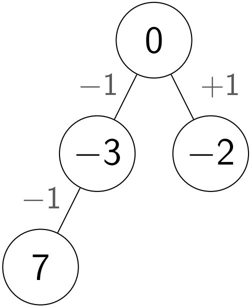
Prediction Suffix Trees

Assume $y_t \in \{-1, +1\}$

Now, a PST is a binary tree

Each node has a value

Left links labeled -1, right $+1$



To predict y_t follow the path labeled $y_{t-1}, y_{t-2}, \ldots$

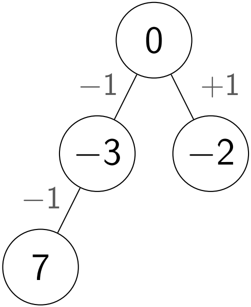
y_t is the sign of a weighted sum of visited values

Prediction Suffix Trees

Assume $y_t \in \{-1, +1\}$

Now, a PST is a binary tree

Each node has a value



Left links labeled -1, right $+1$

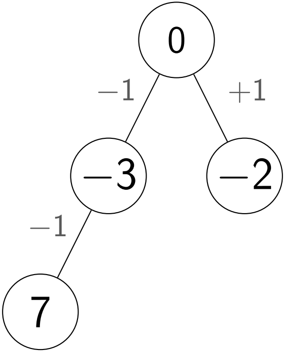
To predict y_t follow the path labeled $y_{t-1}, y_{t-2}, \ldots$

y_t is the sign of a weighted sum of visited values

Earlier symbols are discounted more than recent ones

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

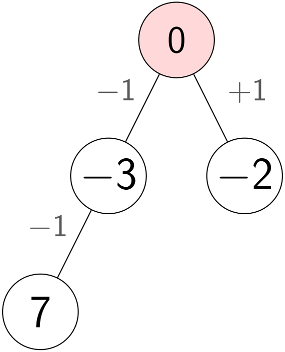


Input Sequence:

Decision:

