Scalable Large-Margin Structured Learning: Theory and Algorithms

Liang Huang  Kai Zhao  Lemao Liu
The City University of New York (CUNY)

slides at: http://acl.cs.qc.edu/~lhuang/
What is Structured Prediction?

\[ x \]

\[ y = +1 \]

\[ y = -1 \]
What is Structured Prediction?

- binary classification: output is binary
What is Structured Prediction?

- binary classification: output is binary
- multiclass classification: output is a (small) number
What is Structured Prediction?

- **binary classification**: output is binary
- **multiclass classification**: output is a (small) number
- **structured classification**: output is a structure (seq., tree, graph)
- part-of-speech tagging, parsing, summarization, translation
What is Structured Prediction?

- binary classification: output is binary
- multiclass classification: output is a (small) number
- structured classification: output is a structure (seq., tree, graph)
  - part-of-speech tagging, parsing, summarization, translation
What is Structured Prediction?

- **binary classification**: output is binary
- **multiclass classification**: output is a (small) number
- **structured classification**: output is a structure (seq., tree, graph)
  - part-of-speech tagging, parsing, summarization, translation
  - exponentially many classes: search (inference) efficiency is crucial!
What is Structured Prediction?

- binary classification: output is binary
- multiclass classification: output is a (small) number
- structured classification: output is a structure (seq., tree, graph)
- part-of-speech tagging, parsing, summarization, translation
- exponentially many classes: search (inference) efficiency is crucial!

NLP is all about structured prediction!
Examples of Bad Structured Prediction

- Slip carefully
- CAREFUL DROWNING
- Lookout knockhead
- Slip and fall down carefully

Archived at www.ChineseEnglish.com
Learning: Unstructured vs. Structured

- Binary/multiclass
- Structured learning
- Naive Bayes
- HMMs

Generative
(count & divide)
Learning: Unstructured vs. Structured

binary/multiclass

structured learning

naive bayes

HMMs

Conditional

CRFs

logistic regression (maxent)

Conditional

generative

(count & divide)

discriminative

(expectations)
Learning: Unstructured vs. Structured

binary/multiclass

structured learning

naive bayes

HMMs

Conditional

logistic regression (maxent)

CRFs

Conditional

Online+ Viterbi

Online+ Viterbi

perceptron

structured perceptron

generative

(count & divide)

discriminative

(expectations)

(argmax)
Learning: Unstructured vs. Structured

**Binary/Multiclass**
- Naive Bayes
- Conditional logistic regression (maxent)
- SVM

**Structured Learning**
- HMMs
- CRFs
- Structured perceptron
- Structured SVM

**Generative**
- (count & divide)

**Discriminative**
- (expectations)
  - (argmax)
  - (loss-augmented argmax)
Why Perceptron (Online Learning)?

- because we want scalability on big data!
- learning time has to be linear in the number of examples
  - can make only constant number of passes over training data
  - only online learning (perceptron/MIRA) can guarantee this!
  
- SVM scales between $O(n^2)$ and $O(n^3)$; CRF no guarantee

- and inference on each example must be super fast

- another advantage of perceptron: just need argmax
Why Perceptron (Online Learning)?

- because we want scalability on big data!
- learning time has to be linear in the number of examples
  - can make only constant number of passes over training data
  - only online learning (perceptron/MIRA) can guarantee this!
- SVM scales between $O(n^2)$ and $O(n^3)$; CRF no guarantee
- and inference on each example must be super fast
- another advantage of perceptron: just need argmax

SVM

CRF
Why Perceptron (Online Learning)?

- because we want scalability on big data!
- learning time has to be linear in the number of examples
  - can make only constant number of passes over training data
  - only online learning (perceptron/MIRA) can guarantee this!
- SVM scales between $O(n^2)$ and $O(n^3)$; CRF no guarantee
- and inference on each example must be super fast
- another advantage of perceptron: just need argmax
Perceptron: from binary to structured

binary perceptron (Rosenblatt, 1959)

\[ x, y = \begin{cases} +1 & \text{if } y = +1 \\ -1 & \text{if } y = -1 \end{cases} \]

2 classes

\[ x \rightarrow y \rightarrow z \]

trivial

exact inference

update weights if \( y \neq z \)
Perceptron: from binary to structured

**Binary Perceptron**
(Rosenblatt, 1959)

2 classes

**Multiclass Perceptron**
(Freund/Schapire, 1999)

constant # of classes

**Trivial**

**Easy**

exact inference

update weights if $y \neq z$
Perceptron: from binary to structured

**Binary Perceptron**
(Rosenblatt, 1959)

- 2 classes
- Exact inference

**Multiclass Perceptron**
(Freund/Schapire, 1999)

- Constant # of classes
- Exact inference

**Structured Perceptron**
(Collins, 2002)

- Exponential # of classes
- Exact inference
Scalability Challenges

- Inference (on one example) is too slow (even w/ DP)
  - Can we sacrifice search exactness for faster learning?
  - Would inexact search interfere with learning?
  - If so, how should we modify learning?
- Even fastest inexact inference is still too slow
  - Due to too many training examples
  - Can we parallelize online learning?
Scalability Challenges

• inference (on one example) is too slow (even w/ DP)
  • can we sacrifice search exactness for faster learning?
  • would inexact search interfere with learning?
  • if so, how should we modify learning?
• even fastest inexact inference is still too slow
  • due to too many training examples
• can we parallelize online learning?
Scalability Challenges

• inference (on one example) is too slow (even w/ DP)
• can we sacrifice search exactness for faster learning?
• would inexact search interfere with learning?
• if so, how should we modify learning?
• even fastest inexact inference is still too slow
• due to too many training examples
• can we parallelize online learning?
Tutorial Outline

• Overview of Structured Learning
  • Challenges in Scalability

• Structured Perceptron
  • convergence proof

• Structured Perceptron with Inexact Search

• Latent-Variable Structured Perceptron

• Parallelizing Online Learning (Perceptron & MIRA)
Generic Perceptron

- perceptron is the simplest machine learning algorithm
- online-learning: one example at a time
- learning by doing
  - find the best output under the current weights
  - update weights at mistakes

\[
\begin{align*}
  x_i & \rightarrow \text{inference} \\
  z_i & \rightarrow y_i \\
  w & \rightarrow \text{update weights}
\end{align*}
\]
Structured Perceptron

Inputs: Training set \((x_i, y_i)\) for \(i = 1 \ldots n\)

Initialization: \(W = 0\)

Define: \(F(x) = \arg\max_{y \in \text{GEN}(x)} \Phi(x, y) \cdot W\)

Algorithm: For \(t = 1 \ldots T, i = 1 \ldots n\)

\[ z_i = F(x_i) \quad \text{If} \ (z_i \neq y_i) \]

\[ W \leftarrow W + \Phi(x_i, y_i) - \Phi(x_i, z_i) \]

Output: Parameters \(W\)

the man bit the dog

\(x_i \rightarrow \text{inference} \rightarrow z_i \rightarrow W \rightarrow \text{update weights} \rightarrow y_i \)
Example: POS Tagging

\[ \Phi(x, y) \]
\[ \Phi(x, z) \]
Example: POS Tagging

- **gold-standard:**
  - DT NN VBD DT NN
  - the man bit the dog

- **current output:**
  - DT NN NN DT NN
  - the man bit the dog

- Assume only two feature classes
  - Tag bigrams
  - Word/tag pairs
Example: POS Tagging

- gold-standard: \[ \text{DT NN VBD DT NN} \]
  \[ \text{the man bit the dog} \]

- current output: \[ \text{DT NN NN DT NN} \]
  \[ \text{the man bit the dog} \]

- assume only two feature classes
  - tag bigrams
  - word/tag pairs
Example: POS Tagging

- **gold-standard:**
  
  - DT NN VBD DT NN
  
  - the man bit the dog

- **current output:**

  - DT NN NN DT NN

  - the man bit the dog

- **assume only two feature classes**

  - **tag bigrams**

    - $t_{i-1} t_i$

  - **word/tag pairs**

    - $W_i$
### Example: POS Tagging

#### Gold-Standard:

<table>
<thead>
<tr>
<th>word</th>
<th>tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>DT</td>
</tr>
<tr>
<td>man</td>
<td>NN</td>
</tr>
<tr>
<td>bit</td>
<td>VBD</td>
</tr>
<tr>
<td>the</td>
<td>DT</td>
</tr>
<tr>
<td>dog</td>
<td>NN</td>
</tr>
</tbody>
</table>

\[
\Phi(x, y) = \Phi(x, y)
\]

#### Current Output:

<table>
<thead>
<tr>
<th>word</th>
<th>tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>DT</td>
</tr>
<tr>
<td>man</td>
<td>NN</td>
</tr>
<tr>
<td>bit</td>
<td>NN</td>
</tr>
<tr>
<td>the</td>
<td>DT</td>
</tr>
<tr>
<td>dog</td>
<td>NN</td>
</tr>
</tbody>
</table>

\[
\Phi(x, z) = \Phi(x, z)
\]

- Assume only two feature classes
  - Tag bigrams
    \[
t_{i-1} \quad t_i
\]
  - Word/tag pairs
    \[
    W_i
    \]

\[
\Phi(x, y) = \Phi(x, y)
\]
Example: POS Tagging

- gold-standard:  DT  NN  VBD  DT  NN
  the  man  bit  the  dog
  y  \( \Phi(x, y) \)

- current output:  DT  NN  NN  DT  NN
  the  man  bit  the  dog
  z  \( \Phi(x, z) \)
  x

- assume only two feature classes
- tag bigrams
- word/tag pairs
- weights ++:  (NN, VBD)  (VBD, DT)  (VBD \( \rightarrow \) bit)
Example: POS Tagging

- **gold-standard:**
  
<table>
<thead>
<tr>
<th>DT</th>
<th>NN</th>
<th>VBD</th>
<th>DT</th>
<th>NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>man</td>
<td>bit</td>
<td>the</td>
<td>dog</td>
</tr>
</tbody>
</table>

