On “Structured Perceptron with Inexact Search”, *NAACL 2012*

John Hewitt

CIS 700-006 : Structured Prediction for NLP

2017-09-23

All graphs from Huang, Fayong, and Guo (2012) unless otherwise specified. All other figures are original to this lecture.

Setup: tagging and perceptrons

x: The pizza runs

Setup: tagging and perceptrons

x: The pizza runs

y: *Determiner, Noun, Verb* : [D,V,N] (part of speech (POS) tagging)

Setup: tagging and perceptrons

x: The pizza runs

y: *Determiner, Noun, Verb* : [D,V,N] (part of speech (POS) tagging)

$\Phi(\mathbf{x}, [N,V,N])$: a featurization of the input sequence and output tags

Ex. previous word, previous POS, 2-character suffix

Setup: tagging and perceptrons

x: The pizza runs

y: *Determiner, Noun, Verb* : [D,V,N] (part of speech (POS) tagging)

Φ(**x**, [N,V,N]) : a featurization of the input sequence and output tags

Ex. previous word, previous POS, 2-character suffix

w*Φ(**x**, [N,V,N]) : score given to the output sequence [N,V,N] by the model for input **x.**

Setup: tagging and perceptrons

x: The pizza runs

y: *Determiner, Noun, Verb* : [D,V,N] (part of speech (POS) tagging)

Φ(**x**, [N,V,N]) : a featurization of the input sequence and output tags
Ex. previous word, previous POS, 2-character suffix

w*Φ(**x**, [N,V,N]) : score given to the output sequence [N,V,N] by the model for input **x.**

Perceptron update rule *(informal)*: when you detect that the model makes a mistake, update the weight vector, **w**.

Perceptron learning algorithm

1. Initialize weight vector $\mathbf{w} = \mathbf{0}$.
2. For each of i iterations (i in 1,4, 8, 16, etc.):
    a. For each $(\mathbf{x},\mathbf{y}) \in$ observation set $\mathbf{D}$

Perceptron learning algorithm

1. Initialize weight vector **w = 0**.
2. For each of i iterations (maximum may be 1, 4, 8, 16, etc.):
   a. For each (**x**,**y**) ∈ observation set **D**
      i. Compute the most likely label sequence according to the model:

$$\mathbf{y}^* = \arg\max_y \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})$$

Perceptron learning algorithm

1. Initialize weight vector **w = 0**.
2. For each of i iterations (maximum may be 1, 4, 8, 16, etc.):
    a. For each (**x**,**y**) ∈ observation set **D**
        i. Compute the most likely label sequence according to the model:

        $$\mathbf{y}^* = \arg\max_y \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})$$

        ii. If **y*** != **y**, this is considered a ***violation.*** Update the weight vector (to be described below.)

Perceptron learning algorithm

1. Initialize weight vector **w = 0**.
2. For each of i iterations (maximum may be 1, 4, 8, 16, etc.):
   a. For each (**x**,**y**) ∈ observation set **D**
      i. Compute the most likely label sequence according to the model:

$$\mathbf{y}^* = \arg\max_y \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})$$

      ii. If **y*** != **y**, this is considered a ***violation.*** Update the weight vector (to be described below.)

Foreshadowing: *the argmax isn't always tractable. Do we actually need an argmax, or do we just need to find* ***some*** *incorrect label sequence that's ranked more highly than the correct sequence (a violation)?*

A correct violation detection

Scoring according to the model

x: The pizza runs

low	**high**

A correct violation detection

x: The pizza runs

Scoring according to the model

low	$w^*\Phi(\mathbf{x}, [N,V,N])$	$w^*\Phi(\mathbf{x}, [D,N,V])$	**high**

Correct sequence

A correct violation detection

x: The pizza runs

Scoring according to the model

low	$w^*\Phi(\mathbf{x}, [N,V,N])$	$w^*\Phi(\mathbf{x}, [D,N,V])$	$w^*\Phi(\mathbf{x}, [N,N,N])$	**high**

Correct sequence

Violation

Update weight vector: $\mathbf{w} \leftarrow \mathbf{w} + \Phi(x, [D,N,V]) - \Phi(\mathbf{x}, [N,N,N])$

The problem is in inference

$$\arg\max_y \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})$$

low **w*Φ(x,** [N,V,N]**)** **w*Φ(x,** [D,N,V]**)** **w*Φ(x,** [N,N,N]**)** **high**

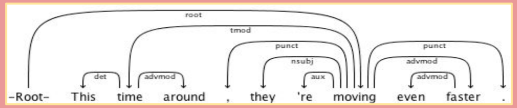
The difficulty of solving this argmax (or *argtop*$_k$) depends heavily on the problem.
One consideration is the size of the space **Y(x)**.
For some problems with only local features, **Y(x)** can be enumerated in its entirety.

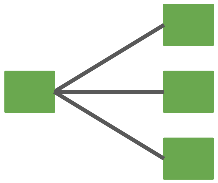
The problem is in inference

$$\arg\max_y \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}, \boldsymbol{y})$$

$\rightarrow$ **low** w*Φ(**x,** [N,V,N]) w*Φ(**x,** [D,N,V]) w*Φ(**x,** [N,N,N]) **high**

The difficulty of solving this argmax (or *argtop*$_k$) depends heavily on the problem.
One consideration is the size of the space **Y(x)**.
For some problems with only local features,**Y(x)** can be enumerated in its entirety.

