Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Anders Björkelund and Jonas Kuhn

anders@ims.uni-stuttgart.de

IMS, Stuttgart

ACL '14, June 23rd

Table of Contents

Title Breakdown

Learning <u>Structured Perceptrons</u> for <u>Coreference Resolution</u> with <u>Latent Antecedents</u> and <u>Non-local Features</u>

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Coreference Resolution

Group references to the same real-world entities in a document together

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Coreference Resolution

Group references to the same real-world entities in a document together

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Structured Perceptron

Adaptation of a *perceptron classifier* to more complex outputs (structures), e.g., parse trees.

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Latent Antecedents

Let the machine learning algorithm decide on the fly what is the most likely antecedent for a given mention.

(more on next slide)

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Latent Antecedents

Let the machine learning algorithm decide on the fly what is the most likely antecedent for a given mention.

(more on next slide)

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Non-local Features

Allow the classifier to access information beyond a pair of mentions

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Non-local Features

Allow the classifier to access information beyond a pair of mentions

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Title Breakdown

Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features

Non-local Features

Allow the classifier to access information beyond a pair of mentions

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Why latent antecedents?

- Popular approach to learn **pairwise** models use the following heuristic to create training instances (Soon et al., 2001):

  For every non-discourse-first coreferent mention, create

  - a positive instance pairing this mention with its closest preceding coreferent mention
  - negative instances for all pairs with intervening mentions

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Why latent antecedents?

- Popular approach to learn **pairwise** models use the following heuristic to create training instances (Soon et al., 2001):

  For every non-discourse-first coreferent mention, create

  - a positive instance pairing this mention with its closest preceding coreferent mention
  - negative instances for all pairs with intervening mentions

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Why latent antecedents?

- Popular approach to learn **pairwise** models use the following heuristic to create training instances (Soon et al., 2001):

  For every non-discourse-first coreferent mention, create

  - a positive instance pairing this mention with its closest preceding coreferent mention
  - negative instances for all pairs with intervening mentions

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Why latent antecedents?

- Popular approach to learn **pairwise** models use the following heuristic to create training instances (Soon et al., 2001):

  For every non-discourse-first coreferent mention, create

  - a positive instance pairing this mention with its closest preceding coreferent mention
  - negative instances for all pairs with intervening mentions

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Why latent antecedents?

- Popular approach to learn **pairwise** models use the following heuristic to create training instances (Soon et al., 2001):

  For every non-discourse-first coreferent mention, create

  - a positive instance pairing this mention with its closest preceding coreferent mention
  - negative instances for all pairs with intervening mentions

- **Bad** choice for positive example

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Why latent antecedents?

- Popular approach to learn **pairwise** models use the following heuristic to create training instances (Soon et al., 2001):

  For every non-discourse-first coreferent mention, create

  - a positive instance pairing this mention with its closest preceding coreferent mention
  - negative instances for all pairs with intervening mentions

- **Bad** choice for positive example

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Why latent antecedents?

- Popular approach to learn **pairwise** models use the following heuristic to create training instances (Soon et al., 2001):

  For every non-discourse-first coreferent mention, create

  - a positive instance pairing this mention with its closest preceding coreferent mention
  - negative instances for all pairs with intervening mentions

- **Bad** choice for positive example
- No treatment of discourse-first

???

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Coreference Model Paradigms

- **Mention-pair models** recast the problem as a binary classification problem where two mentions are classified as *coreferent* or *disreferent*
  - \+ Rich features (anything from either mention, or the relation between them)
  - – Little context (only two mentions)
- **Entity-mention models** decide whether to merge a single mention into a (partially built) cluster
  - – Poor features (has no explicit *pivot* to compare the mention to)
  - \+ Rich context (can see all mentions of the partially built cluster)
- **Our work** combines the two approaches, keeping the strengths of both

Coreference Model Paradigms

- **Mention-pair models** recast the problem as a binary classification problem where two mentions are classified as *coreferent* or *disreferent*
  - \+ Rich features (anything from either mention, or the relation between them)
  - – Little context (only two mentions)
- **Entity-mention models** decide whether to merge a single mention into a (partially built) cluster
  - – Poor features (has no explicit *pivot* to compare the mention to)
  - \+ Rich context (can see all mentions of the partially built cluster)
- **Our work** combines the two approaches, keeping the strengths of both