- **current output:**
  
<table>
<thead>
<tr>
<th>DT</th>
<th>NN</th>
<th>NN</th>
<th>DT</th>
<th>NN</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>man</td>
<td>bit</td>
<td>the</td>
<td>dog</td>
</tr>
</tbody>
</table>

- **assume only two feature classes**
  - tag bigrams
    
    \( t_{i-1} \rightarrow t_i \)
  - word/tag pairs
    
    \( W_i \)

- **weights ++:**
  - (NN, VBD)
  - (VBD, DT)
  - (VBD → bit)

- **weights --:**
  - (NN, NN)
  - (NN, DT)
  - (NN → bit)
Inference: Dynamic Programming

tagging: $O(nT^3)$

CKY parsing: $O(n^3)$
the inference \((\text{argmax})\) must be efficient

- either the search space \(\text{GEN}(x)\) is small, or \textit{factored}
- features must be local to \(y\) (but can be global to \(x\))
  - e.g. bigram tagger, but look at all input words (cf. CRFs)
Efficiency vs. Expressiveness

- the inference (argmax) must be efficient
  - either the search space $\text{GEN}(x)$ is small, or factored
  - features must be local to $y$ (but can be global to $x$)
    - e.g. bigram tagger, but look at all input words (cf. CRFs)
Efficiency vs. Expressiveness

- the inference (argmax) must be efficient
  - either the search space $\text{GEN}(x)$ is small, or factored
  - features must be local to $y$ (but can be global to $x$)
    - e.g. bigram tagger, but look at all input words (cf. CRFs)
Efficiency vs. Expressiveness

- The inference (argmax) must be efficient.
- Either the search space $\text{GEN}(x)$ is small, or factored.
- Features must be local to $y$ (but can be global to $x$).
  - E.g. bigram tagger, but look at all input words (cf. CRFs).
the inference (argmax) must be efficient

- either the search space $\text{GEN}(x)$ is small, or factored
- features must be local to $y$ (but can be global to $x$)
  - e.g. bigram tagger, but look at all input words (cf. CRFs)
Averaged Perceptron

- much more stable and accurate results
- approximation of voted perceptron (large-margin) (Freund & Schapire, 1999)

Inputs: Training set \((x_i, y_i)\) for \(i = 1 \ldots n\)

Initialization: \(W_0 = 0\)

Define: \(F(x) = \operatorname{arg\,max}_{y \in \text{GEN}(x)} \Phi(x, y) \cdot W\)

Algorithm: For \(t = 1 \ldots T, i = 1 \ldots n\)
\[
z_i = F(x_i) \\
\text{If } (z_i \neq y_i) \quad W_{j+1} \leftarrow W_j + \Phi(x_i, y_i) - \Phi(x_i, z_i)
\]

Output: Parameters \(W = \sum_j W_j\)
Averaging => Large Margin

• much more stable and accurate results

• approximation of voted perceptron (large-margin) (Freund & Schapire, 1999)
Efficient Implementation of Averaging

- naive implementation (running sum) doesn’t scale
- very clever trick from Daume (2006, PhD thesis)

```
Algorithm AVERAGED_STRUCTURED_PERCEPTRON(x1:N, y1:N, I)
1: \( w_0 \leftarrow \langle 0, \ldots, 0 \rangle \)
2: \( w_a \leftarrow \langle 0, \ldots, 0 \rangle \)
3: \( c \leftarrow 1 \)
4: for \( i = 1 \ldots I \) do
5:   for \( n = 1 \ldots N \) do
6:     \( \hat{y}_n \leftarrow \text{arg max}_{y \in Y} w_0^\top \Phi(x_n, y_n) \)
7:     if \( y_n \neq \hat{y}_n \) then
8:       \( w_0 \leftarrow w_0 + \Phi(x_n, y_n) - \Phi(x_n, \hat{y}_n) \)
9:     \( w_a \leftarrow w_a + c\Phi(x_n, y_n) - c\Phi(x_n, \hat{y}_n) \)
10:   end if
11:   \( c \leftarrow c + 1 \)
12: end for
13: end for
14: return \( w_0 - w_a / c \)
```

Figure 2.3: The averaged structured perceptron learning algorithm.
Efficient Implementation of Averaging

- naive implementation (running sum) doesn’t scale
- very clever trick from Daume (2006, PhD thesis)

```plaintext
Algorithm AVERAGED_STRUCTURED_PERCEPTRON(x_1:N, y_1:N, I)
1: \( w_0 \leftarrow \langle 0, \ldots, 0 \rangle \)
2: \( w_a \leftarrow \langle 0, \ldots, 0 \rangle \)
3: \( c \leftarrow 1 \)
4: for \( i = 1 \ldots I \) do
5:     for \( n = 1 \ldots N \) do
6:         \( \hat{y}_n \leftarrow \arg\max_{y \in Y} w_0^\top \Phi(x_n, y_n) \)
7:         if \( y_n \neq \hat{y}_n \) then
8:             \( w_0 \leftarrow w_0 + \Phi(x_n, y_n) - \Phi(x_n, \hat{y}_n) \)
9:             \( w_a \leftarrow w_a + c\Phi(x_n, y_n) - c\Phi(x_n, \hat{y}_n) \)
10:         end if
11:     end for
12:     \( c \leftarrow c + 1 \)
13: end for
14: return \( w_0 - w_a / c \)
```

Figure 2.3: The averaged structured perceptron learning algorithm.
Perceptron vs. CRFs

- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; \( \sim \) same accuracy

CRFs

(Lafferty et al, 2001)

\[
\sum_{(x,y) \in D} \sum_{z \in \text{GEN}(x)} \frac{\exp(w \cdot \Phi(x, z))}{Z(x)}
\]

for \( (x, y) \in D \), \( \arg\max_{z \in \text{GEN}(x)} w \cdot \Phi(x, z) \)

structured perceptron

(Collins, 2002)
Perceptron vs. CRFs

- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; ~same accuracy

CRFs (Lafferty et al, 2001)

\[
\sum_{(x,y) \in D} \sum_{z \in \text{GEN}(x)} \frac{\exp(w \cdot \Phi(x, z))}{Z(x)}
\]

stochastic gradient descent (SGD)

for \((x, y) \in D\), \(\arg\max_w w \cdot \Phi(x, z)\) for \(z \in \text{GEN}(x)\)

structured perceptron (Collins, 2002)
Perceptron vs. CRFs

- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; ~same accuracy

CRFs
- (Lafferty et al, 2001)
- stochastic gradient descent (SGD)

structured perceptron
- (Collins, 2002)
Perceptron vs. CRFs

- perceptron is online and Viterbi approximation of CRF
- simpler to code; faster to converge; ~same accuracy

CRFs (Lafferty et al, 2001)

\[
\sum_{(x, y) \in D} \sum_{z \in \text{GEN}(x)} \frac{\exp(w \cdot \Phi(x, z))}{Z(x)}
\]

for \((x, y) \in D\), \(\arg\max_{z \in \text{GEN}(x)} w \cdot \Phi(x, z)\)

structured perceptron (Collins, 2002)

stochastic gradient descent (SGD)

hard/Viterbi CRFs

online

Viterbi

Viterbi

online
Perceptron Convergence Proof

- binary classification: converges iff. data is separable
- structured prediction: converges iff. data is separable
  - there is an oracle vector that correctly labels all examples
  - one vs the rest (correct label better than all incorrect labels)
- **theorem**: if separable, then \( \# \text{ of updates} \leq \frac{R^2}{\delta^2} \)

Novikoff => Freund & Schapire => Collins

1962 1999 2002
Perceptron Convergence Proof

- binary classification: converges iff. data is separable
- structured prediction: converges iff. data is separable
  - there is an oracle vector that correctly labels all examples
- one vs the rest (correct label better than all incorrect labels)
- theorem: if separable, then \# of updates \( \leq \frac{R^2}{\delta^2} \)  
  
\[ R: \text{diameter} \]

Novikoff => Freund & Schapire => Collins
1962                1999                  2002
Perceptron Convergence Proof

• binary classification: converges iff. data is separable

• structured prediction: converges iff. data is separable
  • there is an oracle vector that correctly labels all examples
  • one vs the rest (correct label better than all incorrect labels)

• theorem: if separable, then \# of updates \( \leq \frac{R^2}{\delta^2} \)
  \( R \): diameter

Novikoff => Freund & Schapire => Collins
1962                 1999                  2002
Perceptron Convergence Proof

- binary classification: converges iff. data is separable
- structured prediction: converges iff. data is separable
  - there is an oracle vector that correctly labels all examples
  - one vs the rest (correct label better than all incorrect labels)
- theorem: if separable, then \# of updates $\leq \frac{R^2}{\delta^2}$

Novikoff => Freund & Schapire => Collins
1962                  1999                  2002
Geometry of Convergence Proof

1: repeat
2:   for each example \((x, y)\) in \(D\) do
3:     \(z \leftarrow \text{EXACT}(x, w)\)
4:     if \(z \neq y\) then
5:       \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6:   until converged
Geometry of Convergence Proof pt 1

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

\[ y \] correct label

\[ z \] 1-best

\[ x \quad \text{exact inference} \quad z \quad \text{update weights} \quad \text{if } y \neq z \]

current model \(w(k)\)
Geometry of Convergence Proof pt 1

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

perceptron update:
\[ w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z) \]
Geometry of Convergence Proof pt 1

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

update weights if \(y \neq z\)

perceptron update:
\[ w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z) \]
Geometry of Convergence Proof pt 1

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

update weights if \(y \neq z\)

perceptron update:
\[
w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z)
\]
\[
\delta \geq \delta \text{ margin}
\]

exact inference

update weights

correct label

unit oracle vector \(u\)

\(\delta\)

separation

\(\Delta \Phi(x, y, z)\)

update

new model

current model \(w^{(k)}\)

\(w^{(k+1)}\)

\(x\)

\(z\)