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

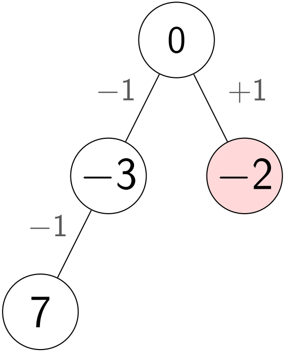


Input Sequence: $\ldots, +1$

Decision: 0

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

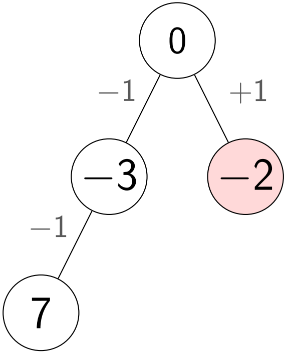


Input Sequence: $\ldots, +\mathbf{1}$

Decision: $0 + \frac{1}{2} \cdot -2$

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

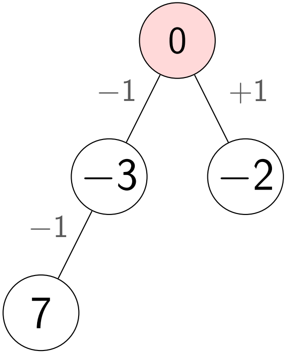


Input Sequence: $\ldots, +\mathbf{1}$

Decision: $0 + \frac{1}{2} \cdot -2 = -1 \xrightarrow{\textit{sign}} -1$

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

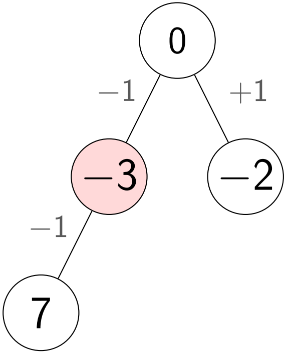


Input Sequence: $\ldots, +1, -1$

Decision: 0

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

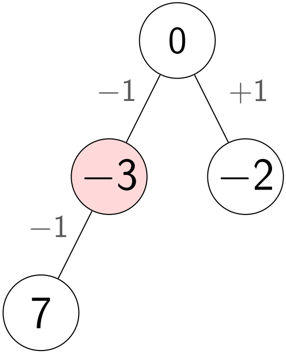


Input Sequence: $\ldots, +1, \mathbf{-1}$

Decision: $0 + \frac{1}{2} \cdot -3$

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

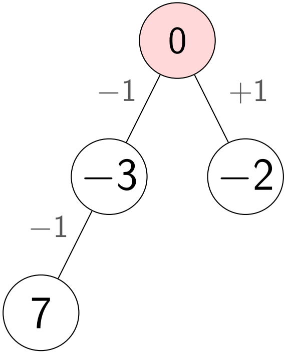


Input Sequence: $\ldots, +1, \mathbf{-1}$

Decision: $0 + \frac{1}{2} \cdot -3 = -\frac{3}{2} \xrightarrow{\textit{sign}} -1$

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

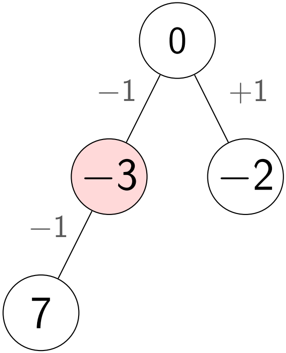


Input Sequence: $\ldots, -1, -1$

Decision: 0

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

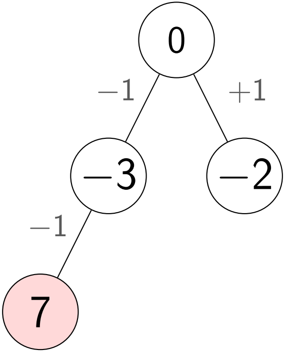


Input Sequence: $\ldots, -1, -\mathbf{1}$

Decision: $0 + \frac{1}{2} \cdot -3$

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$

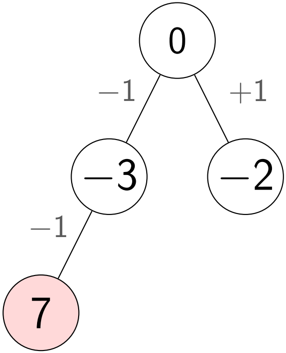


Input Sequence: $\ldots, -\mathbf{1}, -1$

Decision: $0 + \frac{1}{2} \cdot -3 + \frac{1}{4} \cdot 7$

Example

Discounting: Values discounted by $\left(\frac{1}{2}\right)^{\text{depth}}$



Input Sequence: $\ldots, -\mathbf{1}, -1$

Decision: $0 + \frac{1}{2} \cdot -3 + \frac{1}{4} \cdot 7 = \frac{1}{4} \xrightarrow{\textit{sign}} +1$

A Linear Prediction Problem

Node indexed by the suffix s leading to it

A Linear Prediction Problem

Node indexed by the suffix s leading to it

Let $w_{t,s}$ be the values in the tree at time t

A Linear Prediction Problem

Node indexed by the suffix s leading to it

Let $w_{t,s}$ be the values in the tree at time t

Decision can be written as $\operatorname{sign}(\langle w_t, x_t^+\rangle)$

$$x_{t,s}^+ = \begin{cases} \beta^{|s|} & \text{if } s \text{ is a suffix of } y_1, \ldots, y_{t-1} \\ 0 & \text{otherwise} \end{cases}$$

A Linear Prediction Problem

Node indexed by the suffix s leading to it

Let $w_{t,s}$ be the values in the tree at time t

Decision can be written as $\text{sign}(\langle w_t, x_t^+ \rangle)$

$$x_{t,s}^+ = \begin{cases} \beta^{|s|} & \text{if } s \text{ is a suffix of } y_1, \ldots, y_{t-1} \\ 0 & \text{otherwise} \end{cases}$$