Coreference Model Paradigms

- **Mention-pair models** recast the problem as a binary classification problem where two mentions are classified as *coreferent* or *disreferent*
  - \+ Rich features (anything from either mention, or the relation between them)
  - – Little context (only two mentions)
- **Entity-mention models** decide whether to merge a single mention into a (partially built) cluster
  - – Poor features (has no explicit *pivot* to compare the mention to)
  - \+ Rich context (can see all mentions of the partially built cluster)
- **Our work** combines the two approaches, keeping the strengths of both

Table of Contents

Notation

- $M = \{m_0, m_1, ..., m_n\}$ – set of mentions
  - m_0 – special dummy mention (*root*)
- **Mention-pair**

$$\langle a_i, m_i \rangle, \quad a_i < m_i$$

- **Coreference assignment**

$$y = \{\langle a_1, m_1 \rangle, \langle a_2, m_2 \rangle, ..., \langle a_n, m_n \rangle\}$$

  - Set of mention-pairs, every m_i occurs exactly once as the second mention of a pair
  - Every mention has exactly one antecedent – can be thought of as a tree

Notation

- $M = \{m_0, m_1, ..., m_n\}$ – set of mentions
  - m_0 – special dummy mention (*root*)
- **Mention-pair**

$$\langle a_i, m_i \rangle, \quad a_i < m_i$$

- **Coreference assignment**

$$y = \{\langle a_1, m_1 \rangle, \langle a_2, m_2 \rangle, ..., \langle a_n, m_n \rangle\}$$

  - Set of mention-pairs, every m_i occurs exactly once as the second mention of a pair
  - Every mention has exactly one antecedent – can be thought of as a tree

Notation

- $M = \{m_0, m_1, ..., m_n\}$ – set of mentions
  - m_0 – special dummy mention (*root*)
- **Mention-pair**

$$\langle a_i, m_i \rangle, \quad a_i < m_i$$

- **Coreference assignment**

$$y = \{\langle a_1, m_1 \rangle, \langle a_2, m_2 \rangle, ..., \langle a_n, m_n \rangle\}$$

  - Set of mention-pairs, every m_i occurs exactly once as the second mention of a pair
  - Every mention has exactly one antecedent – can be thought of as a tree

Example

Assignment

$$
\begin{aligned}
y = \{&\langle m_0, \text{Drug Emporium Inc.}\rangle \\
&\langle \text{Drug Emporium Inc.}, \text{this drugstore chain}\rangle \\
&\langle \text{Drug Emporium Inc.}, \text{the company}\rangle \\
&\langle \text{the company}, \text{company}\rangle \\
&\langle m_0, \text{Gary Wilber}\rangle \\
&\langle \text{Gary Wilber}, \text{He}\rangle \\
&\langle \text{Gary Wilber}, \text{Gary Wilber}\rangle\}
\end{aligned}
$$

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Example

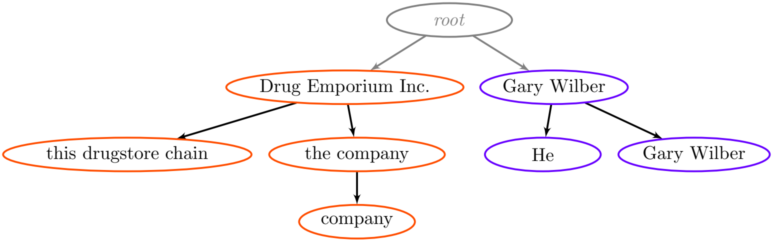
Assignment

$$
\begin{aligned}
y = \{ & \langle m_0, \text{Drug Emporium Inc.} \rangle \\
& \langle \text{Drug Emporium Inc.}, \text{this drugstore chain} \rangle \\
& \langle \text{Drug Emporium Inc.}, \text{the company} \rangle \\
& \langle \text{the company}, \text{company} \rangle \\
& \langle m_0, \text{Gary Wilber} \rangle \\
& \langle \text{Gary Wilber}, \text{He} \rangle \\
& \langle \text{Gary Wilber}, \text{Gary Wilber} \rangle \}
\end{aligned}
$$