\(y\)
Geometry of Convergence Proof

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

perceptron update:
\[
\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta \Phi(x, y, z)
\]
\[
\mathbf{u} \cdot \mathbf{w}^{(k+1)} = \mathbf{u} \cdot \mathbf{w}^{(k)} + \mathbf{u} \cdot \Delta \Phi(x, y, z) \\
\geq \delta \quad \text{margin}
\]
(by induction)

\[
||\mathbf{w}^{k+1}|| \geq k\delta
\]
(part 1: lowerbound)
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

\(x \rightarrow \text{exact inference} \rightarrow z \rightarrow \text{update weights if } y \neq z\)

\(y\) correct label
Geometry of Convergence Proof pt 2

1: repeat
2: for each example $(x, y)$ in $D$ do
3:     $z \leftarrow \text{EXACT}(x, w)$
4:     if $z \neq y$ then
5:         $w \leftarrow w + \Delta \Phi(x, y, z)$
6: until converged
Geometry of Convergence Proof

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

update weights if \(y \neq z\)

perceptron update:
\[
\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta \Phi(x, y, z)
\]
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

**perceptron update:**
\[
\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta \Phi(x, y, z)
\]
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3:  \(z \leftarrow \text{EXACT}(x, w)\)
4:  if \(z \neq y\) then
5:     \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

perceptron update:
\[
\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta \Phi(x, y, z)
\]
\[
\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta \Phi(x, y, z)\|^2
\]
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

**perceptron update:**
\[
\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \Delta \Phi(x, y, z)
\]
\[
\|\mathbf{w}^{(k+1)}\|^2 = \|\mathbf{w}^{(k)} + \Delta \Phi(x, y, z)\|^2
\]
\[
= \|\mathbf{w}^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2
\]
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

\[\|w^{(k+1)}\|^2 = \|w^{(k)} + \Delta \Phi(x, y, z)\|^2\]

\[\leq R^2\]

perceptron update:

\[w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z)\]

update weights if \(y \neq z\)

exact inference

\(x\) 

\(y\) 

update weights if \(y \neq z\)

\(z\)

\(\Delta \Phi(x, y, z)\)

update

current model \(w^{(k)}\)

new model \(w^{(k+1)}\)

R: max diameter

1-best

exact label

update

\(x\) 

\(y\)

\(z\)
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

**exact inference**

\(y\)

\(w\)

\(x\) → \(z\)

\(\Delta \Phi(x, y, z)\)

**update weights** if \(y \neq z\)

**perceptron update:**

\[
\begin{align*}
\|w^{(k+1)}\|^2 & = \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 \\
& = \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2 w^{(k)} \cdot \Delta \Phi(x, y, z) \\
& \leq R^2
\end{align*}
\]

**diameter**
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

Exact inference

Violation: incorrect label scored higher

Perceptron update:

\[
\begin{align*}
    w^{(k+1)} &= w^{(k)} + \Delta \Phi(x, y, z) \\
    \|w^{(k+1)}\|^2 &= \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 \\
    &= \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2w^{(k)} \cdot \Delta \Phi(x, y, z) + R^2 \\
    &\leq 0
\end{align*}
\]

Violation

R: max diameter
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

Violation: incorrect label scored higher

Perceptron update:
\[
\begin{align*}
\|w^{(k+1)}\|^2 &= \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 \\
&= \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2 \langle w^{(k)}, \Delta \Phi(x, y, z) \rangle + \|\Delta \Phi(x, y, z)\|^2 \\
&\leq R^2
\end{align*}
\]

Violation:
\[
\|w^{k+1}\|^2 \leq kR^2,
\|w^{k+1}\| \leq \sqrt{kR}
\]

By induction:
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

\(x \rightarrow \text{exact inference} \rightarrow z \rightarrow \text{update weights} \quad \text{if } y \neq z\)
\(y \rightarrow \)

violation: incorrect label scored higher

perceptron update:
\[
w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z)
\]

\[
\|w^{(k+1)}\|^2 = \|w^{(k)} + \Delta \Phi(x, y, z)\|^2
\]

\[
= \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2w^{(k)} \cdot \Delta \Phi(x, y, z)
\]

\[
\leq R^2
\]

diameter

by induction:
\[
\|w^{k+1}\|^2 \leq kR^2
\]

\[
\|w^{k+1}\| \leq \sqrt{kR}
\]

combine with:
\[
\|w^{k+1}\| \geq k\delta
\]

bound on \# of updates:
\[
k \leq \frac{R^2}{\delta^2}
\]
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

violation: incorrect label scored higher

perceptron update:
\[
w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z)
\]
\[
\|w^{(k+1)}\|^2 = \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 
\]
\[
= \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2 w^{(k)} \cdot \Delta \Phi(x, y, z) 
\]
\[
\leq R^2
\]
\[
\leq 0
\]

by induction:
\[
\|w^{k+1}\|^2 \leq kR^2
\]
\[
\|w^{k+1}\| \leq \sqrt{kR}
\]

bound on \# of updates:
\[
k \leq R^2 / \delta^2
\]

combine with:
\[
\|w^{k+1}\| \geq k\delta
\]

update weights if \(y \neq z\)

\((\text{part 2: upperbound})\)

exact inference

\((\text{part 1: lowerbound})\)
Geometry of Convergence Proof pt 2

summary: the proof uses 3 facts:
1. separation (margin)
2. diameter (always finite)
3. violation (guaranteed by exact search)

violation: incorrect label scored higher

perceptron update:
\[ w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z) \]
\[ \|w^{(k+1)}\|^2 = \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 \leq R^2 \]

by induction: \[ \|w^{k+1}\|^2 \leq kR^2 \]

violation (part 2: upperbound)

bound on # of updates:
\[ k \leq \frac{R^2}{\delta^2} \]
Tutorial Outline

- Overview of Structured Learning
  - Challenges in Scalability
- Structured Perceptron
  - convergence proof
- Structured Perceptron with Inexact Search
- Latent-Variable Perceptron
- Parallelizing Online Learning (Perceptron & MIRA)
Scalability Challenge 1: Inference

binary classification

\[ x \quad y = +1 \quad y = -1 \]

---

\[ w \]

exact inference

update weights if \( y \neq z \)
Scalability Challenge 1: Inference

**Binary classification**

\[ y = \begin{cases} +1 & \text{if } x = +1 \\ -1 & \text{if } x = -1 \end{cases} \]

**Structured classification**

- The man bit the dog
- DT NN VBD DT NN

**Diagram**

- Exact inference
- Update weights if \( y \neq z \)
Scalability Challenge 1: Inference

binary classification

structured classification

the man bit the dog

DT NN VBD DT NN
Scalability Challenge 1: Inference

- **challenge**: search efficiency (exponentially many classes)
- often use dynamic programming (DP)
- but DP is still too slow for repeated use, e.g. parsing $O(n^3)$
- Q: can we sacrifice search exactness for faster learning?
Perceptron w/ Inexact Inference

```
the man bit the dog

DT NN VBD DT NN
```

```
x y

inexact inference

w

x x z

update weights if y ≠ z

greedy search

beam search
```
Perceptron w/ Inexact Inference

- routine use of inexact inference in NLP (e.g. beam search)
**Perceptron w/ Inexact Inference**

- routine use of inexact inference in NLP (e.g. beam search)
- how does structured perceptron work with inexact search?

```
the man bit the dog
```

```
DT NN VBD DT NN
```

```
x x → inexact inference → z
```

```
update weights if \( y \neq z \)
```

- greedy search
- beam search
Perceptron w/ Inexact Inference

- routine use of inexact inference in NLP (e.g. beam search)
- how does structured perceptron work with inexact search?
  - so far most structured learning theory assume exact search
Perceptron w/ Inexact Inference

- Routine use of inexact inference in NLP (e.g. beam search)
- How does structured perceptron work with inexact search?
- So far most structured learning theory assume exact search
- Would search errors break these learning properties?

Q: Does perceptron still work???

Greedy search

Beam search

the man bit the dog

DT NN VBD DT NN

x x → inexact inference

w y y

z update weights if y ≠ z

Q: does perceptron still work???
Perceptron w/ Inexact Inference

- routine use of inexact inference in NLP (e.g. beam search)
- how does structured perceptron work with inexact search?
  - so far most structured learning theory assume exact search
  - would search errors break these learning properties?
  - if so how to modify learning to accommodate inexact search?

Q: does perceptron still work???

greedy search
beam search

the man bit the dog
DT NN VBD DT NN

inexact inference

update weights if y ≠ z

w

x x y y z

inexact inference

w

update weights if y ≠ z
Bad News and Good News

the man bit the dog

DT NN VBD DT NN

x x ➞ inference ➞ z

update weights if $y \neq z$

A: it no longer works as is, but we can make it work by some magic.
Bad News and Good News

- bad news: no more guarantee of convergence

A: it no longer works as is, but we can make it work by some magic.