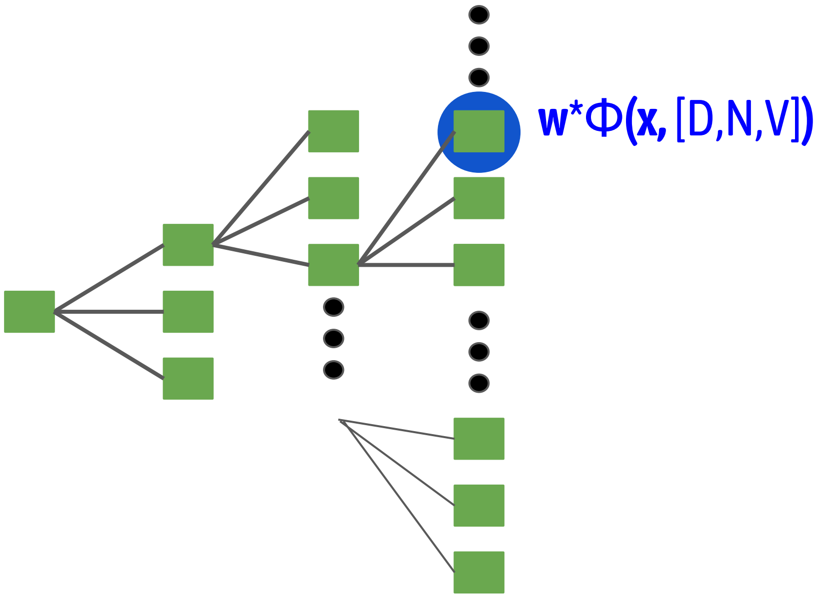
Trigram POS Tagging		Incremental Parsing
45 tags Bigram condition → search space is 45^3 = 91125	$<$	
Exact Inference Possible	(difficulty)	**Not so much**



Exact Search

Finds true argmax

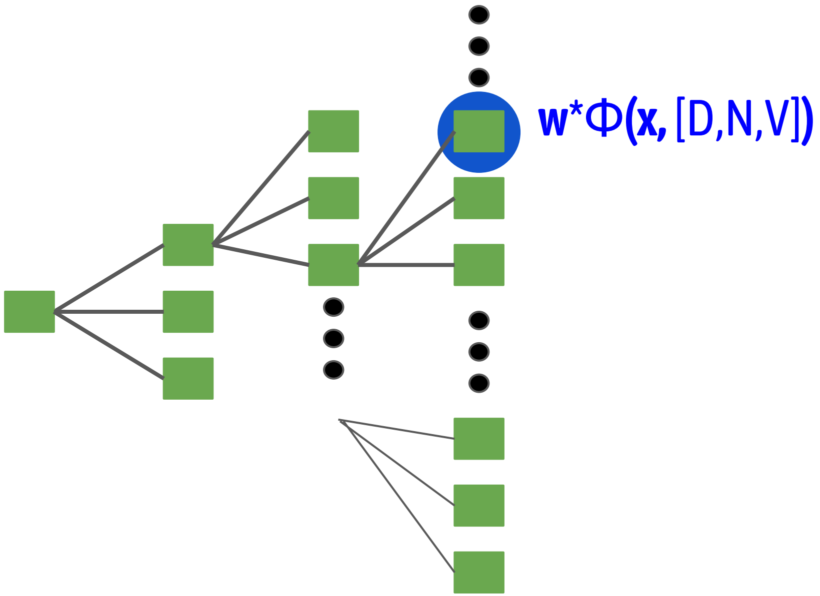
: subsequence up to this point will be scored by the model



Exact Search

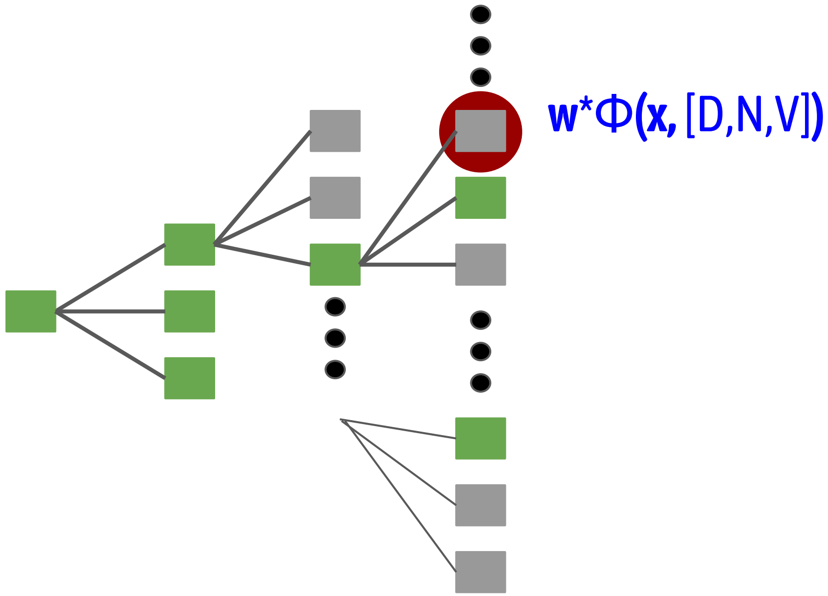
Finds true argmax

: subsequence up to this point will be scored by the model



Exact Search

Finds true argmax



Beam Search (beam = 3)