Online setting

Balanced Winnow

$\theta_1 \leftarrow 0$

for $t = 1, \ldots, T$ **do**

$$w_{t,i} \leftarrow \frac{e^{\theta_{t,i}}}{\sum_j e^{\theta_{t,j}}}$$

$\hat{y}_t \leftarrow \langle w_t, x_t \rangle$

if $y_t \hat{y}_t \leq 0$

$\theta_{t+1} \leftarrow \theta_t + \alpha y_t x_t$

else

$\theta_{t+1} \leftarrow \theta_t$

Balanced Winnow

$\theta_1 \leftarrow 0$

for $t = 1, \ldots, T$ **do**

$\quad w_{t,i} \leftarrow \dfrac{e^{\theta_{t,i}}}{\sum_j e^{\theta_{t,j}}}$

$\quad \hat{y}_t \leftarrow \langle w_t, x_t \rangle$

$\quad$ **if** $y_t \hat{y}_t \leq 0$

$\qquad \theta_{t+1} \leftarrow \theta_t + \alpha y_t x_t$

$\quad$ **else**

$\qquad \theta_{t+1} \leftarrow \theta_t$

Important

$x_t = [x_t^+, -x_t^+] =$
$[x_{t,1}^+, \ldots, x_{t,d}^+, -x_{t,1}^+, \ldots, -x_{t,d}^+]$

Balanced Winnow

$\theta_1 \leftarrow 0$

for $t = 1, \ldots, T$ **do**

$$w_{t,i} \leftarrow \frac{e^{\theta_{t,i}}}{\sum_j e^{\theta_{t,j}}}$$

$\hat{y}_t \leftarrow \langle w_t, x_t \rangle$

if $y_t \hat{y}_t \leq 0$

$\theta_{t+1} \leftarrow \theta_t + \alpha y_t x_t$

else

$\theta_{t+1} \leftarrow \theta_t$

Important

$x_t = [x_t^+, -x_t^+] =$
$[x_{t,1}^+, \ldots, x_{t,d}^+, -x_{t,1}^+, \ldots, -x_{t,d}^+]$

Known fact

Let $\theta_t = [\theta_t^+, \theta_t^-]$ then
$\langle w_t, x_t \rangle \propto \sum_i \sinh(\theta_{t,i}^+) x_{t,i}^+$

$\sinh(\theta_{t,i}^+) = 0$ iff $\theta_{t,i}^+ = 0$

Learning a PST

Winnow can learn good θ_t^+ values for the tree

Learning a PST

Winnow can learn good θ_t^+ values for the tree

To keep the tree small θ_t^+ must be sparse

Learning a PST

Winnow can learn good θ_t^+ values for the tree

To keep the tree small θ_t^+ must be sparse

Initially $\theta_1 = 0$. The tree has one node

Learning a PST

Winnow can learn good θ_t^+ values for the tree

To keep the tree small θ_t^+ must be sparse

Initially $\theta_1 = 0$. The tree has one node

As mistakes are made, the tree grows

Learning a PST

Winnow can learn good θ_t^+ values for the tree

To keep the tree small θ_t^+ must be sparse

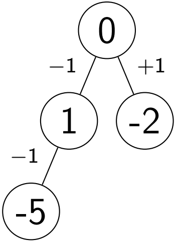
Initially $\theta_1 = 0$. The tree has one node

As mistakes are made, the tree grows

Winnow/Perceptron update quickly leads to large trees

Illustration

Input: $\ldots, -1, +1, -1, +1, -1, ?$

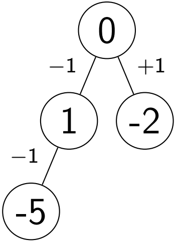


$$x^+_{t,s} = \begin{cases} \left(\frac{1}{2}\right)^{|s|} & \text{if } s \text{ is a suffix} \\ 0 & \text{otherwise} \end{cases}$$

Decision:

Illustration

Input: $\ldots, -1, +1, -1, +1, -1, ?$

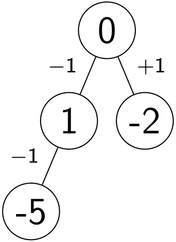


$$x^+_{t,s} = \begin{cases} \left(\frac{1}{2}\right)^{|s|} & \text{if } s \text{ is a suffix} \\ 0 & \text{otherwise} \end{cases}$$