- greedy search
- beam search

bad news: no more guarantee of convergence
Bad News and Good News

• bad news: no more guarantee of convergence
  • in practice perceptron degrades a lot due to search errors

A: it no longer works as is, but we can make it work by some magic.
Bad News and Good News

- bad news: no more guarantee of convergence
  - in practice perceptron degrades a lot due to search errors
- good news: new update methods guarantee convergence

**A:** it no longer works as is, but we can make it work by some magic.
Bad News and Good News

- bad news: no more guarantee of convergence
  - in practice perceptron degrades a lot due to search errors
- good news: new update methods guarantee convergence
  - new perceptron variants that “live with” search errors

A: it no longer works as is, but we can make it work by some magic.
Bad News and Good News

- **bad news**: no more guarantee of convergence
  - in practice perceptron degrades a lot due to search errors
- **good news**: new update methods guarantee convergence
  - new perceptron variants that “live with” search errors
  - in practice they work really well with inexact search

A: it no longer works as is, but we can make it work by some magic.
Convergence with Exact Search

Training example:

- Time: flies
- N: V

Output space:

- \( \{N, V\} \times \{N, V\} \)

Correct label:

- NV

Current model:

- VV, V, N
- V, N, V N

Graph:

- Current model nodes:
  - VV (brown)
  - V (red)
  - N (blue)
  - V N (yellow)

- Training example node:
  - NV (blue box)

Connections:

- VV → V
- V → N
- N → V N
- V N → NV
Convergence with Exact Search

current model

training example
time  flies
N   V

output space
{N,V} x {N,V}

$w(x)$

$V V V V$

$V V V V$

$N V V V$

correct label

$N V V V$

$V V V V$

$V V V V$

$N V V V$

$V V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$

$N V V V$

$V V V V$
Convergence with Exact Search

current model

training example

time flies
N V

output space
{N,V} x {N,V}

w(x)

VV V

N

V N

NV correct label
Convergence with Exact Search

current model

training example

update

w(k)

output space

{N,V} x {N,V}

correct label

time flies
N V
Convergence with Exact Search

current model

training example

output space

\{N,V\} \times \{N,V\}

Convergence with Exact Search

current model

training example

output space

\{N,V\} \times \{N,V\}
Convergence with Exact Search

Current model

Training example

Time flies

Output space

\{N,V\} x \{N,V\}

Structured perceptron converges with exact search
No Convergence w/ Greedy Search

current model

training example

time flies

N V

output space

\{N, V\} \times \{N, V\}
No Convergence w/ Greedy Search

Current model

Training example

time  flies
N     V

Output space

\{N,V\} \times \{N,V\}
No Convergence w/ Greedy Search

current model

training example

output space

{N,V} x {N,V}
No Convergence w/ Greedy Search

current model

w^{(k)}

training example

time flies

N V

output space

\{N,V\} \times \{N,V\}

VV V

V

N

V

V V N

NV

N

V N

V

V N

N

Correct label
No Convergence w/ Greedy Search

training example

time     flies
N         V

output space

\{N,V\} \times \{N,V\}
No Convergence w/ Greedy Search

current model

training example
time flies
N V

output space
{N,V} × {N,V}

V V V
N

V V
N

V
N

V N
N

w^{(k)}

w^{(k)}

w^{(k)}

V V

NV

correct label
No Convergence w/ Greedy Search

Training example:
- Time: N
- Flies: V

Output space:
\{N, V\} x \{N, V\}

Current model:

- \( w^{(k)} \)

Correct label:
- NV

Diagram:

- Nodes represent different states:
  - \( V \): Verb
  - \( N \): Noun
- Edges represent transitions with weights:
  - \( w^{(k)} \)
- Connections show the flow of transitions from the current model to the correct label.
No Convergence w/ Greedy Search

training example

time
flies
N
V

output space
{N,V} x {N,V}

current model

w^{(k)}

correct label

update

\Delta \phi(x, y, z)
No Convergence w/ Greedy Search

current model

\[ w^{(k)} \]

new model

\[ w^{(k+1)} \]

training example

\[
\begin{array}{c}
\text{time} \\
N \quad V
\end{array}
\]

output space

\[
\{N, V\} \times \{N, V\}
\]

correct label

update

\[ \Delta \phi(x, y, z) \]
No Convergence w/ Greedy Search

structured perceptron does not converge with greedy search

training example

time     flies
         N     V
output space

{N,V} x {N,V}

current model

new model

$w^{(k)} \rightarrow VV \quad V \quad VN \rightarrow NV$ (correct label)

$w^{(k)} \rightarrow N \rightarrow N \rightarrow VN$ (current model)

$w^{(k+1)} \rightarrow VV \quad V \quad VN \rightarrow NV$ (new model)
No Convergence w/ Greedy Search

Which part of the convergence proof no longer holds?
No Convergence w/ Greedy Search

Which part of the convergence proof no longer holds?

The proof only uses 3 facts:
1. separation (margin)
2. diameter (always finite)
3. violation (guaranteed by exact search)
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

\(w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z)\)

\[ \|w^{(k+1)}\|^2 = \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 \]

\[ = \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2 \cdot w^{(k)} \cdot \Delta \Phi(x, y, z) \]

\[ \leq R^2 \]

violation: incorrect label scored higher
Geometry of Convergence Proof pt 2

1. repeat
2. for each example \((x, y)\) in \(D\) do
3. \(z \leftarrow \text{EXACT}(x, w)\)
4. if \(z \neq y\) then
5. \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6. until converged

Violation: incorrect label scored higher

**Exact Inference**

**Perceptron Update:**

\[
\begin{align*}
\|w^{(k+1)}\|^2 &= \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 \\
&= \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2 w^{(k)} \cdot \Delta \Phi(x, y, z) \\
&\leq R^2 \\
&\leq 0
\end{align*}
\]

Violation
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \(z \leftarrow \text{EXACT}(x, w)\)
4: if \(z \neq y\) then
5: \(w \leftarrow w + \Delta \Phi(x, y, z)\)
6: until converged

exact inference

\(w\)

update weights
if \(y \neq z\)

\(x\)

\(y\)

\(z\)

violation: incorrect label scored higher

perceptron update:

\[
\begin{align*}
\|w^{(k+1)}\|^2 &= \|w^{(k)} + \Delta \Phi(x, y, z)\|^2 \\
&= \|w^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2w^{(k)} \cdot \Delta \Phi(x, y, z) \\
&\leq R^2 \\
&\leq 0
\end{align*}
\]

violation
Geometry of Convergence Proof pt 2

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \[ z \leftarrow \text{EXACT}(x, w) \]
4: if \(z \neq y\) then
5: \[ w \leftarrow w + \Delta \Phi(x, y, z) \]
6: until converged

violation: incorrect label scored higher

**perceptron update:**
\[
w^{(k+1)} = w^{(k)} + \Delta \Phi(x, y, z)
\]
\[
\|w^{(k+1)}\|^2 = \|w^{(k)} + \Delta \Phi(x, y, z)\|^2
\]
\[
\leq R^2
\]

inexact search doesn’t guarantee violation!
Observation: Violation is all we need!

- exact search is **not** really required by the proof
- rather, it is only used to ensure violation!

the proof only uses 3 facts:

1. separation (margin)
2. diameter (always finite)
3. violation (but no need for exact)
Observation: Violation is all we need!

- exact search is **not** really required by the proof
- rather, it is only used to ensure violation!

The proof only uses 3 facts:

1. separation (margin)
2. diameter (always finite)
3. violation (but no need for exact)
Violation-Fixing Perceptron

- if we guarantee violation, we don’t care about exactness!
- violation is good b/c we can at least fix a mistake

**same mistake bound as before!**

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \((x, y', z)\) = FINDVIOLATION\((x, y, w)\)
4: if \(z \neq y\) then \(\triangleright (x, y', z)\) is a violation
5: \(w \leftarrow w + \Delta \Phi(x, y', z)\)
6: until converged

---

**exact inference**

- **standard perceptron**
  - update weights if \(y \neq z\)

**violation-fixing perceptron**

- find violation
  - update weights if \(y' \neq z\)
Violation-Fixing Perceptron

- if we guarantee violation, we don’t care about exactness!
- violation is good b/c we can at least fix a mistake

**same mistake bound as before!**

1: repeat
2: for each example \((x, y)\) in \(D\) do
3: \((x, y', z)\) = FINDVIOLATION\((x, y, w)\)
4: if \(z \neq y\) then \(\triangleright (x, y', z)\) is a viol
5: \(w \leftarrow w + \Delta \Phi(x, y', z)\)
6: until converged

---

standard perceptron

violations

violation-fixing perceptron

x \(\rightarrow\) exact

inference

x \(\rightarrow\) find

violation

y \(\rightarrow\) update weights

if \(y \neq z\)

y \(\rightarrow\) update weights

if \(y' \neq z\)
Violation-Fixing Perceptron

- if we guarantee violation, we don’t care about exactness!
- violation is good b/c we can at least fix a mistake

\[ w \leftarrow w + \Delta \Phi(x, y', z) \]

**same mistake bound as before!**

1. repeat
2. for each example \((x, y)\) in \(D\) do
3. \((x, y', z) = \text{FINDVIOLATION}(x, y, w)\)
4. if \(z \neq y\) then \(\triangleright (x, y', z)\) is a violation
5. \(w \leftarrow w + \Delta \Phi(x, y', z)\)
6. until converged

---

**standard perceptron**

- exact inference
- update weights if \(y \neq z\)

**violation-fixing perceptron**

- find violation
- update weights if \(y' \neq z\)
What if can’t guarantee violation
What if can’t guarantee violation

- this is why perceptron doesn’t work well w/ inexact search
  - because not every update is guaranteed to be a violation
  - thus the proof breaks; no convergence guarantee
What if can’t guarantee violation

• this is why perceptron doesn’t work well w/ inexact search
  • because not every update is guaranteed to be a violation
  • thus the proof breaks; no convergence guarantee
• example: beam or greedy search
  • the model might prefer the correct label (if exact search)
  • but the search prunes it away
  • such a non-violation update is “bad” because it doesn’t fix any mistake
• the new model still misguides the search
What if can’t guarantee violation

- this is why perceptron doesn’t work well w/ inexact search
  - because not every update is guaranteed to be a violation
  - thus the proof breaks; no convergence guarantee
- example: beam or greedy search
  - the model might prefer the correct label (if exact search)
  - but the search prunes it away
- such a non-violation update is “bad” because it doesn’t fix any mistake
  - the new model still misguides the search
- Q: how can we always guarantee violation?
Solution 1: Early update (Collins/Roark 2004)

- Current model: $w^{(k)}$
- New model: $w^{(k+1)}$

Training example:
- Time: N
- Flies: V

Output space: \{N, V\} x \{N, V\}

Standard perceptron does not converge with greedy search.

Stop and update at the first mistake.
Solution 1: Early update (Collins/Roark 2004)

- Current model: $w(k)$
- New model: $w(k+1)$

Training example:
- Time: flies
- Label: N: V

Output space: $\{N,V\} \times \{N,V\}$

Standard perceptron does not converge with greedy search.