May prune correct hypothesis due to low scores early in search

: subsequence up to this point will be scored by the model

An ***incorrect*** violation detection

x: The pizza runs

Scoring according to the model

low $w^{*}\Phi(\mathbf{x}, [N,V,N])$ $w^{*}\Phi(\mathbf{x}, [D,N,V])$ **high**

Not a Violation

Correct sequence (missing due to search error)

An ***incorrect*** violation detection

Scoring according to the model

x: The pizza runs

low $w^*\Phi(\mathbf{x}, [N,V,N])$ $w^*\Phi(\mathbf{x}, [D,N,V])$ **high**

Not a Violation

Correct sequence (missing due to search error)

We still update the weight vector, *even though it was correct*:

$$\mathbf{w} \leftarrow \mathbf{w} + \Phi(x, [D,N,V]) - \Phi(\mathbf{x}, [N,N,N])$$

Perceptron Convergence

Perceptrons are proven to converge in a finite number of updates as long as they are given valid violations with which to update weight vectors. If one cannot guarantee a proposed violation is valid, convergence *may be* lost*.

Perceptron Convergence

Perceptrons are proven to converge in a finite number of updates as long as they are given valid violations with which to update weight vectors. If one cannot guarantee a proposed violation is valid, convergence *may be* lost*.

*In this paper, it is assumed that convergence is lost without valid violations found during inference, but there are relaxations of this requirement not discussed in the paper.

Notation for structured perceptron proofs

Notation 1:

Let $\mathbf{x}$ *be an input sequence,* $\mathbf{y}$ *be the correct output sequence, and* $\mathbf{z}$ *be some incorrect output sequence. Then we denote the difference between the featurization of the correct hypothesis and that of* $\mathbf{z}$ *to be the difference:*

$$\Delta\Phi(x,y,z) = \Phi(x,y) - \Phi(x,z)$$

Notation for structured perceptron proofs

Notation 1:

*Let **x** be an input sequence, **y** be the correct output sequence, and **z** be some incorrect output sequence. Then we denote the difference between the featurization of the correct hypothesis and that of **z** to be the difference:*

$$\Delta\Phi(x,y,z) = \Phi(x,y) - \Phi(x,z)$$

Definition 1: Confusion Set

*Let **D** be a dataset. Let **x** be an input sequence and Y(**x**) the set of possible output sequences for **x.** Let **y** be the correct output sequence, and **z** be some incorrect output sequence. Then the confusion set of **D** is the set of triples of **x, y**, and any sequence **z.***

$$C(D) := \{(x, y, z) | (x, y) \in D, z \in Y(x) - \{y\}\}.$$

Notation for structured perceptron proofs

Definition 2:

*A dataset **D** is said to be **linearly separable** with margin δ if there exists an oracle vector **u**, **||u|| = 1,** such that for all **x**, **y**, and **z** in **D**, the weight vector **u** scores the sequence **y** at least δ better than **z**.*

$$\boldsymbol{u} \cdot \Delta\Phi(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \geq \delta$$

Notation for structured perceptron proofs

Definition 2:

*A dataset **D** is said to be **linearly separable** with margin δ if there exists an oracle vector **u**, **||u|| = 1,** such that for all **x**, **y**, and **z** in **D,** the weight vector **u** scores the sequence **y** at least δ better than **z**.*

$$\mathbf{u} \cdot \Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z}) \geq \delta$$

Definition 3:

The **diameter**, denoted **R,** of dataset **D** is the largest norm of the vector difference between the featurization of any pair (**x**,**y**) and (**x**,**z**).

$$R = \max_{(x,y,z)} ||\Delta\Phi(x, y, z)||$$

Theorem 1: ***Structured Perceptron Convergence***

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \leq R^2/\delta^2$$

Theorem 1: *Structured Perceptron Convergence*

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \leq R^2/\delta^2$$

Proof:

We bound the norm of w, $|w_k|$, from two directions. First:

*Recall that the weight update when a violation **z** is found is*

$$\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$$

Theorem 1: *Structured Perceptron Convergence*

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \leq R^2/\delta^2$$

Proof:

We bound the norm of w, $|w_k|$, from two directions. First:

*Recall that the weight update when a violation **z** is found is*

$$\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$$

*Let **u** be an oracle weight vector that achieves δ separation. Then dot both sides of this equation by **u**.*

$$\begin{aligned} \mathbf{u}\cdot\mathbf{w}_{i+1} &= \mathbf{u}\cdot\mathbf{w}_i + \mathbf{u}\cdot\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z}) \\ &\geq \mathbf{u}\cdot\mathbf{w}_i + \delta \end{aligned}$$

Theorem 1: *Structured Perceptron Convergence*

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \leq R^2/\delta^2$$

Proof:

We bound the norm of w, $|w_k|$, from two directions. First:

*Recall that the weight update when a violation **z** is found is*

$$\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$$

*Let **u** be an oracle weight vector that achieves δ separation. Then dot both sides of this equation by **u**.*

$$\begin{aligned}\mathbf{u}\cdot\mathbf{w}_{i+1} &= \mathbf{u}\cdot\mathbf{w}_i + \mathbf{u}\cdot\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z}) \\ &\geq \mathbf{u}\cdot\mathbf{w}_i + \delta\end{aligned}$$

*By induction, we have $\mathbf{u}\cdot\mathbf{w}_{i+1} \geq k\delta$. Further, we recall that $|\mathbf{a}||\mathbf{b}| \geq \mathbf{a}\cdot\mathbf{b}$, for any **a**,**b**.*

$$|\mathbf{u}||\mathbf{w}_{i+1}| \geq \mathbf{u}\cdot\mathbf{w}_{i+1} \geq k\delta$$

$|\mathbf{w}_{i+1}| \geq k\delta$ *(since **u** is a unit vector.)*

Theorem 1: *Structured Perceptron Convergence*

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \leq R^2/\delta^2$$

Proof:

Now we bound |w| from below.

