Structured Perceptron with Inexact Search

Liang Huang     Suphan Fayong     Yang Guo
Information Sciences Institute
University of Southern California

NAACL 2012     Montréal     June 2012
Structured Perceptron (Collins 02)

binary classification

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

update weights if \( y \neq z \)
Structured Perceptron (Collins 02)

Binary classification:
- \( x \) and \( y = +1 \)
- \( x \) and \( y = -1 \)

Structured classification:
- The man bit the dog
- DT NN VBD DT NN

Update weights if \( y \neq z \)
Structured Perceptron (Collins 02)

binary classification

structured classification

the man bit the dog

DT NN VBD DT NN

x

y=+1

y=-1

update weights

if y ≠ z

x

exact
inference

y

w

x

z

update weights

if y ≠ z

y

exact
inference

w

x

y

z
• challenge: search efficiency (exponentially many classes)
• often use dynamic programming (DP)
• but still too slow for repeated use, e.g. parsing is $O(n^3)$
• and can’t use non-local features in DP
Perceptron w/ Inexact Inference

The man bit the dog

DT NN VBD DT NN

Greedy search

Beam search

Inexact inference

Update weights if \( y \neq z \)
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?
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?
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?

the man bit the dog

DT NN VBD DT NN

x x inexact inference

y y

z

update weights if \( y \neq z \)

w

does it still work???

beam search

greedy search
Prior work: Early update (Collins/Roark)

- Greedy or beam
- Early update on prefixes y', z'
Prior work: Early update (Collins/Roark)

- a partial answer: “early update” (Collins & Roark, 2004)
  - a heuristic for perceptron with greedy or beam search
  - updates on prefixes rather than full sequences
  - works much better than standard update in practice, but...
Prior work: Early update (Collins/Roark)

- a partial answer: “early update” (Collins & Roark, 2004)
  - a heuristic for perceptron with greedy or beam search
  - updates on prefixes rather than full sequences
  - works much better than standard update in practice, but...

- two major problems for early update
  - there is no theoretical justification -- why does it work?
  - it learns too slowly (due to partial examples); e.g. 40 epochs
Prior work: Early update (Collins/Roark)

• a partial answer: “early update” (Collins & Roark, 2004)
  • a heuristic for perceptron with greedy or beam search
  • updates on prefixes rather than full sequences
  • works much better than standard update in practice, but...

• two major problems for early update
  • there is no theoretical justification -- why does it work?
  • it learns too slowly (due to partial examples); e.g. 40 epochs

we’ll solve problems in a much larger framework
Our Contributions

- **theory**: a framework for perceptron w/ inexact search
  - explains early update (and others) as a special case
- **practice**: new update methods within the framework
  - converges faster and better than early update
  - real impact on state-of-the-art parsing and tagging
  - more advantageous when search error is severer
In this talk...

- Motivations: Structured Learning and Search Efficiency
- Structured Perceptron and Inexact Search
  - perceptron does not converge with inexact search
  - early update (Collins/Roark ’04) seems to help; but why?
- New Perceptron Framework for Inexact Search
  - explains early update as a special case
  - convergence theory with *arbitrarily* inexact search
  - new update methods within this framework
- Experiments
Structured Perceptron (Collins 02)

• simple generalization from binary/multiclass perceptron
• online learning: for each example \((x, y)\) in data
  • inference: find the best output \(z\) given current weight \(w\)
  • update weights when \(y \neq z\)

\[
\begin{align*}
\text{update weights} \\
\text{if } y \neq z
\end{align*}
\]

\[
\begin{align*}
\text{exact} \\
inference
\end{align*}
\]
Structured Perceptron (Collins 02)

- simple generalization from binary/multiclass perceptron
- online learning: for each example \((x, y)\) in data
  - inference: find the best output \(z\) given current weight \(w\)
  - update weights when if \(y \neq z\)

\[
\begin{align*}
x & \quad y = +1 \\
& \quad y = -1
\end{align*}
\]

```
the man bit the dog
```

```
DT NN VBD DT NN
```

```
x
```

```
y
```

```
exact inference
```

```
update weights if \(y \neq z\)
```
Structured Perceptron (Collins 02)

- simple generalization from binary/multiclass perceptron
- online learning: for each example \((x, y)\) in data
  - inference: find the best output \(z\) given current weight \(w\)
  - update weights when if \(y \neq z\)

```
the man bit the dog
DT NN VBD DT NN
```
Structured Perceptron (Collins 02)