Stop and update at the first mistake.
Solution 1: Early update (Collins/Roark 2004)

- Current model:
  - \( w^{(k)} \)
- New model:
  - \( w^{(k+1)} \)

Training example:
- Time: flies
  - N
  - V

Output space:
- \( \{N,V\} \times \{N,V\} \)

Standard perceptron does not converge with greedy search.

Stop and update at the first mistake.
**Solution 1: Early update** (Collins/Roark 2004)

**current model**

\[ w^{(k)} \]

**new model**

\[ w^{(k+1)} \]

**training example**

<table>
<thead>
<tr>
<th>time</th>
<th>flies</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>V</td>
</tr>
</tbody>
</table>

**output space**

\( \{N,V\} \times \{N,V\} \)

**standard perceptron does not converge with greedy search**

**stop and update at the first mistake**

<table>
<thead>
<tr>
<th>✔️</th>
<th>✔️</th>
<th>⋯</th>
<th>✔️</th>
<th>✗</th>
</tr>
</thead>
<tbody>
<tr>
<td>update</td>
<td>skip</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Solution 1: Early update (Collins/Roark 2004)

Training example:
- Time: flies
- Output space: \{N, V\} \times \{N, V\}

Current model:
- \( w(k) \) to \( N \) to \( V \)

New model:
- \( w(k+1) \) to \( N \) to \( V \)

Correct label:
- \( V \)

Stop and update at the first mistake:

Standard perceptron does not converge with greedy search.
Solution 1: Early update \hfill (Collins/Roark 2004)

- Current model: $w^{(k)}$
- New model: $w^{(k+1)}$

Training example:
- Time: N
- Flies: V

Output space: $\{N, V\} \times \{N, V\}$

Standard perceptron does not converge with greedy search.

Stop and update at the first mistake.
Early Update: Guarantees Violation

Training example:
- **time**
- **flies**
  - **N**
  - **V**

Output space:
- \{N, V\} x \{N, V\}

Standard update doesn't converge because it doesn't guarantee violation.
Early Update: Guarantees Violation

training example
time flies
N V
output space
\{N,V\} x \{N,V\}

standard update
doesn’t converge
b/c it doesn’t guarantee violation

\[ w^{(k)} \]

\[ w^{(k+1)} \]

\[ \Delta \phi(x, y, z) \]
Early Update: Guarantees Violation

- Training example: time flies
  - correct label: N, V

- Output space: \{N,V\} x \{N,V\}

- Standard update doesn't converge because it doesn't guarantee violation

- Table:
  - Update: \(\sqrt{\sqrt{\cdots}}\)  
  - Skip: \(\times\)
Early Update: Guarantees Violation

training example

time flies
N V

output space
\{N,V\} \times \{N,V\}

standard update doesn't converge b/c it doesn't guarantee violation

<table>
<thead>
<tr>
<th>✓</th>
<th>✓</th>
<th>⋯</th>
<th>✓</th>
<th>✗</th>
</tr>
</thead>
<tbody>
<tr>
<td>← update</td>
<td>→ skip</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Early Update: Guarantees Violation

training example
time flies
N V

output space
{N,V} x {N,V}

standard update
doesn’t converge
b/c it doesn’t
guarantee violation

early update: incorrect prefix
scores higher: a violation!
Early Update: from Greedy to Beam

• beam search is a generalization of greedy (where $b=1$)
  • at each stage we keep top $b$ hypothesis
  • widely used: tagging, parsing, translation...

• early update -- when correct label first falls off the beam
  • up to this point the incorrect prefix should score higher

• standard update (full update) -- no guarantee!
Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where \( b = 1 \))
  - at each stage we keep top \( b \) hypothesis
  - widely used: tagging, parsing, translation...
- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

![Diagram showing correct label falling off beam (pruned) and standard update (no guarantee)]
Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where $b=1$)
  - at each stage we keep top $b$ hypothesis
  - widely used: tagging, parsing, translation...
- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

[Diagram showing correct and incorrect paths, with correct label falling off the beam (pruned) and standard update (no guarantee!)]
Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where b=1)
  - at each stage we keep top b hypothesis
  - widely used: tagging, parsing, translation...
- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

![Diagram showing early update and standard update](image-url)
Early Update: from Greedy to Beam

- beam search is a generalization of greedy (where $b=1$)
  - at each stage we keep top $b$ hypothesis
  - widely used: tagging, parsing, translation...
- early update -- when correct label first falls off the beam
  - up to this point the incorrect prefix should score higher
- standard update (full update) -- no guarantee!

![Diagram showing early update and standard update](image)
Early Update as Violation-Fixing

- beam
- find violation
- update weights
- prefix violations
- standard update (bad!)
- early update
- correct label falls off beam (pruned)
Early Update as Violation-Fixing

also new definition of “beam separability”: a correct prefix should score higher than any incorrect prefix of the same length (maybe too strong)

cf. Kulesza and Pereira, 2007
Solution 2: Max-Violation (Huang et al 2012)

- we now established a theory for early update (Collins/Roark)
- but it learns too slowly due to partial updates
- max-violation: use the prefix where violation is maximum
  - “worst-mistake” in the search space
- all these update methods are violation-fixing perceptrons
Solution 2: Max-Violation  (Huang et al 2012)

- we now established a theory for early update  (Collins/Roark)
- but it learns too slowly due to partial updates
- max-violation: use the prefix where violation is maximum
  - “worst-mistake” in the search space
- all these update methods are violation-fixing perceptrons
Four Experiments

part-of-speech tagging

the man bit the dog

DT NN VBD DT NN

incremental parsing

the man bit the dog

bottom-up parsing w/ cube pruning

the man bit the dog

machine translation

the man bit the dog

那 人 咬 了 狗
Max-Violation > Early >> Standard

- exp 1 on part-of-speech tagging w/ beam search (on CTB5)
- early and max-violation >> standard update at smallest beams
  - this advantage shrinks as beam size increases
- max-violation converges faster than early (and slightly better)
Max-Violation > Early >> Standard

- exp 1 on part-of-speech tagging w/ beam search (on CTB5)
- early and max-violation >> standard update at smallest beams
- this advantage shrinks as beam size increases
- max-violation converges faster than early (and slightly better)
Max-Violation > Early >> Standard

- exp 1 on part-of-speech tagging w/ beam search (on CTB5)
- early and max-violation >> standard update at smallest beams
- this advantage shrinks as beam size increases
- max-violation converges faster than early (and slightly better)
Max-Violation > Early >> Standard

- exp 2 on my incremental dependency parser (Huang & Sagae 10)
- standard update is horrible due to search errors
- early update: 38 iterations, 15.4 hours (92.24)
- max-violation: 10 iterations, 4.6 hours (92.25)
Max-Violation > Early >> Standard

- exp 2 on my incremental dependency parser (Huang & Sagae 10)
- standard update is horrible due to search errors
- early update: 38 iterations, 15.4 hours (92.24)
- max-violation: 10 iterations, 4.6 hours (92.25)
Max-Violation > Early >> Standard

- exp 2 on my incremental dependency parser (Huang & Sagae 10)
- standard update is horrible due to search errors
- early update: 38 iterations, 15.4 hours (92.24)
- max-violation: 10 iterations, 4.6 hours (92.25)

max-violation is 3.3x faster than early update
Why standard update so bad for parsing

- standard update works horribly with severe search error
  - due to large number of invalid updates (non-violation)

![Parsing Accuracy vs Training Time](image1)

![Tagging Accuracy vs Training Time](image2)
Why standard update so bad for parsing

- standard update works horribly with severe search error
- due to large number of invalid updates (non-violation)

![Graph showing parsing and tagging accuracy over training time.](image)
Why standard update so bad for parsing

- standard update works horribly with severe search error
- due to large number of invalid updates (non-violation)

\[ O(n^{11}) \Rightarrow O(nb) \]

\[ O(nT^3) \Rightarrow O(nb) \]

- % of invalid updates in standard update
Why standard update so bad for parsing

- standard update works horribly with severe search error
- due to large number of invalid updates (non-violation)

**take-home message:**
early/max-violation more helpful for harder search problems!
Exp 3: Bottom-up Parsing

- CKY parsing with cube pruning for higher-order features
- We extended our framework from graphs to hypergraphs

(Zhang et al. 2013)
Exp 3: Bottom-up Parsing

- CKY parsing with cube pruning for higher-order features
- We extended our framework from graphs to hypergraphs

![Diagram of CKY parsing with cube pruning](image)

![Graph showing UAS on Penn-YM dev](image)

(Zhang et al 2013)
Exp 4: Machine Translation

- Standard perceptron works poorly for machine translation
  - Because invalid update ratio is very high (search quality is low)
- Max-violation converges faster than early update
- First truly successful effort in large-scale training for translation

(Yu et al 2013)
Exp 4: Machine Translation

- standard perceptron works poorly for machine translation
- b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update
- first truly successful effort in large-scale training for translation

(Yu et al 2013)
Comparison of Four Exps

- the harder your search, the more advantageous
Related Work and Discussions
Related Work and Discussions

- Our “violation-fixing” framework include as special cases:
  - Early-update (Collins and Roark, 2004)
  - LaSO (Daume and Marcu, 2005)
  - Not sure about Searn (Daume et al, 2009)
Related Work and Discussions

• our “violation-fixing” framework include as special cases
  • early-update (Collins and Roark, 2004)
  • LaSO (Daume and Marcu, 2005)
  • not sure about Searn (Daume et al, 2009)

• “beam-separability” or “greedy-separability” related to:
  • “algorithmic-separability” of (Kulesza and Pereira, 2007)
  • but these conditions are too strong to hold in practice
Related Work and Discussions