Recall that $|\mathbf{a}+\mathbf{b}|^2 = |\mathbf{a}|^2 + |\mathbf{b}|^2 + 2\mathbf{a}\cdot\mathbf{b}$

Theorem 1: *Structured Perceptron Convergence*

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \leq R^2/\delta^2$$

Proof:

Now we bound |w| from below.

Recall that $|\mathbf{a}+\mathbf{b}|^2 = |\mathbf{a}|^2 + |\mathbf{b}|^2 + 2\mathbf{a}\cdot\mathbf{b}$

Thus,

$$|\mathbf{w}_{i+1}|^2 = |\mathbf{w}_i + \Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})|^2$$
$$= |\mathbf{w}_i|^2 + |\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})|^2 + 2\mathbf{w}_i\cdot\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$$

Theorem 1: *Structured Perceptron Convergence*

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \le R^2/\delta^2$$

Proof:

Now we bound |w| from below.

Recall that $|\mathbf{a}+\mathbf{b}|^2 = |\mathbf{a}|^2 + |\mathbf{b}|^2 + 2\mathbf{a}\cdot\mathbf{b}$

Thus,

$$|\mathbf{w}_{i+1}|^2 = |\mathbf{w}_i + \Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})|^2$$

$$= |\mathbf{w}_i|^2 + |\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})|^2 + 2\mathbf{w}_i\cdot\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$$

$$\le |\mathbf{w}_i|^2 + \underbrace{R^2}_{\text{Replacing each term with the maximum (diameter.)}} + \underbrace{0}_{\text{By the definition of a valid violation, } w^*\Phi(\mathbf{x}, \mathbf{y}) < \mathbf{w}^*\Phi(\mathbf{x}, \mathbf{z}).} \rightarrow \textit{induction gives } |\mathbf{w}_i| \le kR^2$$

Theorem 1: ***Structured Perceptron Convergence***

*For a dataset **D** separable under Φ with margin δ and diameter R, the perceptron with exact search is guaranteed to converge after k updates, where*

$$k \leq R^2/\delta^2$$

Proof:

Now we bound |w| from below.

Recall that $|\mathbf{a}+\mathbf{b}|^2 = |\mathbf{a}|^2 + |\mathbf{b}|^2 + 2\mathbf{a}\cdot\mathbf{b}$

Thus,

$$|\mathbf{w}_{i+1}|^2 = |\mathbf{w}_i + \Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})|^2$$

$$= |\mathbf{w}_i|^2 + |\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})|^2 + 2\mathbf{w}_i\cdot\Delta\Phi(\mathbf{x}, \mathbf{y}, \mathbf{z})$$

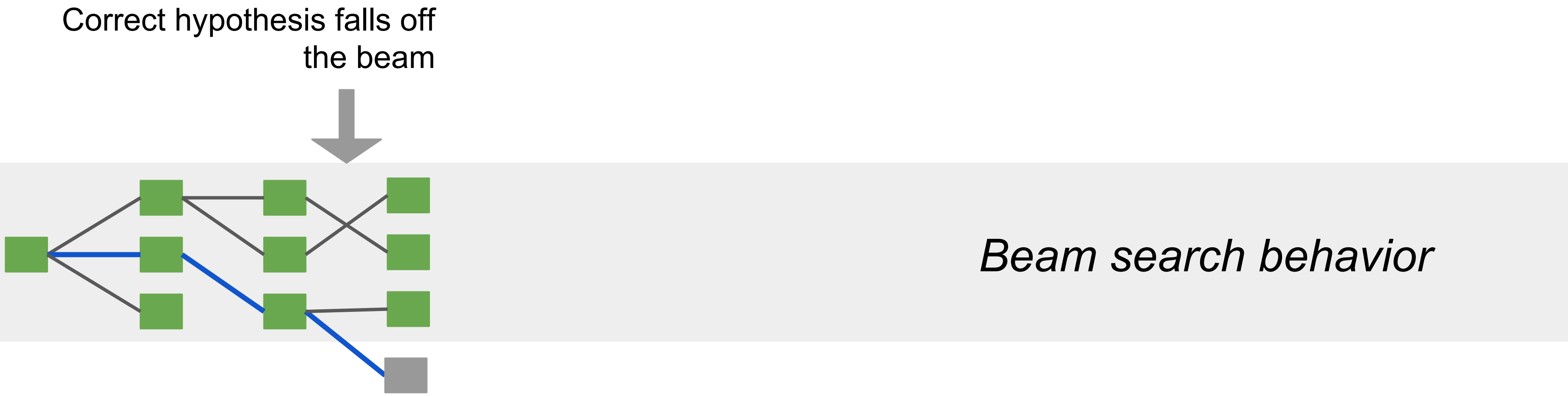
$$\leq |\mathbf{w}_i|^2 + \underbrace{R^2}_{\text{Replacing each term with the maximum (diameter.)}} + \underbrace{\mathbf{0}}_{\text{By the definition of a violation, } w^*\Phi(\mathbf{x}, \mathbf{y}) < \mathbf{w}^*\Phi(\mathbf{x}, \mathbf{z}).} \rightarrow \text{induction gives } |\mathbf{w}_i| \leq kR^2$$