Decision: $\frac{1}{2} \cdot \sinh(1)$

Illustration

Input: $\ldots, -1, +1, -1, +1, -1, ?$

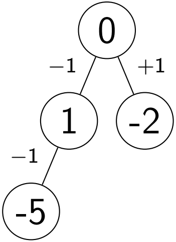


$$x^+_{t,s} = \begin{cases} \left(\frac{1}{2}\right)^{|s|} & \text{if } s \text{ is a suffix} \\ 0 & \text{otherwise} \end{cases}$$

Decision: $\frac{1}{2} \cdot \sinh(1) > 0$

Illustration

Input: $\ldots, -1, +1, -1, +1, -1, ?$

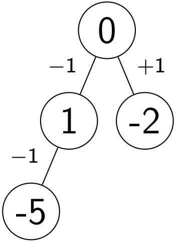


$$x^{+}_{t,s} = \begin{cases} \left(\frac{1}{2}\right)^{|s|} & \text{if } s \text{ is a suffix} \\ 0 & \text{otherwise} \end{cases}$$

Decision: $\frac{1}{2} \cdot \sinh(1) > 0 \overset{sign}{\rightarrow} +1$

Illustration

Input: $\ldots, -1, +1, -1, +1, -1, \ -1$



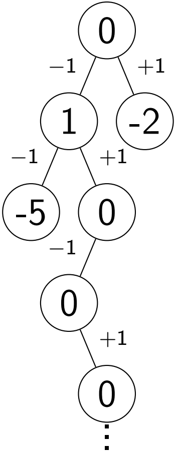
$$x^+_{t,s} = \begin{cases} \left(\frac{1}{2}\right)^{|s|} & \text{if } s \text{ is a suffix} \\ 0 & \text{otherwise} \end{cases}$$

Decision: $\frac{1}{2} \cdot \sinh(1) > 0 \xrightarrow{sign} +1$

Update: $\theta^+_{t+1,s} = \theta^+_{t,s} - x^+_{t,s}$

Illustration

Input: $\ldots, -1, +1, -1, +1, -1, -1$



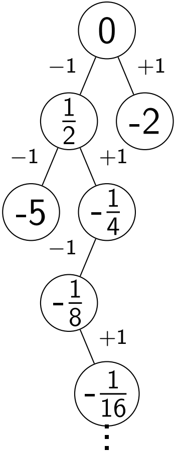
$$x^+_{t,s} = \begin{cases} \left(\frac{1}{2}\right)^{|s|} & \text{if } s \text{ is a suffix} \\ 0 & \text{otherwise} \end{cases}$$

Decision: $\frac{1}{2} \cdot \sinh(1) > 0 \xrightarrow{sign} +1$

Update: $\theta^+_{t+1,s} = \theta^+_{t,s} - x^+_{t,s}$

Illustration

Input: $\ldots, -1, +1, -1, +1, -1, -1$



$$x^+_{t,s} = \begin{cases} \left(\frac{1}{2}\right)^{|s|} & \text{if } s \text{ is a suffix} \\ 0 & \text{otherwise} \end{cases}$$

Decision: $\frac{1}{2} \cdot \sinh(1) > 0 \stackrel{sign}{\rightarrow} +1$

Update: $\theta^+_{t+1,s} = \theta^+_{t,s} - x^+_{t,s}$

Observations and Ideas

Mistake at time t: $O(t)$ nodes are inserted

Observations and Ideas

Mistake at time t: $O(t)$ nodes are inserted

$x^{+}_{t,s}$ is non-zero even when s is very long

Observations and Ideas

Mistake at time t: $O(t)$ nodes are inserted

$x^{+}_{t,s}$ is non-zero even when s is very long

Bad idea: change $x^{+}_{t,s}$ to avoid this

Observations and Ideas

Mistake at time t: $O(t)$ nodes are inserted

$x_{t,s}^{+}$ is non-zero even when s is very long

Bad idea: change $x_{t,s}^{+}$ to avoid this

Better idea: Have an adaptive bound d_t on the depth up to which the tree can grow on round t.