• our “violation-fixing” framework include as special cases
  • early-update (Collins and Roark, 2004)
  • LaSO (Daume and Marcu, 2005)
  • not sure about Searn (Daume et al, 2009)
• “beam-separability” or “greedy-separability” related to:
  • “algorithmic-separability” of (Kulesza and Pereira, 2007)
  • but these conditions are too strong to hold in practice
• under-generating (beam) vs. over-generating (LP-relax.)
  • Kulesza & Pereira and Martins et al (2011): LP-relaxation
  • Finley and Joachims (2008): both under and over for SVM
Conclusions So Far

- structured perceptron is simple, scalable, and powerful
  - (almost) same convergence proof from multiclass perceptron
- but it doesn’t work very well with inexact search
- solution: violation-fixing perceptron framework
  - convergence under new defs of separability
  - learn to “live with” search errors
    - in particular, “max-violation” works great
      - converges fast, and results in high accuracy
  - they are more helpful to harder search problems!
Tutorial Outline

- Overview of Structured Learning
  - Challenges in Scalability
- Structured Perceptron
  - convergence proof
- Structured Perceptron with Inexact Search
- Latent-Variable Perceptron
- Parallelizing Online Learning (Perceptron & MIRA)
Learning with Latent Variables

- aka “weakly-supervised” or “partially-observed” learning
- learning from “natural annotations”; more scalable
- examples: translation, transliteration, semantic parsing...

parallel text

(Liang et al 2006; Yu et al 2013; Xiao and Xiong 2013)
Learning with Latent Variables

• aka “weakly-supervised” or “partially-observed” learning
• learning from “natural annotations”; more scalable
• examples: translation, transliteration, semantic parsing...

parallel text
(Liang et al 2006; Yu et al 2013; Xiao and Xiong 2013)

transliteration
(Knight & Graehl, 1998; Kondrak et al 2007, etc.)
Learning with Latent Variables

- aka “weakly-supervised” or “partially-observed” learning
- learning from “natural annotations”; more scalable
- examples: translation, transliteration, semantic parsing...

parallel text
(Liang et al 2006; Yu et al 2013; Xiao and Xiong 2013)

transliteration
(Knight & Graehl, 1998; Kondrak et al 2007, etc.)

QA pairs
(Clark et al 2010; Liang et al 2013; Kwiatkowski et al 2013)
Learning Latent Structures

- **Binary/Multiclass**
  - Naive Bayes
  - Conditional

- **Structured Learning**
  - HMMs
  - Conditional

- **Latent Structures**
  - CRFs
  - Online + Viterbi

- **Perceptron**
  - Max Margin
  - SVM

- **Structured Perceptron**
  - Max Margin
  - Structured SVM
Learning Latent Structures

- **binary/multiclass**
  - naive bayes
  - Conditional

- **structured learning**
  - HMMs
  - Conditional

- **latent structures**
  - EM (forward-backward)

- **naive bayes**
  - Logistic regression (maxent)

- **perceptron**
  - Online+ Viterbi
  - Max margin

- **SVM**

- **structured learning**
  - Structured perceptron

- **structured SVM**
Learning Latent Structures

binary/multiclass

naive bayes

Conditional

logistic regression (maxent)

Online+ Viterbi

perceptron

SVM

structured learning

structured perceptron

structured SVM

structured SVM

latent structures

HMMs

Online+ Viterbi

EM

(forward-backward)

CRFs

latent CRFs

Conditional

Conditional

Conditional

latent CRFs

Conditional

conditional
Learning Latent Structures

- **binary/multiclass**
  - naive bayes
  - Conditional
  - logistic regression (maxent)
  - Online+ Viterbi
  - perceptron
  - max margin
  - SVM

- **structured learning**
  - HMMs
  - Conditional
  - CRFs
  - Online+ Viterbi
  - structured perceptron
  - max margin
  - structured SVM

- **latent structures**
  - EM (forward-backward)
  - latent CRFs
  - latent perceptron
Learning Latent Structures

binary/multiclass

naive bayes

conditional

logistic regression (maxent)

Online+ Viterbi

perceptron

max margin

SVM

structured learning

HMMs

EM

(forward-backward)

latent structures

CRFs

latent CRFs

latent perceptron

structured perceptron

max margin

structured SVM

latent structured SVM
Latent Structured Perceptron

• no explicit positive signal

• hallucinate the “correct” derivation by current weights

*training example*

![Image showing the man bit the dog](image.png)

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

training example

那 人 咬 了 狗 \( x \)

the man bit the dog \( y \)

during online learning...

那人咬了狗 \( x \)

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

<table>
<thead>
<tr>
<th>那 人 咬 了 狗</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>the man bit the dog</td>
<td>y</td>
</tr>
</tbody>
</table>

* during online learning...

<table>
<thead>
<tr>
<th>那 人 咬 了 狗</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>full search space</td>
<td></td>
</tr>
</tbody>
</table>

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

**training example**

```
那人咬了狗  
x
```

```
the man bit the dog  
y
```

**during online learning...**

```
那人咬了狗  
x
```

```
full search space  
```

```
最高-得分 derivation  
```

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

**training example**

```
那人咬了狗
```

```
the man bit the dog
```

**during online learning...**

```
那人咬了狗
```

```
the dog bit the man
```

(full search space)

highest-scoring derivation

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

\[
\text{那人咬了狗} \quad x
\]

\[
\text{the man bit the dog} \quad y \neq
\]

*during online learning...*

\[
\text{那人咬了狗} \quad x
\]

\[
\text{full search space}
\]

\[
\text{the dog bit the man} \quad \hat{d} \quad \sim
\]

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

那 人 咬 了 狗

*forced decoding space*

the man bit the dog

*x*

*y* ≠ 

*full search space*

that the dog bit the man

*d*

*highest-scoring derivation*

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

**training example**

<table>
<thead>
<tr>
<th>那人咬了狗</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>the man bit the dog</td>
<td>y ≠</td>
</tr>
</tbody>
</table>

**forced decoding space**

\(d^*\)

highest-scoring gold derivation

**during online learning...**

<table>
<thead>
<tr>
<th>那人咬了狗</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>the dog bit the man</td>
<td>(\hat{z})</td>
</tr>
</tbody>
</table>

highest-scoring derivation

(Liang et al 2006; Yu et al 2013)
Latent Structured Perceptron

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

\[ w \leftarrow w + \Phi(x, d^*) - \Phi(x, \hat{d}) \]

(reward correct, penalize wrong)

(Liang et al 2006; Yu et al 2013)
Unconstrained Search

- example: beam search phrase-based decoding

*Bushi yu Shalong juxing le huitan* → ???
Unconstrained Search

- example: beam search phrase-based decoding

Bushi yu Shalong juxing le huitan

???
Unconstrained Search

- example: beam search phrase-based decoding

Bushi yu Shalong juxing le huitan

Diagram:
- Bush
- ... meeting
- ... talks
- ... talk
Unconstrained Search

• example: beam search phrase-based decoding

Bushi yu Shalong juxing le huitan

... talks

... talk

... meeting

... talks

... talk

???

... Shalong

... Shalong

... Sharon

... Sharon
Unconstrained Search

- example: beam search phrase-based decoding

*Bushi yu Shalong juxing le huitan*

- Bush
  - ... talks
  - ... talk
- ... meeting

- ???
  - ... Shalong
  - ... Sharon
Constrained Search

- forced decoding: must produce the exact reference translation

Bushi yu Shalong juxing le huitan  Bush held talks with Sharon

- ___ ... meeting
- ___ ... talks
- ___ ... talk
- ... Shalong
- ... Sharon
Constrained Search

- forced decoding: must produce the exact reference translation

*Bushi yu Shalong juxing le huitan*  →  *Bush held talks with Sharon*

*Bush* → *... talks* → *... Sharon*

one gold derivation
Constrained Search

- forced decoding: must produce the exact reference translation

Bush yu Shalong juxing le huitan

Bush held talks with Sharon

one gold derivation

gold derivation lattice

held talks

with Sharon
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

那 人 咬 了 狗  $x$

the man bit the dog  $y$

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

training example

那 人 咬 了 狗  \( x \)

the man bit the dog \( y \)

during online learning...

那 人 咬 了 狗  \( x \)

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

那 人 咬 了 狗

* during online learning...

那 人 咬 了 狗

full search space

the man bit the dog

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

$$\text{那人咬了狗}$$

the man bit the dog

*during online learning...*

$$\text{那人咬了狗}$$

满搜索空间

highest-scoring derivation

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

\begin{itemize}
  \item \textit{training example}
  \begin{align*}
    \text{那人咬了狗} & \\
    \text{the man bit the dog} & \\
  \end{align*}
  \end{itemize}

\begin{itemize}
  \item \textit{during online learning...}
  \begin{align*}
    \text{那人咬了狗} & \\
    \text{the man bit the dog} & \\
    \text{the dog bit the man} & \\
    \text{highest-scoring derivation} & \\
  \end{align*}
  \end{itemize}

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

那 人 咬 了 狗 \( x \)

*the man bit the dog* \( y \) ≠

*during online learning...*

那 人 咬 了 狗 \( x \)

full search space

的 狗 咬 了 人 \( \hat{d} \)

the dog bit the man \( \approx \)

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

(training example)

强迫解码空间

the man bit the dog  
x

(Liang et al 2006; Yu et al 2013)

during online learning...

highest-scoring derivation

那 人 咬 了 狗

(right)

the dog bit the man  

z

full search space

hat d
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

*training example*

```
那人咬了狗
```

highest-scoring gold derivation

```
the man bit the dog
```

*forced decoding space*

```
x
```

```
y ≠
```

*during online learning...*

```
那人咬了狗
```

highest-scoring derivation

```
the dog bit the man
```

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

• no explicit positive signal
• hallucinate the “correct” derivation by current weights

\[ w \leftarrow w + \Phi(x, d^*) - \Phi(x, \hat{d}) \]

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

training example

强迫解码空间

最高得分为星号金属性 derivation

the man bit the dog

during online learning...