Now, we combine our two bounds to get $k^2\delta^2 \leq |w_{i+1}| \leq kR^2$*, which gives us*

$$k \leq R^2/\delta^2$$

Eliminating Invalid Updates

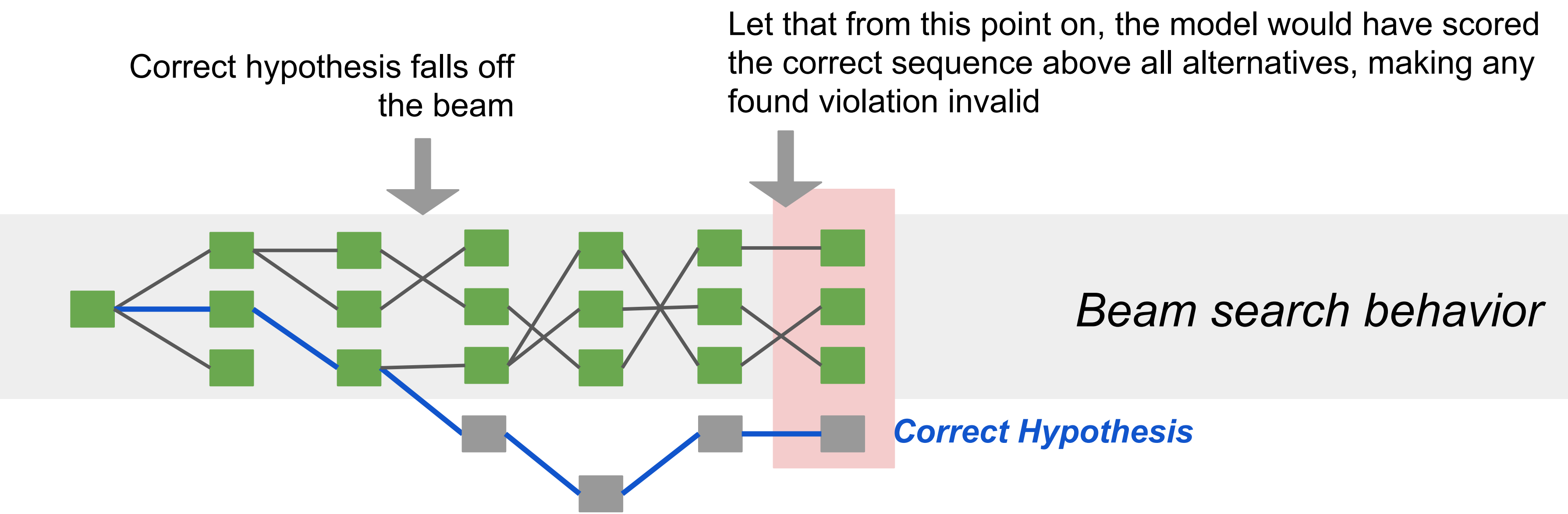
We want ways to ensure the validity of violations without requiring exact search.



Beam search behavior

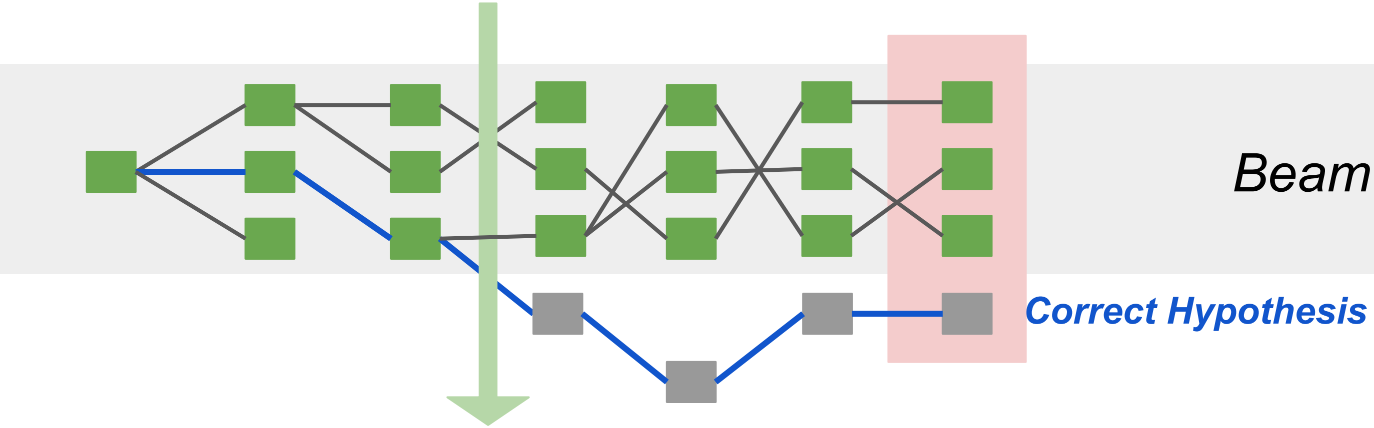
Eliminating Invalid Updates

We want ways to ensure the validity of violations without requiring exact search.