- simple generalization from binary/multiclass perceptron
- online learning: for each example \((x, y)\) in data
  - inference: find the best output \(z\) given current weight \(w\)
  - update weights when if \(y \neq z\)

```
the man bit the dog
DT NN VBD DT NN
```

```
exact inference
```

```
trivial
```

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

```
exact inference
```

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

```
constant classes
```

```
exponential classes
```

```
hard
```

```
exact inference
```

```
trivial
```

```
exponential classes
```

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

```
exact inference
```

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

```
trivial
```

```
exponential classes
```

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

```
exact inference
```

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

```
trivial
```

```
exponential classes
```

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

```
exact inference
```

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

```
trivial
```

```
exponential classes
```

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

```
exact inference
```

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

```
trivial
```

```
exponential classes
```

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

```
exact inference
```

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

```
trivial
```

```
exponential classes
```

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

```
exact inference
```

```
update weights if \(y \neq z\)
```
Convergence with Exact Search

- linear classification: converges iff. data is separable
- structured: converges iff. data separable & search exact
  - 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 \( \# \text{ of updates} \leq \frac{R^2}{\delta^2} \)
  \( R \): diameter

Rosenblatt => Collins
1957 => 2002
Convergence with Exact Search

- linear classification: converges iff. data is separable
- structured: converges iff. data separable & search exact
  - 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 \( \# \text{ of updates} \leq \frac{R^2}{\delta^2} \)
  \( R: \) diameter

\[ \begin{align*}
  &y = +1 \\
  &y = -1
\end{align*} \]

Rosenblatt => Collins
1957 => 2002

\( z \neq y_{100} \)
Convergence with Exact Search

- Linear classification: converges iff. data is separable
- Structured: converges iff. data separable & search exact
- 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

Rosenblatt => Collins
1957 => 2002
Convergence with Exact Search

- linear classification: converges iff. data is separable
- structured: converges iff. data separable & search exact
  - 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 \( \# \text{ of updates} \leq \frac{R^2}{\delta^2} \)

\( R \): diameter

Rosenblatt \(\Rightarrow\) Collins
1957 \(\Rightarrow\) 2002
No Convergence w/ Greedy Search

current model

training example

output space

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

current model

training example

time flies

N V

output space

{N,V} x {N,V}

V V V

V N V

N N V

N V V

w(k)

w(k)
No Convergence w/ Greedy Search

current model

training example

output space

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

time     flies
N         V

w^{(k)}

w^{(k)}

V V V

V N

N

V

V N

N

N V

correct label
No Convergence w/ Greedy Search

current model

training example

output space

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

training example

time   flies
N      V

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

current model

$w^{(k)}$

$V V$  $V$
$V$  $N$
$N$

$V V$  $V$
$V$  $N$
$N$

correct label

V N
N

V N
N
No Convergence w/ Greedy Search

current model

training example

output space

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

current model

training example

output space

{N,V} x {N,V}

correct label

w^{(k)}
No Convergence w/ Greedy Search

current model

training example

time flies

output space

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

\[ w(k) \]

\[ \Delta \varphi(x, y, z) \]

\[ \text{update} \]

\[ \text{correct label} \]

\[ \text{correct label} \]
No Convergence w/ Greedy Search

training example

\[ \text{time} \quad \text{flies} \]
\[ N \quad V \]

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

current model

\[ w^{(k)} \]

new model

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

update

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

current model

\[ w^{(k)} \]

new model

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

training example

time flies

\[ N \quad V \]

output space

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

standard perceptron does not converge with greedy search
Early update (Collins/Roark 2004) to rescue

current model

new model

training example
time flies
N V

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

standard perceptron
does not converge
with greedy search

stop and update at the first mistake
Early update (Collins/Roark 2004) to rescue

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>
Early update (Collins/Roark 2004) to rescue

current model

\[ w^{(k)} \]

new model

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

V V V
N
V N

training example
time flies
N V

output space
\{N,V\} x \{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>
Early update (Collins/Roark 2004) to rescue

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.
Early update \((\text{Collins/Roark 2004})\) to rescue

- **Training example:**
  - Time: flies
  - Output space: \(\{\text{N, V}\} \times \{\text{N, V}\}\)

- Standard perceptron does not converge with greedy search.

- Stop and update at the first mistake.
Early update \((\text{Collins/Roark 2004})\) to rescue

- 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.
Why?