完全搜索空间

最高得分为星号 beam derivation

the dog bit the man

\[ \mathbf{w} \leftarrow \mathbf{w} + \Phi(x, \mathbf{d}^*) - \Phi(x, \hat{\mathbf{d}}) \]

problem: search errors

reward correct
penalize wrong

(Liang et al 2006; Yu et al 2013)
Search Errors in Decoding

- no explicit positive signal
- hallucinate the “correct” derivation by current weights

\[ w \leftarrow w + \Phi(x, d^*) - \Phi(x, \hat{d}) \]

(reward correct, penalize wrong)

(Liang et al 2006; Yu et al 2013)
Search Error: Gold Derivations Pruned

gold derivation lattice

Bush

held talks

held

talks

with Sharon

with

Sharon

real decoding beam search
Search Error: Gold Derivations Pruned

gold derivation lattice

Bush

held talks

held

talks

with Sharon

with

Sharon

real decoding beam search
Search Error: Gold Derivations Pruned

gold derivation lattice

Bush

held

talks

with Sharon

real decoding beam search
Search Error: Gold Derivations Pruned

gold derivation lattice

Bush

held talks

0 1 2 3 4 5 6

with Sharon

real decoding beam search

held talks

with Sharon

0 1 2 3 4 5 6

Bush held talks with Sharon

0 1 2 3 4 5 6

real decoding beam search
Search Error: Gold Derivations Pruned

gold derivation lattice

Bush

held talks

with Sharon

real decoding beam search

held

talks

with

Sharon
gold derivation lattice

held talks

with Sharon

Bush

held

talks

with

Sharon

real decoding beam search

Search Error: Gold Derivations Pruned
Search Error: Gold Derivations Pruned

gold derivation lattice

Bush

held talks

held

talks

with Sharon

real decoding beam search

should address search errors here!

0 1 2 3 4 5 6
Fixing Search Error 1: Early Update

standard update
(no guarantee!)
Fixing Search Error 1: Early Update

- **early update** (Collins/Roark’04) when the correct falls off beam
- up to this point the incorrect prefix should score higher
- that’s a “violation” we want to fix; proof in (Huang et al 2012)
Fixing Search Error 1: Early Update

- **early update** \((Collins/Roark'04)\) when the correct falls off beam
- up to this point the incorrect prefix should score higher
- that’s a “violation” we want to fix; proof in \((Huang et al 2012)\)

![Diagram showing model, correct sequence falls off beam, and standard update (no guarantee)]
Fixing Search Error 1: Early Update

- early update (Collins/Roark’04) when the correct falls off beam
- up to this point the incorrect prefix should score higher
- that’s a “violation” we want to fix; proof in (Huang et al 2012)
Fixing Search Error 1: Early Update

- early update (Collins/Roark’04) when the correct falls off beam
- up to this point the incorrect prefix should score higher
- that’s a “violation” we want to fix; proof in (Huang et al 2012)

![Diagram showing correct and incorrect sequences with early update and standard update](image-url)
Fixing Search Error 1: Early Update

• early update (Collins/Roark’04) when the correct falls off beam
  • up to this point the incorrect prefix should score higher
  • that’s a “violation” we want to fix; proof in (Huang et al 2012)
• standard perceptron does not guarantee violation
  • the correct sequence (pruned) might score higher at the end!
• “invalid” update b/c it reinforces the model error

![Diagram showing the correct and incorrect sequences, with early update and standard update highlighted. The diagram illustrates the violation guaranteed: incorrect prefix scores higher up to this point.](image-url)
Early Update w/ Latent Variable

- the gold-standard derivations are not annotated
- we treat any reference-producing derivation as good

gold derivation lattice

Bush held talks with Sharon

Model
the gold-standard derivations are not annotated

we treat any reference-producing derivation as good
gold derivation lattice

Bush held talks with Sharon

Model
Early Update w/ Latent Variable

- the gold-standard derivations are **not** annotated
- we treat any reference-producing derivation as good

![Diagram showing gold derivation lattice and held talks with Sharon]
Early Update w/ Latent Variable

- the gold-standard derivations are **not** annotated
- we treat any reference-producing derivation as good

Gold derivation lattice:

- Bush
- held talks
- with Sharon

Correct and all correct derivations fall off.
the gold-standard derivations are not annotated

we treat any reference-producing derivation as good

gold derivation lattice

All correct derivations fall off

Model
Early Update w/ Latent Variable

- the gold-standard derivations are **not** annotated
- we treat any reference-producing derivation as good

---

Model

---

violated guaranteed:
incorrect prefix scores
higher up to this point

all correct derivations fall off
Early Update w/ Latent Variable

- the gold-standard derivations are **not** annotated
- we treat any reference-producing derivation as good

---

- Bush held talks with Sharon
- all correct derivations fall off early update
- violation guaranteed: incorrect *prefix* scores higher *up to this point*
- stop decoding
Fixing Search Error 2: Max-Violation

- early update works but learns slowly due to partial updates
- max-violation: use the prefix where violation is maximum
  - “worst-mistake” in the search space
- now extended to handle latent-variable
Latent-Variable Perceptron

- Best in the beam
- Worst in the beam

Correct sequence

Early max-violation

Last valid update

Full (standard)

Model $w$

Standard update is invalid

Invalid update!
Roadmap of Techniques

structured perceptron
(Collins, 2002)
Roadmap of Techniques

- structured perceptron (Collins, 2002)
- latent-variable perceptron (Zettlemoyer and Collins, 2005; Sun et al., 2009)
Roadmap of Techniques

structured perceptron (Collins, 2002)

- latent-variable perceptron
  (Zettlemoyer and Collins, 2005; Sun et al., 2009)

- perceptron w/ inexact search
  (Collins & Roark, 2004; Huang et al., 2012)
Roadmap of Techniques

1. **structured perceptron** (Collins, 2002)
2. **latent-variable perceptron** (Zettlemoyer and Collins, 2005; Sun et al., 2009)
3. **latent-variable perceptron w/ inexact search** (Yu et al 2013; Zhao et al 2014)
4. **perceptron w/ inexact search** (Collins & Roark, 2004; Huang et al 2012)
Roadmap of Techniques

structured perceptron (Collins, 2002)

latent-variable perceptron (Zettlemoyer and Collins, 2005; Sun et al., 2009)

perceptron w/ inexact search (Collins & Roark, 2004; Huang et al 2012)

latent-variable perceptron w/ inexact search (Yu et al 2013; Zhao et al 2014)

MT syntactic parsing semantic parsing transliteration
Experiments: Discriminative Training for MT

- standard update (Liang et al’s “bold”) works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update
Experiments: Discriminative Training for MT

- standard update (Liang et al’s “bold”) works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update

this explains why Liang et al ’06 failed
std ~ “bold”; local ~ “local”
Experiments: Discriminative Training for MT

- standard update (Liang et al’s “bold”) works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update

this explains why Liang et al ’06 failed
std ~ “bold”; local ~ “local”
Experiments: Discriminative Training for MT

- standard update (Liang et al’s “bold”) works poorly
  - b/c invalid update ratio is very high (search quality is low)
- max-violation converges faster than early update

this explains why Liang et al ’06 failed
std ~ “bold”; local ~ “local”

![Graph showing BLEU scores and ratio of invalid updates over iterations and beam sizes.](image-url)
Open Problems in Theory

- latent-variable structured perceptron:
  - does it converge? under what conditions?

- latent-variable structured perceptron with inexact search
  - does it converge? under what conditions?

\[
y = \begin{cases} +1 & \text{if } z \neq y_{100} \\ -1 & \text{otherwise} \end{cases}
\]

\[
\text{#updates} \leq \frac{R^2}{\delta^2}
\]

Open Problems in Theory

- latent-variable structured perceptron:
  - does it converge? under what conditions?

- latent-variable structured perceptron with inexact search
  - does it converge? under what conditions?

Easy to prove but unrealistic

(Sun et al, 2009) (Collins, 2002)
Open Problems in Theory

- latent-variable structured perceptron:
  - does it converge? under what conditions?
- latent-variable structured perceptron with inexact search
  - does it converge? under what conditions?

---

Easy to prove but unrealistic

```
oracle vector
```

(Sun et al, 2009)

Ideal situation but hard to prove

```
#updates ≤ \frac{R^2}{\delta^2}
```

???
Tutorial Outline

• Overview of Structured Learning
  • Challenges in Scalability

• Structured Perceptron
  • convergence proof

• Structured Perceptron with Inexact Search

• Latent-Variable Perceptron

• Parallelizing Online Learning (Perceptron & MIRA)
Online Learning from Big Data

• online learning has linear-time guarantee

• contrast: popular methods such as SVM/CRF are superlinear

• but online learning can still be too slow if data is too big
  • even with the fastest inexact search (e.g. greedy)

• how to parallelize online learning for big data?
Online Learning from Big Data

- online learning has linear-time guarantee
  - contrast: popular methods such as SVM/CRF are superlinear
  - but online learning can still be too slow if data is too big
    - even with the fastest inexact search (e.g. greedy)
- how to parallelize online learning for big data?
Online Learning from Big Data

- online learning has linear-time guarantee
- contrast: popular methods such as SVM/CRF are superlinear
- but online learning can still be too slow if data is too big
  - even with the fastest inexact search (e.g. greedy)
- how to parallelize online learning for big data?
Aside: Perceptron => MIRA

- perceptron is simple but...
  - only “aims to” fixes one mistake (violation) on each example
  - but structured prediction may have many violations
  - may under- or over-correct on a violation
- MIRA (margin infused relaxation algorithm)
  - 1-best MIRA: corrects one violation “just enough” (Crammer 03)
  - \( k \)-best MIRA: corrects \( k \) violations at one time (McDonald et al 05)

\[
Z = \underbrace{\argmin\{w' : \forall z \in Z, w' \cdot \Delta \Phi(x, y, z) \geq \ell(y, z)\}}_{\text{exact 1-best}} \quad w^{t+1} = \begin{cases} w^t \cdot \Phi(x, z) & \text{if } w^t \cdot \Phi(x, z) \geq \ell(y, z) \\ \argmin\{w' : \forall z \in Z, w' \cdot \Delta \Phi(x, y, z) \geq \ell(y, z)\} & \text{otherwise} \end{cases}
\]
Geometry of 1-best & k-best MIRA

\[ w^{t+1} = \arg\min_{w': \forall z \in Z, w' \cdot \Delta \Phi(x, y, z) \geq \ell(y, z)} \| w' - w^t \|^2 \]

1-best MIRA

- single constraint
- \( \Rightarrow \) analytic solution

k-best MIRA

- up to \( k \) constraints
- \( \Rightarrow \) convex optimization
Can We Parallelize Online Learning?