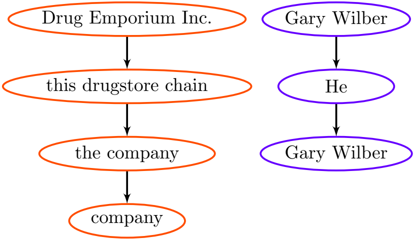
$\boxed{m_0}$

[Drug Emporium Inc.] said [Gary Wilber] was named CEO of [this drugstore chain]. [He] succeeds his father, Philip T. Wilber, who founded [the company] and remains chairman. Robert E. Lyons III, who headed the [company]'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber].

Viewing this as a tree



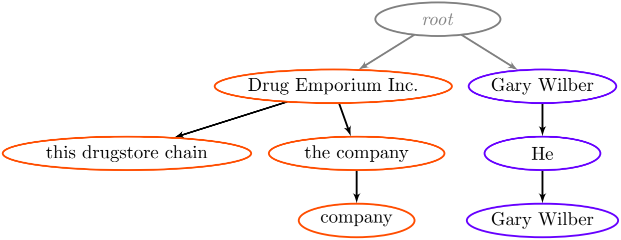
Old way for training



- Unintuitive antecedents
- No root node

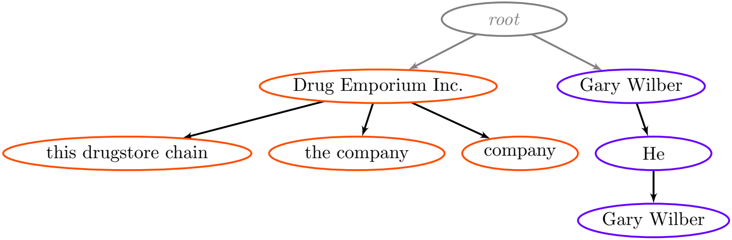
Note that there might be multiple trees

Correct



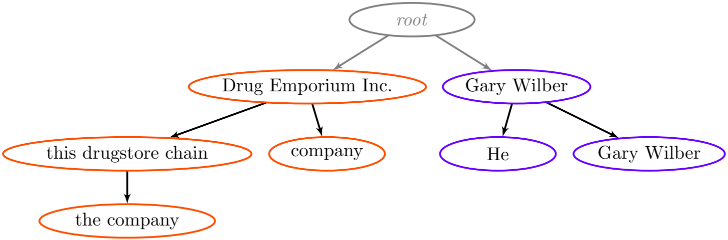
Note that there might be multiple trees

Correct



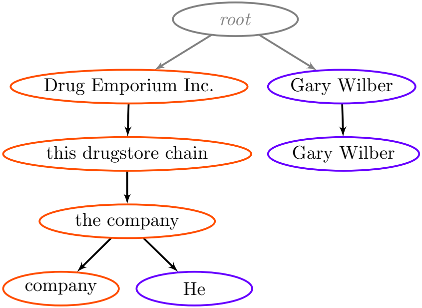
Note that there might be multiple trees

Correct



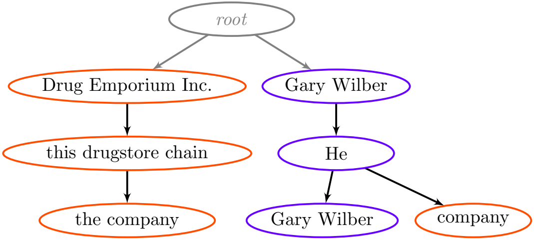
Note that there might be multiple trees

Incorrect



Note that there might be multiple trees

Incorrect



Scoring

- Feature mapping function

$$\phi : M \times M \rightarrow \mathbb{R}^n$$

  maps pairs of mentions to high-dimensional feature vector

- Weight vector w and feature vector gives score of mention pair:

$$\text{score}(\langle a_i, m_i \rangle) = w \cdot \phi(\langle a_i, m_i \rangle)$$

- Score of a tree y

$$\text{score}(y) = \sum_{\langle a_i, m_i \rangle \in y} \text{score}(\langle a_i, m_i \rangle)$$