- why does inexact search break convergence property?
- what is required for convergence? exactness?
- why does early update (Collins/Roark 04) work?
  - it works well in practice and is now a standard method
  - but there has been no theoretical justification
- we answer these Qs by inspecting the convergence proof
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
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

\(w\)

\(x\)

\(y\)

\(z\)

exact inference

update weights if \(y \neq z\)

\(y\) correct label

\(z\) 1-best

\(w\) exact inference

current model \(w(k)\)
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

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

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

perceptron update:

exact inference

update weights
if \(y \neq z\)

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

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

update

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

correct label

l-best
Geometry of Convergence Proof pt 1

1: repeat
2: for each example (x, y) in D do
3:   z ← EXACT(x, w)
4:   if z ≠ y then
5:     w ← w + ΔΦ(x, y, z)
6: until converged

exact inference

update weights if y ≠ z

perceptron update:

w^(k+1) = w^(k) + ΔΦ(x, y, z)

u · w^(k+1) = u · w^(k) + u · ΔΦ(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
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

exact inference

update weights if \(y \neq z\)

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

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

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

**Diagram**

- Input: \(x\)
- Correct label: \(y\)
- Inference: \(z\)
- Update weights if \(y \neq z\)
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

\[ y \leftarrow \text{correct label} \]

exact inference

update weights if \(y \neq z\)

\[ x \rightarrow w \rightarrow z \rightarrow y \]

current model \(w(k)\)

1-best
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

update weights if \(y \neq z\)

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

update weights

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

\(y\) \rightarrow \text{correct label}

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

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

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

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

**exact inference**

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

**perceptron update:**

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

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

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

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

exact inference

update weights if \(y \neq z\)

perceptron update:

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

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

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

\(\leq R^2\) 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

exact inference

update weights if \(y \neq z\)

perceptron update:

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

\(R: \text{max diameter}\)
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

update weights if \(y \neq z\)

violation: incorrect label scored higher

perceptron update:
\[
\begin{align*}
\mathbf{w}^{(k+1)} &= \mathbf{w}^{(k)} + \Delta \Phi(x, y, z) \\
\|\mathbf{w}^{(k+1)}\|^2 &= \|\mathbf{w}^{(k)} + \Delta \Phi(x, y, z)\|^2 \\
&= \|\mathbf{w}^{(k)}\|^2 + \|\Delta \Phi(x, y, z)\|^2 + 2 \mathbf{w}^{(k)} \cdot \Delta \Phi(x, y, z) \\
&\leq R^2 \quad \text{diameter} \\
&\leq 0 \quad \text{violation}
\end{align*}
\]

by induction: \(\|\mathbf{w}^{(k+1)}\|^2 \leq kR^2\) (part 2: upperbound)

R: max diameter

exact 1-best

update

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

correct label

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

violation
by induction: \(\|w^{(k+1)}\|^2 \leq kR^2\) (part 2: upperbound)

parts 1+2 => update bounds:
\[
k \leq \frac{R^2}{\delta^2}
\]
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 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

all violations

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

violations-fixing perceptron
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

**standard perceptron**

```
1: repeat
2: for each example (x, y) in D do
3:   (x, y', z) = FINDVIOLATION(x, y, w)
4:   if z ≠ y then ▷ (x, y', z) is a violation
5:     w ← w + ΔΦ(x, y', z)
6: until converged
```

**violation-fixing perceptron**

```
x → w

exact inference

update weights if y ≠ z

y
```

```
x → w

find violation

update weights if y' ≠ z

y' → y
```
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: \((x, y', z) = \text{FINDVIOLATION}(x, y, w)\)
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
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
Standard Update: No Guarantee

training example
time flies
N V

output space
\{N,V\} x \{N,V\}
Standard Update: No Guarantee

training example

time     flies
N         V

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

NN V V
V N

correct label

w\((k)\)

w\((k)\)
Standard Update: No Guarantee

Training example:
- Time: N
- Flies: V

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

Correct label: NV
Standard Update: No Guarantee

training example

\begin{aligned}
\text{time} & \quad \text{flies} \\
N & \quad V
\end{aligned}

output space

\{N,V\} \times \{N,V\}
Standard Update: No Guarantee

training example

\[
\text{time} \quad \text{flies}
\]

\[
\begin{align*}
N & \quad V \\
\end{align*}
\]

output space

\[
\{N,V\} \times \{N,V\}
\]
Standard Update: No Guarantee

correct label scores higher: non-violation: bad update!

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
Early Update: Guarantees Violation

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

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

- Update rule:
  - \( w^{(k+1)} = w^{(k)} + \Delta \varphi(x, y, z) \)

- Correct label:
  - Sample input: \( w^{(k)} \)
  - Output class: \( N \)
  - Correct label: \( V \)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>...</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>√</td>
<td>√</td>
<td></td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td>← update</td>
<td>→</td>
<td>skip</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

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>⋮</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>√</td>
<td>√</td>
<td></td>
<td>√</td>
<td></td>
</tr>
</tbody>
</table>

← update → skip →
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

<table>
<thead>
<tr>
<th>√</th>
<th>√</th>
<th>⋯</th>
<th>√</th>
<th>x</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
  - Label: N \rightarrow V

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

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

- Early update: incorrect prefix scores higher: a violation!