Observations and Ideas

Mistake at time t: $O(t)$ nodes are inserted

$x^+_{t,s}$ is non-zero even when s is very long

Bad idea: change $x^+_{t,s}$ to avoid this

Better idea: Have an adaptive bound d_t on the depth up to which the tree can grow on round t.

- d_t will be growing slowly if necessary

Winnow for PSTs

New update: $\theta_{t+1} = \theta_t + \alpha y_t x_t + \alpha n_t$

Winnow for PSTs

New update: $\theta_{t+1} = \theta_t + \alpha y_t x_t + \alpha n_t$

n_t: a "noise" vector that prunes the tree

$$n_{t,s} = \begin{cases} -y_t x_{t,s} & \text{if } |s| > d_t, \ s \text{ suffix} \\ 0 & \text{otherwise} \end{cases}$$

Winnow for PSTs

New update: $\theta_{t+1} = \theta_t + \alpha y_t x_t + \alpha n_t$

n_t: a "noise" vector that prunes the tree

$$n_{t,s} = \begin{cases} -y_t x_{t,s} & \text{if } |s| > d_t,\ s \text{ suffix} \\ 0 & \text{otherwise} \end{cases}$$

Let $P_t = \sum_{i \in J_t} ||n_i||_\infty = \sum_{i \in J_t} \beta^{d_i+1}$. J_t is the set of rounds up to t in which mistakes were made

Winnow for PSTs

New update: $\theta_{t+1} = \theta_t + \alpha y_t x_t + \alpha n_t$

n_t: a "noise" vector that prunes the tree

$$n_{t,s} = \begin{cases} -y_t x_{t,s} & \text{if } |s| > d_t,\ s \text{ suffix} \\ 0 & \text{otherwise} \end{cases}$$

Let $P_t = \sum_{i \in J_t} ||n_i||_\infty = \sum_{i \in J_t} \beta^{d_i+1}$. J_t is the set of rounds up to t in which mistakes were made

- P_t is the effect of noise in the analysis

Winnow for PSTs

New update: $\theta_{t+1} = \theta_t + \alpha y_t x_t + \alpha n_t$

n_t: a "noise" vector that prunes the tree

$$n_{t,s} = \begin{cases} -y_t x_{t,s} & \text{if } |s| > d_t,\ s \text{ suffix} \\ 0 & \text{otherwise} \end{cases}$$

Let $P_t = \sum_{i \in J_t} ||n_i||_\infty = \sum_{i \in J_t} \beta^{d_i+1}$. J_t is the set of rounds up to t in which mistakes were made

- P_t is the effect of noise in the analysis

d_t is set to guarantee $P_t \leq |J_t|^{2/3}$. Suffices to set

$$d_t = \left\lceil \log_\beta \left(\sqrt[3]{P_{t-1}^3 + 2P_{t-1}^{3/2} + 1} - P_{t-1} \right) - 1 \right\rceil$$

Theorems

Mistake Bound

Let there be a tree u ($||u||_1 = 1, u_i \geq 0$) which over the input sequence $y_1, y_2, \ldots, y_T$ attains loss $L = \sum_{t=1}^{T} \max(0, \delta - y_t \langle u, x_t \rangle)$, then our algorithm's mistakes M_T will be at most

$$\max\left\{\frac{2L}{\delta} + \frac{8\log T}{\delta^2}, \frac{64}{\delta^3}\right\}$$

Theorems

Mistake Bound and Growth Bound

Let there be a tree u ($||u||_1 = 1, u_i \geq 0$) which over the input sequence $y_1, y_2, \ldots, y_T$ attains loss $L = \sum_{t=1}^{T} \max(0, \delta - y_t\langle u, x_t\rangle)$, then our algorithm's mistakes M_T will be at most

$$\max\left\{\frac{2L}{\delta} + \frac{8\log T}{\delta^2}, \frac{64}{\delta^3}\right\}$$