Scoring

- Feature mapping function

$$\phi : M \times M \rightarrow \mathbb{R}^n$$

  maps pairs of mentions to high-dimensional feature vector

- Weight vector w and feature vector gives score of mention pair:

$$\text{score}(\langle a_i, m_i \rangle) = w \cdot \phi(\langle a_i, m_i \rangle)$$

- Score of a tree y

$$\text{score}(y) = \sum_{\langle a_i, m_i \rangle \in y} \text{score}(\langle a_i, m_i \rangle)$$

Scoring

- Feature mapping function

$$\phi : M \times M \to \mathbb{R}^n$$

  maps pairs of mentions to high-dimensional feature vector

- Weight vector w and feature vector gives score of mention pair:

$$\text{score}(\langle a_i, m_i \rangle) = w \cdot \phi(\langle a_i, m_i \rangle)$$

- Score of a tree y

$$\text{score}(y) = \sum_{\langle a_i, m_i \rangle \in y} \text{score}(\langle a_i, m_i \rangle)$$

Features

Various feature templates

- **Distance**, **StringMatch**, **Nestedness**
- **Lexicalized** – First, last, previous, following, head word
- **Syntactic** information from the mentions
- ...

All **local** – looks at one mention, or one particular pair

Features

Various feature templates

- **Distance**, **StringMatch**, **Nestedness**
- **Lexicalized** – First, last, previous, following, head word
- **Syntactic** information from the mentions
- ...

All **local** – looks at one mention, or one particular pair

Some more notation

- Let
$$\mathcal{Y}(M)$$
denote the set of **possible** trees over M

- Let
$$\tilde{\mathcal{Y}}(M)$$
denote the set of all **correct** trees over M

- **Note** that
$$\tilde{\mathcal{Y}}(M) \subseteq \mathcal{Y}(M)$$

Some more notation

- Let

$$\mathcal{Y}(M)$$

denote the set of **possible** trees over M

- Let

$$\tilde{\mathcal{Y}}(M)$$

denote the set of all **correct** trees over M

- **Note** that

$$\tilde{\mathcal{Y}}(M) \subseteq \mathcal{Y}(M)$$

Search problem(s)

- The **search problem** becomes

- Prediction

$$\hat{y} = \arg\max_{y \in \mathcal{Y}(\mathcal{M})} \mathit{score}(y)$$

Search problem(s)

- The **search problem** becomes
- Prediction

$$\hat{y} = \arg\max_{y \in \mathcal{Y}(\mathcal{M})} \mathit{score}(y)$$

- Latent tree

$$\tilde{y} = \arg\max_{y \in \tilde{\mathcal{Y}}(\mathcal{M})} \mathit{score}(y)$$

Solving the search problem

- Can't afford to enumerate and score all possible trees
- However, with only local features, the search problem can be solved exactly using greedy search:

$y = \{\}$
for $i \in 1..n$ **do** ▷ For every mention
$\quad y = y \cup \underset{m_q \in M, q<i}{\arg\max}\ \text{score}(\langle m_q, m_i \rangle)$ ▷ Find best antecedent
return y

Solving the search problem

- Can't afford to enumerate and score all possible trees
- However, with only local features, the search problem can be solved exactly using greedy search:

$y = \{\}$
for $i \in 1..n$ **do** ▷ For every mention
$\quad y = y \cup \arg\max_{m_q \in M, q<i} \text{score}(\langle m_q, m_i \rangle)$ ▷ Find best antecedent
return y