A diagram showing the process of search and update weights with conditions:
- Search
- Update weights $w$ if $y \neq z$
- $x \rightarrow z \rightarrow y$
Can We Parallelize Online Learning?

- can we parallelize online learning?
- harder than parallelizing batch (CRF)
- losing dependency b/w examples
- each iteration faster, but accuracy lower
Can we parallelize online learning?

- Harder than parallelizing batch (CRF)
- Losing dependency b/w examples
- Each iteration faster, but accuracy lower

Method 1: Iterative Parameter Mixing (IPM) (McDonald et al 2010)

Method 1: IPM (McDonald et al 2010) only ~3-4x faster on 10+ CPUs
Can we parallelize online learning?

• harder than parallelizing batch (CRF)
• losing dependency b/w examples
• each iteration faster, but accuracy lower

• method 1: IPM (McDonald et al 2010)
  only ~3-4x faster on 10+ CPUs

• can we do (a lot) better?

Iterative Parameter Mixing (IPM) (McDonald et al 2010)
Method 2: Minibatch Parallelization

- decode a minibatch in parallel, update in serial

Iterative Parameter Mixing (IPM) (McDonald et al. 2010)

minibatch

(Zhao and Huang, 2012)
Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - “averaging effect” (cf. McDonald et al 2010)
  - easy to prove convergence (still $R^2/\delta^2$)
Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - “averaging effect” (cf. McDonald et al 2010)
- easy to prove convergence (still $R^2/\delta^2$)

update

$$w^{(k+1)} = w^{(k)} + \frac{1}{a} \sum_i \Delta \Phi(x, y, z)$$
Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - “averaging effect” (cf. McDonald et al 2010)
- easy to prove convergence (still $R^2/\delta^2$)

\[
\text{update } \quad w^{(k+1)} = w^{(k)} + \frac{1}{a} \sum_i \Delta \Phi(x, y, z)
\]

lower bound

\[
u \cdot w^{(k+1)} = u \cdot w^{(k)} + \frac{1}{a} \sum_i u \cdot \Delta \Phi(x, y, z) \geq k \delta
\]

\[
u \cdot w^{(k+1)} \geq k \delta
\]

\[
\|w^{(k+1)}\| \geq k \delta \quad \text{(by induction)}
\]
Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
- “averaging effect” (cf. McDonald et al 2010)
- easy to prove convergence (still $R^2/\delta^2$)

update $w^{(k+1)} = w^{(k)} + \frac{1}{a} \sum_i \Delta \Phi(x, y, z)$

lower bound

$$u \cdot w^{(k+1)} = u \cdot w^{(k)} + \frac{1}{a} \sum_i u \cdot \Delta \Phi(x, y, z) \geq \delta$$

$$u \cdot w^{(k+1)} \geq k\delta$$

upper bound

$$\|w^{(k+1)}\|^2 = \|w^{(k)}\|^2 + \left\| \frac{1}{a} \sum_i \Delta \Phi(x, y, z) \right\|^2 \leq R^2$$

$$+ \frac{2}{a} w^{(k)} \cdot \sum_i \Delta \Phi(x, y, z) \leq 0$$

$$\|w^{(k+1)}\|^2 \leq kR^2$$

(by induction)
Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - “averaging effect” (cf. McDonald et al 2010)
  - easy to prove convergence (still $R^2/\delta^2$)

8x constrains in each update
Why Minibatch?

- minibatch also helps in serial mode!
- perceptron: use average updates within minibatch
  - “averaging effect” (cf. McDonald et al 2010)
  - easy to prove convergence (still $R^2/\delta^2$)
- MIRA: optimization over more constraints
  - MIRA is an online approximation of SVM
  - minibatch MIRA: better approximation of SVM
    - approaches SVM at maximum batch size

8x constrains in each update
Geometry of Minibatch MIRA

\[ \text{argmin} \frac{1}{2\|w' - w\|^2} \]

s.t. \[ w' \cdot \Delta \Phi(x, y, z) \geq \ell(y, z) \]

\[ \forall (x, y, z) \in V \]
Load Balancing

• rearrange each minibatch to minimize wasted time

iterative parameter mixing
McDonald et al 2010
Load Balancing

- rearrange each minibatch to minimize wasted time

iterative parameter mixing
McDonald et al 2010

minibatch
Load Balancing

- rearrange each minibatch to minimize wasted time

iterative parameter mixing
McDonald et al 2010

minibatch

load balanced minibatch
Experiments

incremental parsing

part-of-speech tagging

phrase-based translation
Experiment 1: Parsing with MIRA

- beam search incremental parsing (Huang et al 2012) with MIRA
- minibatch learns better even in the serial setting (1 CPU)
Parallelized Minibatch Faster than IPM

- minibatch is much faster than iterative parameter mixing

Minibatch: 9x on 12 cores

McDonald et al: 3x on 12 cores
Experiment 2: Tagging (Perceptron)
Experiment 3: Machine Translation

- minibatch leads to 7x speedup on 24 cores

Graph showing BLEU scores over time for different minibatch sizes and core counts, with lines for MERT and PRO-dense.
Roadmap of Techniques

structured perceptron
(Collins, 2002)
Roadmap of Techniques

- **structured perceptron** (Collins, 2002)

- **latent-variable perceptron** (Zettlemoyer and Collins, 2005; Sun et al., 2009)
Roadmap of Techniques

- Structured perceptron (Collins, 2002)
  - Latent-variable perceptron (Zettlemoyer and Collins, 2005; Sun et al., 2009)
  - Perceptron w/ inexact search (Collins & Roark, 2004; Huang et al. 2012)
Roadmap of Techniques

structured perceptron (Collins, 2002)

- latent-variable perceptron (Zettlemoyer and Collins, 2005; Sun et al., 2009)
- minibatch parallelization (Zhao & Huang, 2013)
- perceptron w/ inexact search (Collins & Roark, 2004; Huang et al. 2012)
Roadmap of Techniques

- **structured perceptron** (Collins, 2002)
  - latent-variable perceptron (Zettlemoyer and Collins, 2005; Sun et al., 2009)
  - minibatch parallelization (Zhao & Huang, 2013)
  - perceptron w/ inexact search (Collins & Roark, 2004; Huang et al., 2012)
  - latent-variable perceptron w/ inexact search & parallelization (Yu et al., 2013; Zhao et al., 2014)

- minibatch parallelization (Zhao & Huang, 2013)
Roadmap of Techniques

- **structured perceptron** (Collins, 2002)
- **latent-variable perceptron** (Zettlemoyer and Collins, 2005; Sun et al., 2009)
- **minibatch parallelization** (Zhao & Huang, 2013)
- **perceptron w/ inexact search** (Collins & Roark, 2004; Huang et al 2012)
- **latent-variable perceptron w/ inexact search & parallelization** (Yu et al 2013; Zhao et al 2014)

MT, syntactic parsing, semantic parsing, transliteration
Roadmap of Techniques

- **structured perceptron** (Collins, 2002)
  - latent-variable perceptron
    - (Zettlemoyer and Collins, 2005; Sun et al., 2009)
  - minibatch parallelization
    - (Zhao & Huang, 2013)
  - perceptron w/ inexact search
    - (Collins & Roark, 2004; Huang et al. 2012)
  - latent-variable perceptron w/ inexact search & parallelization
    - (Yu et al. 2013; Zhao et al. 2014)

- MT
- syntactic parsing
- semantic parsing
- transliteration

(Zhao et al. 2014)
Final Conclusions

- Online structured learning is simple and powerful
- Search efficiency is the key challenge
- Search errors do interfere with learning
  - But we can use violation-fixing perceptron w/ inexact search
- We can extend perceptron to learn latent structures
- We can parallelize online learning using minibatch
Final Conclusions

- Online structured learning is simple and powerful
- Search efficiency is the key challenge
- Search errors do interfere with learning
  - But we can use violation-fixing perceptron with inexact search
- We can extend perceptron to learn latent structures
- We can parallelize online learning using minibatch
Annotated References (1)

- Binary Perceptron
  - original: Rosenblatt, 1959
  - convergence proof: Novikoff, 1962
- Multiclass Perceptron (and voted/average perceptron)
  - Freund and Schapire, 1999
- Structured Perceptron (and inexact search extensions)
  - original: Collins, 2002 (also contains generalization bounds; proofs mostly verbatim from Freund/Schapire, 1999)
  - early-update: Collins and Roark, 2004 (but no justification)
  - max-violation: Huang et al, 2012 (also defines violation-fixing perceptron framework, where early/max-violation are instances)
  - hypergraph inexact search: Zhang et al, 2013 (CKY-style parsing)
Annotated References (2)

- Latent Variable Perceptron (and inexact search extensions)
  - semantic parsing: Zettlemoyer and Collins, 2005
  - machine translation: Liang et al, 2006
  - separability condition: Sun et al, 2009
  - inexact search: Yu et al, 2013
  - hiero w/ inexact search: Zhao et al, 2014
- MIRA
  - 1-best MIRA: Crammer and Singer, 2003
  - $k$-best MIRA: McDonald et al, 2005
Annotated References (3)

• Parallelizing Online Learning
  • iterative parameter mixing: McDonald et al, 2010
  • minibatch: Zhao and Huang, 2013

• Other References
  • CRF: Lafferty et al, 2001
  • M³N (structured SVM): Taskar et al, 2003
  • LaSO: Daumé and Marcu, 2005
  • averaging trick: Daumé, 2006, Ph.D. thesis