Moreover, by setting $\beta = 2^{-1/3}$, the learned tree will have at most $\log_2(M_T) + 4$ levels

Proof Sketch

Growth bound is straightforward

Proof Sketch

Growth bound is straightforward

Mistake bound via potential function

Proof Sketch

Growth bound is straightforward

Mistake bound via potential function

$$\Phi(w_t) = \sum_i u_i \log \frac{u_i}{w_{t,i}} \geq 0$$

Proof Sketch

Growth bound is straightforward

Mistake bound via potential function

$$\Phi(w_t) = \sum_i u_i \log \frac{u_i}{w_{t,i}} \geq 0$$

Upper bound $\Phi(w_1)$ and lower bound decrease in potential with each mistake:

$$\Delta\Phi = \underbrace{\text{effect of full update}}_{\geq f(\alpha,\delta,\text{loss of } u)} - \text{effect of noise}$$

Proof Sketch II

Potential: $\Phi(w_1)$ ⊢———⊣ 0

Noise P_t :

Example	Correct Prediction
x_1	

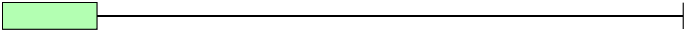
Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential:



$\Phi(w_1)$ 0

Noise P_t :

Example	Correct Prediction
x_1	✗

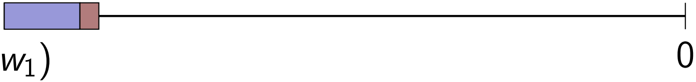
Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential:



Noise P_t :

Example	Correct Prediction
x_1	✗

Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential: $\Phi(w_1)$ 0

Noise P_t :

Example	Correct Prediction
x_1	✗
x_2	

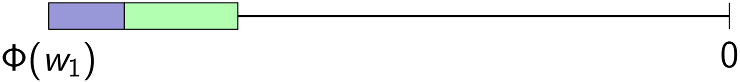
Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential:



Noise P_t :

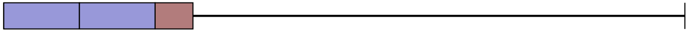
Example	Correct Prediction
x_1	✗
x_2	✗

Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential: 

$\Phi(w_1)$ 0

Noise P_t : 

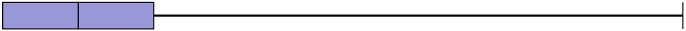
Example	Correct Prediction
x_1	✗
x_2	✗

Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential: 

$\Phi(w_1)$ 0

Noise P_t : 

Example	Correct Prediction
x_1	✗
x_2	✗
x_3	

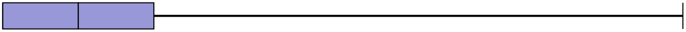
Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential:



$\Phi(w_1)$ 0

Noise P_t :

Example	Correct Prediction
x_1	✗
x_2	✗
x_3	✓

Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential:

$\Phi(w_1)$ 0

Noise P_t :

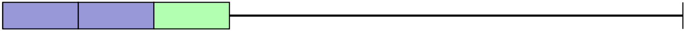
Example	Correct Prediction
x_1	✗
x_2	✗
x_3	✓
x_4	

Progress due to classic update

Net progress Effect of noise

Proof Sketch II

Potential:



$\Phi(w_1)$ 0

Noise P_t :

Example	Correct Prediction
x_1	✗
x_2	✗
x_3	✓
x_4	✗

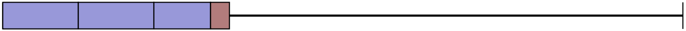
Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential:



$\Phi(w_1)$ 0

Noise P_t :

Example	Correct Prediction
x_1	✗
x_2	✗
x_3	✓
x_4	✗

Progress due to classic update

Net progress

Effect of noise

Proof Sketch II

Potential: $\Phi(w_1)$ 0

Noise P_t :