Finding the weight vector

- Structured perceptron training

$w = \overrightarrow{0}$

for $t \in 1..T$ **do**

  for $M_i \in D$ **do**

    $\hat{y}_i = \arg\max_{y \in \mathcal{Y}(M)} score(y)$

    if $\neg\textsc{Correct}(\hat{y}_i)$ **then**

      $\Delta = \Phi(\hat{y}_i) - \Phi(\tilde{y}_i)$

      $w = w + \Delta$

return w

Finding the weight vector

- Structured perceptron training

$w = \overrightarrow{0}$ ▷ Initialize

for $t \in 1..T$ **do** ▷ For some iterations

  for $M_i \in D$ **do** ▷ For every document

    $\hat{y}_i = \arg\max_{y \in \mathcal{Y}(M)} score(y)$

    if $\neg \textsc{Correct}(\hat{y}_i)$ **then**

      $\Delta = \Phi(\hat{y}_i) - \Phi(\tilde{y}_i)$

      $w = w + \Delta$

return w ▷ Return

Finding the weight vector

- Structured perceptron training

$w = \overrightarrow{0}$

for $t \in 1..T$ **do**

  for $M_i \in D$ **do**

    $\hat{y}_i = \underset{y \in \mathcal{Y}(M)}{\arg\max}\ score(y)$ ▷ Predict

    if $\neg\textsc{Correct}(\hat{y}_i)$ **then** ▷ Correct?

      $\Delta = \Phi(\hat{y}_i) - \Phi(\tilde{y}_i)$ ▷ Distance vector

      $w = w + \Delta$ ▷ Perceptron update

return w

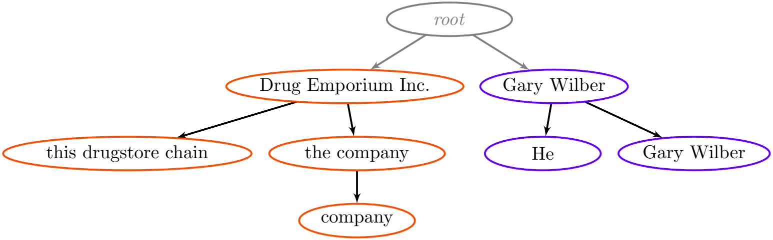
Finding the weight vector

- Structured perceptron training

$w = \overrightarrow{0}$
for $t \in 1..T$ **do**
  for $M_i \in D$ **do**
    $\hat{y}_i = \arg\max_{y \in \mathcal{Y}(M)} score(y)$
    if $\neg\textsc{Correct}(\hat{y}_i)$ **then**
      $\tilde{y}_i = \arg\max_{y \in \tilde{\mathcal{Y}}(M)} score(y)$ ▷ Latent tree
      $\Delta = \Phi(\hat{y}_i) - \Phi(\tilde{y}_i)$
      $w = w + \Delta$
return w

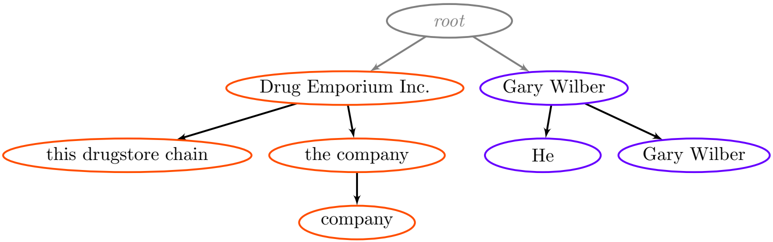
Table of Contents

Non-local features in the tree



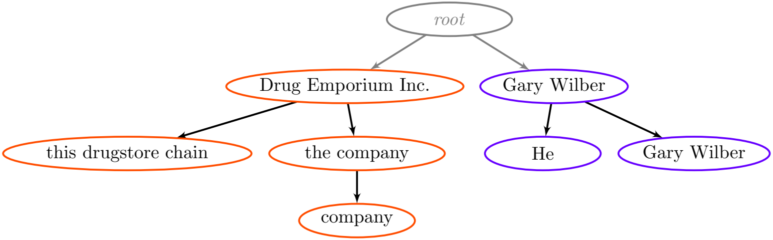
- **Local** features are features over the two mentions that an arc connects
- **Non-local** features can make use of partially predicted (output) structure
  - Head word of grandparent/sibling/etc
  - Current size of cluster
  - How many new clusters begin between head and dependent?
  - (Needs extension of ϕ – see paper)

Non-local features in the tree



- **Local** features are features over the two mentions that an arc connects
- **Non-local** features can make use of partially predicted (output) structure
  - Head word of grandparent/sibling/etc
  - Current size of cluster
  - How many new clusters begin between head and dependent?
  - (Needs extension of ϕ – see paper)