Eliminating Invalid Updates: Early Update

Correct hypothesis falls off the beam. ***Early Update Here!***



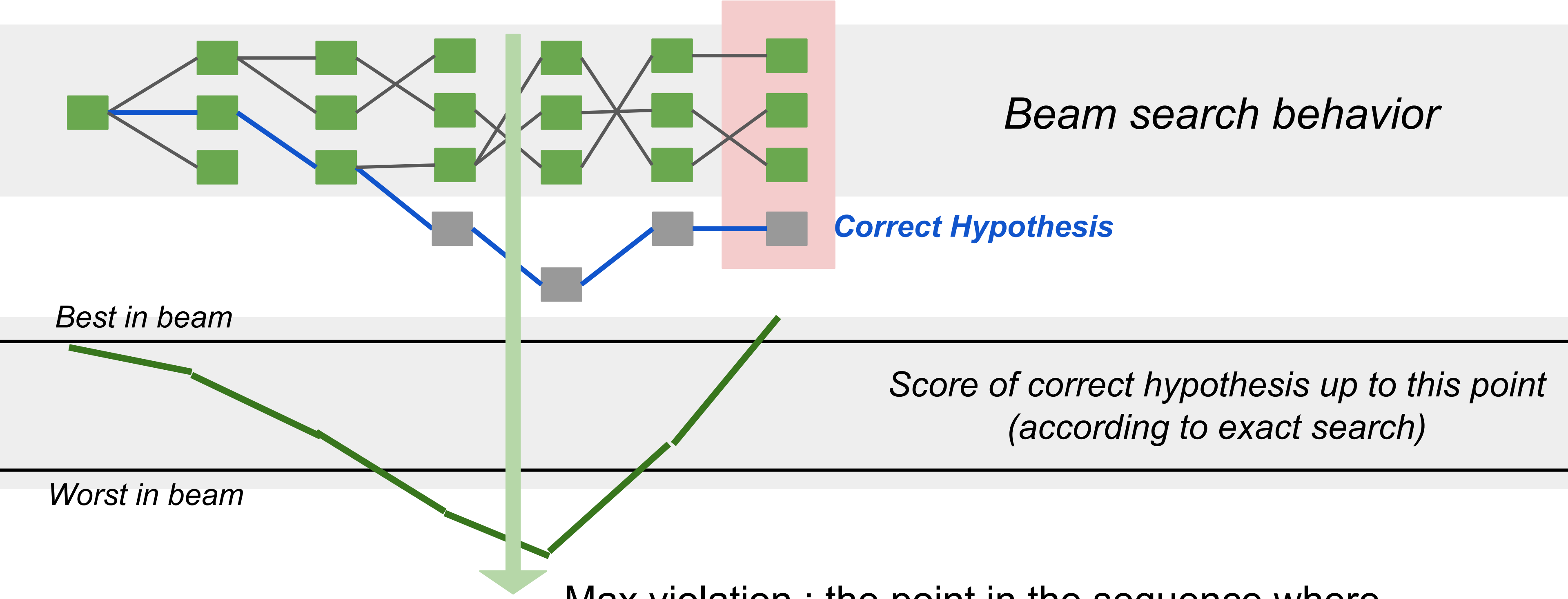
Beam search behavior

When the correct hypothesis falls off the beam, we have $\mathbf{x}_{[1:i]}$, $\mathbf{y}_{[1:i]}$, $\mathbf{z}_{[1:i]}$ such that

$$w^*\Delta\Phi(x_{[1:i]}, y_{[1:i]}, z_{[1:i]}) < 0.$$

That is, the model has clearly made an error, as there are enough subsequences $\mathbf{z}_{[1:i]}$ scored higher than $\mathbf{y}_{[1:i]}$ as to push it off the beam.

Eliminating Invalid Updates: Max Violation



Max violation : the point in the sequence where the model is the most errorful. **Update here!**

Recap

Under exact search, a valid violation, if it exists, is always guaranteed to be found and used to update **w**.

Recap

Under exact search, a valid violation, if it exists, is always guaranteed to be found and used to update **w**.

However, *any violation prefix sequence will do*, because all satisfy $\boldsymbol{w}^* \Delta \Phi(\boldsymbol{x}_{[1:i]}, \boldsymbol{y}_{[1:i]}, \boldsymbol{z}_{[1:i]}) < 0$, which is the crucial condition in perceptron convergence.

Recap

Under exact search, a valid violation, if it exists, is always guaranteed to be found and used to update **w**.

However, *any violation prefix sequence will do*, because all satisfy $\boldsymbol{w}^* \Delta \Phi(\boldsymbol{x}_{[1:i]}, \boldsymbol{y}_{[1:i]}, \boldsymbol{z}_{[1:i]}) < 0$, which is the crucial condition in perceptron convergence.