Example	Correct Prediction
x_1	✗
x_2	✗
x_3	✓
x_4	✗

Progress due to classic update

Net progress

Effect of noise

Proof Sketch III

length of $\boxed{} \leq \Phi(w_1)$

Proof Sketch III

length of $\square \leq \Phi(w_1)$

length of $\square$ $-$length of $\square \leq \Phi(w_1)$

Proof Sketch III

length of $\leq \Phi(w_1)$

$$\underbrace{\text{length of }}_{\geq \text{min } \text{ size} \times \text{mistakes}} - \text{length of } \leq \Phi(w_1)$$

Proof Sketch III

$$\text{length of } \square \leq \Phi(w_1)$$

$$\underbrace{\text{length of } \square}_{\geq \min \square \text{ size} \times \text{mistakes}} - \underbrace{\text{length of } \square}_{\leq \text{mistakes}^{2/3}} \leq \Phi(w_1)$$

Proof Sketch III

$$\text{length of } \square \leq \Phi(w_1)$$

$$\underbrace{\text{length of } \square}_{\geq \min \square \text{ size} \times \text{mistakes}} - \underbrace{\text{length of } \square}_{\leq \text{mistakes}^{2/3}} \leq \Phi(w_1)$$

$$\min \square \text{ size} \cdot \text{mistakes} - \text{mistakes}^{2/3} \leq \Phi(w_1)$$

Results

3 programs, 120 sequences of system calls

Results

3 programs, 120 sequences of system calls

Ran our **winnow** variant and [Dekel et al., 2004]'s self bounded **perceptron** for PSTs

Results

3 programs, 120 sequences of system calls

Ran our **winnow** variant and [Dekel et al., 2004]'s self bounded **perceptron** for PSTs

% Error	Outlook	Excel	Firefox
Perceptron	5.1	22.68	14.86
Winnow	4.43	20.59	13.88

Results

3 programs, 120 sequences of system calls

Ran our **winnow** variant and [Dekel et al., 2004]'s self bounded **perceptron** for PSTs

% Error	Outlook	Excel	Firefox
Perceptron	5.1	22.68	14.86
Winnow	4.43	20.59	13.88

PST Size	Outlook	Excel	Firefox
Perceptron	41239	24402	21081
Winnow	25679	15338	12662

Results

3 programs, 120 sequences of system calls

Ran our **winnow** variant and [Dekel et al., 2004]'s self bounded **perceptron** for PSTs

% Error	Outlook	Excel	Firefox
Perceptron	5.1	22.68	14.86
Winnow	4.43	20.59	13.88

PST Size	Outlook	Excel	Firefox
Perceptron	41239	24402	21081
Winnow	25679	15338	12662

Winnow makes fewer mistakes and grows smaller trees for all 120 sequences

Related Work

Much work on PSTs [Willems et al., 1995], [Ron et al., 1996], [Pereira & Singer, 1999]. . . but with assumptions on the tree structure e.g. a priori bounds on the tree's depth

Related Work

Much work on PSTs [Willems et al., 1995], [Ron et al., 1996], [Pereira & Singer, 1999]. . . but with assumptions on the tree structure e.g. a priori bounds on the tree's depth

[Dekel et al., 2004] self bounded perceptron. Similar ideas, but overfits in practice.

Related Work

Much work on PSTs [Willems et al., 1995], [Ron et al., 1996], [Pereira & Singer, 1999]. . . but with assumptions on the tree structure e.g. a priori bounds on the tree's depth

[Dekel et al., 2004] self bounded perceptron. Similar ideas, but overfits in practice.

[Shalev-Shwartz & Tewari, 2009] get sparse solutions from any p-norm algorithm

Summary

Introduced an online learning algorithm to learn PSTs

Summary

Introduced an online learning algorithm to learn PSTs

- Competitive with best fixed PST in hindsight

Summary

- Introduced an online learning algorithm to learn PSTs
  - Competitive with best fixed PST in hindsight
  - The resulting trees grow slowly if necessary

Summary

Introduced an online learning algorithm to learn PSTs

- Competitive with best fixed PST in hindsight
- The resulting trees grow slowly if necessary

On our task, it made fewer mistakes and grew smaller trees than other state-of-the-art algorithms.