Non-local features in the tree



- **Local** features are features over the two mentions that an arc connects
- **Non-local** features can make use of partially predicted (output) structure
  - Head word of grandparent/sibling/etc
  - Current size of cluster
  - How many new clusters begin between head and dependent?
  - (Needs extension of ϕ – see paper)

Training with non-local features

- The greedy decoder can accommodate non-local features on the partial structure to the left...
- ...at the cost of exact search becoming intractable
- **Dangerous** since we can get incorrect output
  - **not because the weight vector** was wrong, but
  - because **the correct item was discarded** (Huang et al., 2012)
- Standard approach: use **beam search** and **early update** (Collins and Roark, 2004)

Training with non-local features

- The greedy decoder can accommodate non-local features on the partial structure to the left...
- ...at the cost of exact search becoming intractable
- **Dangerous** since we can get incorrect output
  - **not because the weight vector** was wrong, but
  - because **the correct item was discarded** (Huang et al., 2012)
- Standard approach: use **beam search** and **early update** (Collins and Roark, 2004)

Training with non-local features

- The greedy decoder can accommodate non-local features on the partial structure to the left...
- ...at the cost of exact search becoming intractable
- **Dangerous** since we can get incorrect output
  - **not because the weight vector** was wrong, but
  - because **the correct item was discarded** (Huang et al., 2012)
- Standard approach: use **beam search** and **early update** (Collins and Roark, 2004)

Training with non-local features

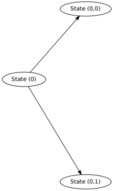
- The greedy decoder can accommodate non-local features on the partial structure to the left...
- ...at the cost of exact search becoming intractable
- **Dangerous** since we can get incorrect output
  - **not because the weight vector** was wrong, but
  - because **the correct item was discarded** (Huang et al., 2012)
- Standard approach: use **beam search** and **early update** (Collins and Roark, 2004)

Beam search

Beam search with $k = 5$

- Start state
- Expand
- Expand
- Prune
- Expand
- Prune
- ...

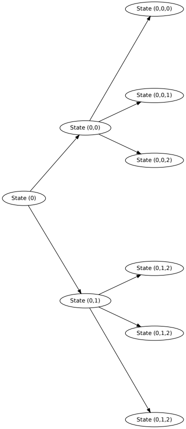
Beam search



Beam search with $k = 5$

- Start state
- Expand
- Expand
- Prune
- Expand
- Prune
- ...

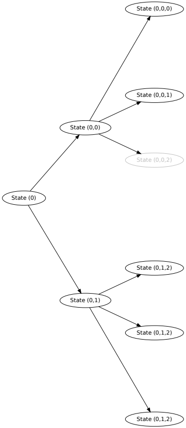
Beam search



Beam search with $k = 5$

- Start state
- Expand
- Expand
- Prune
- Expand
- Prune
- ...

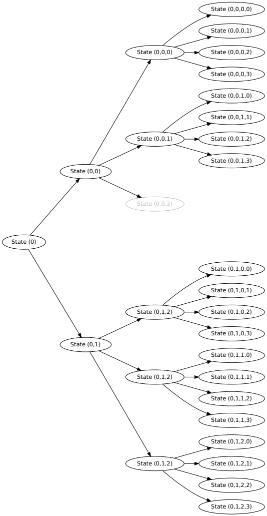
Beam search



Beam search with $k = 5$

- Start state
- Expand
- Expand
- Prune
- Expand
- Prune
- ...

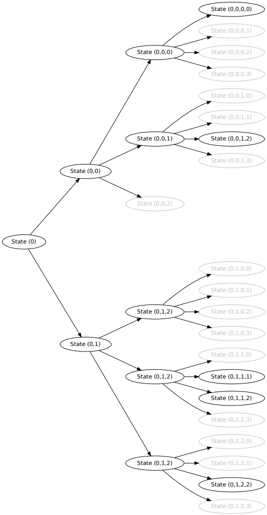
Beam search



Beam search with $k = 5$

- Start state
- Expand
- Expand
- Prune
- Expand
- Prune
- ...