One way to find such a prefix is to take the prefix of some hypothesis on the beam as soon as the correct hypothesis falls off. This is called **early update.**

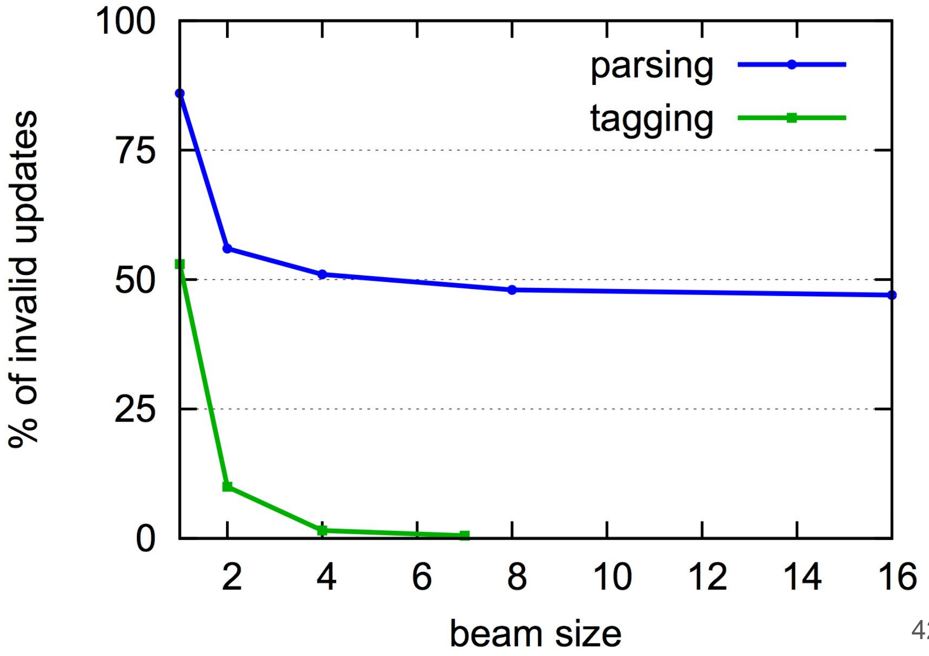
Another way to find such a prefix is to keep scoring the correct hypothesis after it falls off the beam, and take some candidate prefix on the beam at the prefix length at which the correct hypothesis is scored worst. This is called **max violation.**

Results

“I don’t believe any of this for a second.” I want to train my perceptron using full sequences **x**, **y**, and **z**, and ignore potentially invalid updates.

Turns out, this isn’t a huge problem for trigram POS tagging.

For parsing, however, the story is much worse.



Results

method	b	it	time	dev	test
early*		38	15.4 h	92.24	92.09
standard		1	0.4 h	*78.99*	*79.14*
hybrid	8	11	5.6 h	92.26	91.95
latest		9	4.5 h	92.06	91.96
max-viol.		12	5.5 h	**92.32**	**92.18**
early	8	Huang & Sagae 2010			92.1

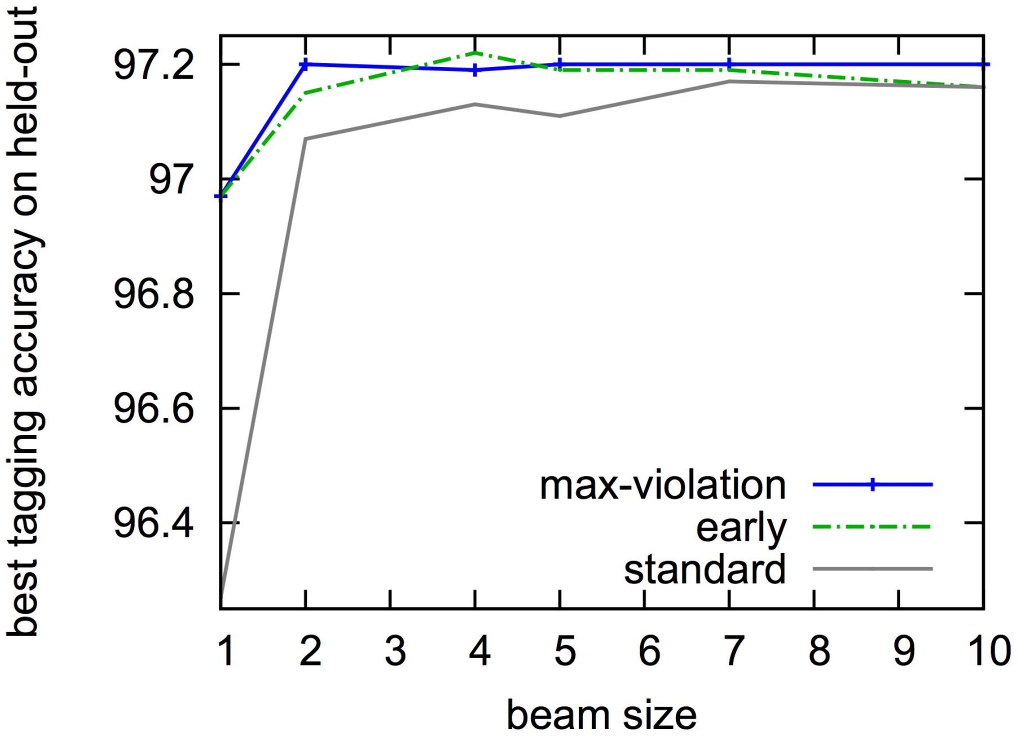
Parsing Results:
Ensuring valid
violations is crucial