References I

Dekel, O., Shalev-Shwartz, S., & Singer, Y. (2004).
The power of selective memory: Self-bounded learning of prediction suffix trees.
Advances in Neural Information Processing Systems, 17.

Pereira, F., & Singer, Y. (1999).
An Efficient Extension to Mixture Techniques for Prediction and Decision Trees.
Machine Learning, *36*, 183–199.

References II

Ron, D., Singer, Y., & Tishby, N. (1996).

The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length.

Machine Learning, *25*, 117–149.

Shalev-Shwartz, S., & Tewari, A. (2009).

Shochastic Methods for ℓ_1 Regularized Loss Minimization.

Proceedings of the 26th ICML.

References III

Willems, F., Shtarkov, Y., & Tjalkens, T. (1995).

The context-tree weighting method: basic properties.

IEEE Transactions on Information Theory, *41*, 653–664.

Differences with [Dekel et al., 2004]

Features: $\beta = 2^{-1/3}$ vs. $\beta = 2^{-1/2}$

P_t: $\sum ||n_i||_\infty$ vs. $\sum ||n_i||_2$

Tolerance: $P_t \leq M_t^{2/3}$ vs. $P_t \leq \frac{1}{2}\sqrt{M_t}$

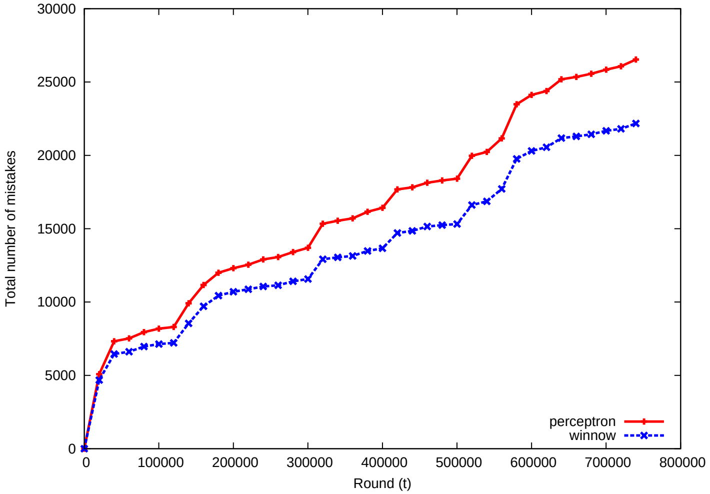
Tweaking [Dekel et al., 2004]

Setting $\beta = 2^{-1/3}$: big trees many mistakes (overfit)

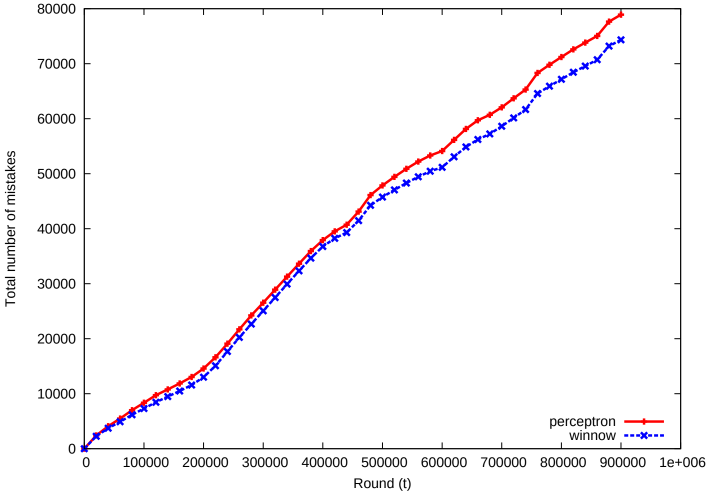
Setting $P_t \leq M_t^{2/3}$: small trees many mistakes (underfit)

Doing both: few mistakes, medium sized trees (less overfit)

More Results



More Results



More Results