- Table:
  | ✓ | ✓ | ⋯ | ✓ | × |
  |---------------|---------|
  | ← update     | → skip  |
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!

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!
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 path with early update and standard update examples]
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!

Violation guaranteed: incorrect prefix scores higher up to this point
Early Update as Violation-Fixing

Find violation

update weights if $y' \neq z$

prefix violations

beam

early update

correct label falls off beam (pruned)

standard update (bad!)
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
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
Experiments

trigram part-of-speech tagging

the man bit the dog

DT NN VBD DT NN

local features only, exact search tractable (proof of concept)

incremental dependency parsing

the man bit the dog

x

y

non-local features, exact search intractable (real impact)
1) Trigram Part of Speech Tagging

- standard update performs terribly with greedy search ($b=1$)
- because search error is severe at $b=1$: half updates are bad!
- no real difference beyond $b=2$: search error becomes rare

% of bad (non-violation) standard updates: 53% 10% 1.5% 0.5%
Max-Violation Reduces Training Time

- max-violation peaks at $b=2$, greatly reduced training time
- early update achieves the highest dev/test accuracy
  - higher than the best published accuracy (Shen et al ‘07)
- future work: add non-local features to tagging

<table>
<thead>
<tr>
<th>beam</th>
<th>iter</th>
<th>time</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>standard</td>
<td>-</td>
<td>6</td>
<td>162m</td>
</tr>
<tr>
<td>early</td>
<td>4</td>
<td>6</td>
<td>37m</td>
</tr>
<tr>
<td>max-violation</td>
<td>2</td>
<td>3</td>
<td>26m</td>
</tr>
</tbody>
</table>

Shen et al (2007) | 97.33
2) Incremental Dependency Parsing

- DP incremental dependency parser (Huang and Sagae 2010)
- non-local history-based features rule out exact DP
- we use beam search, and search error is severe
- baseline: early update. extremely slow: 38 iterations
Max-violation converges much faster

- early update: 38 iterations, 15.4 hours (92.24)
- max-violation: 10 iterations, 4.6 hours (92.25)
  12 iterations, 5.5 hours (92.32)
Comparison b/w tagging & parsing

- search error is much more severe in parsing than in tagging
- standard update is OK in tagging except greedy search (b=1)
- but performs horribly in parsing even at large beam (b=8)
- because ~50% of standard updates are bad (non-violation)!

<table>
<thead>
<tr>
<th></th>
<th>test</th>
<th>standard</th>
<th>early</th>
<th>max-violation</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of bad</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>standard updates</td>
<td>79.1</td>
<td>92.1</td>
<td>92.1</td>
<td>92.2</td>
</tr>
</tbody>
</table>
Comparison b/w tagging & parsing

- search error is much more severe in parsing than in tagging
- standard update is OK in tagging except greedy search \((b=1)\)
- but performs **horribly** in parsing even at large beam \((b=8)\)
  - because \(\sim 50\%\) of standard updates are bad (non-violation)!

<table>
<thead>
<tr>
<th></th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>standard</td>
<td>79.1</td>
</tr>
<tr>
<td>early</td>
<td>92.1</td>
</tr>
<tr>
<td>max-violation</td>
<td>92.2</td>
</tr>
</tbody>
</table>

**take-home message:**
our methods are more helpful for harder search problems!
Related Work and Discussions
Related Work and Discussions

- our “violation-fixing” framework include as special cases
  - early-update (Collins and Roark, 2004)
  - a variant of 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)
  - a variant of 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)
  • a variant of 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

• Structured Learning with Inexact Search is Important

• Two contributions from this work:
  
  • **theory**: a general violation-fixing perceptron framework
    
    • convergence for inexact search under new defs of *separability*
    
    • subsumes previous work (early update & LaSO) as special cases
  
  • **practice**: new update methods within this framework
    
    • “max-violation” learns faster and better than early update
      
      • dramatically reducing training time by 3-5 folds
      
      • improves over state-of-the-art tagging and parsing systems
    
    • our methods are more helpful to harder search problems! :)
Thank you!

Liang apologizes for not able to come due to visa/passport reasons.

Many thanks to Philipp Koehn for presenting it! Danke!

lhuang@isi.edu