Results

method	b	it	time	dev	test
early*		38	15.4 h	92.24	92.09
standard		1	0.4 h	*78.99*	*79.14*
hybrid	8	11	5.6 h	92.26	91.95
latest		9	4.5 h	92.06	91.96
max-viol.		12	5.5 h	**92.32**	**92.18**
early	8	Huang & Sagae 2010			92.1

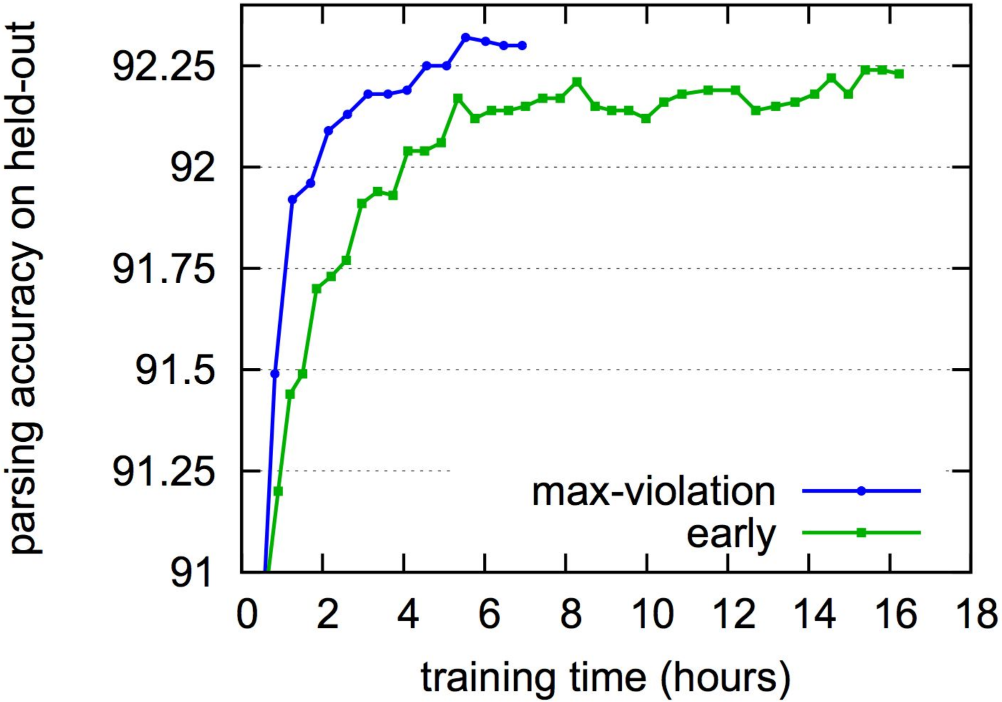
Parsing Results:
Ensuring valid violations is crucial

Tagging Results:
Ensuring valid violations is "helpful"



Tagging and parsing on standard Penn Treebank splits.

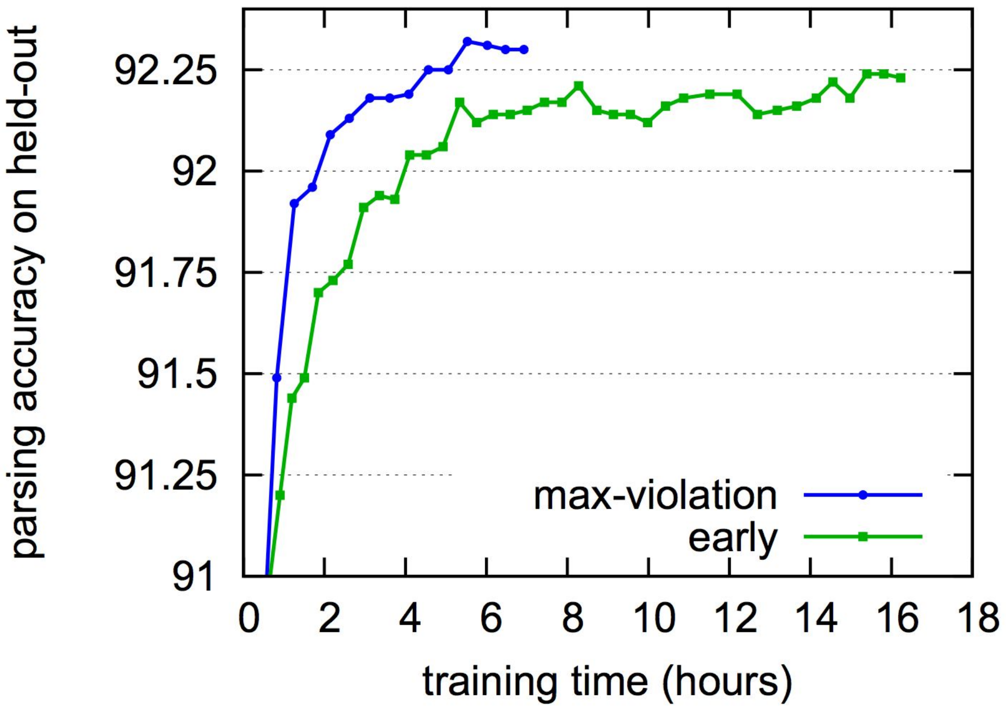
Results



Training time much faster for max-violation than early.

Why?

Results



Training time much faster for max-violation than early.

Why? Probably has to do with the margin of the dataset and informativeness of each update! Early update takes minimally informative updates.