Beam search

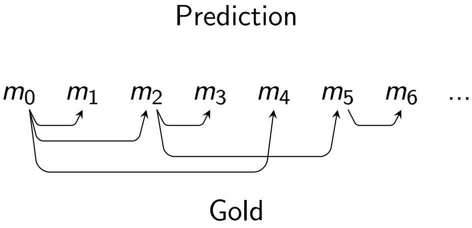


Beam search with $k = 5$

- Start state
- Expand
- Expand
- Prune
- Expand
- Prune
- ...

Early updates (Collins and Roark, 2004)

- Consider one beam item

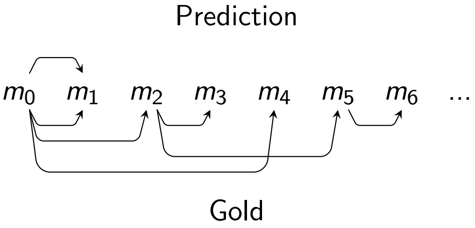


- **Stop** and update weights (on partial structures)
- Move on to next document

- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates (Collins and Roark, 2004)

- Consider one beam item

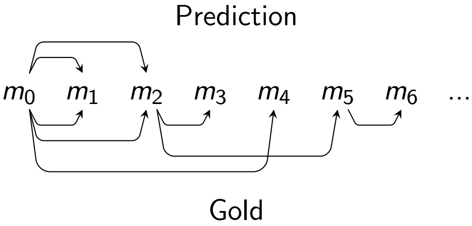


- **Stop** and update weights (on partial structures)
- Move on to next document

- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates (Collins and Roark, 2004)

- Consider one beam item

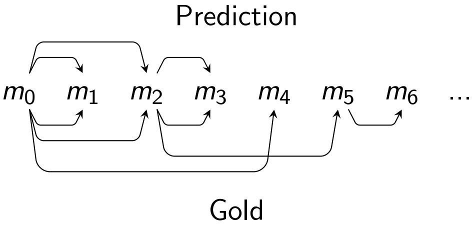


- **Stop** and update weights (on partial structures)
- Move on to next document

- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates (Collins and Roark, 2004)

- Consider one beam item

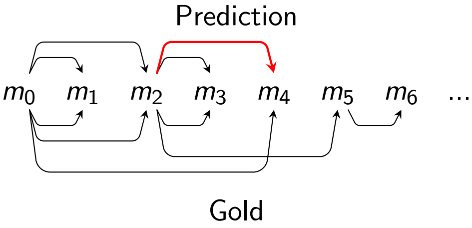


- **Stop** and update weights (on partial structures)
- Move on to next document

- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates (Collins and Roark, 2004)

- Consider one beam item

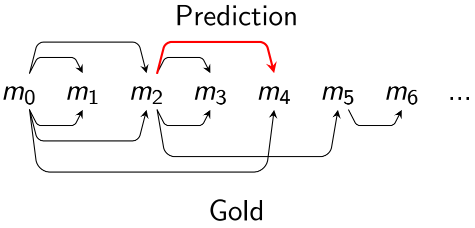


- **Stop** and update weights (on partial structures)
- Move on to next document

- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates (Collins and Roark, 2004)

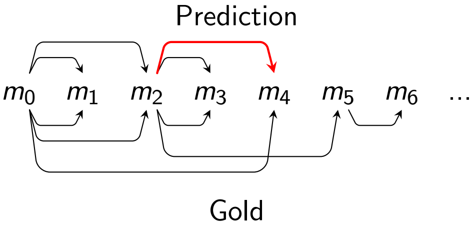
- Consider one beam item



- **Stop** and update weights (on partial structures)
- Move on to next document
- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates (Collins and Roark, 2004)

- Consider one beam item

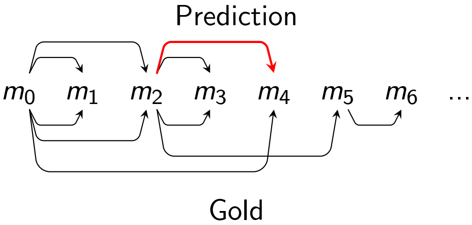


- **Stop** and update weights (on partial structures)
- Move on to next document

- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates (Collins and Roark, 2004)

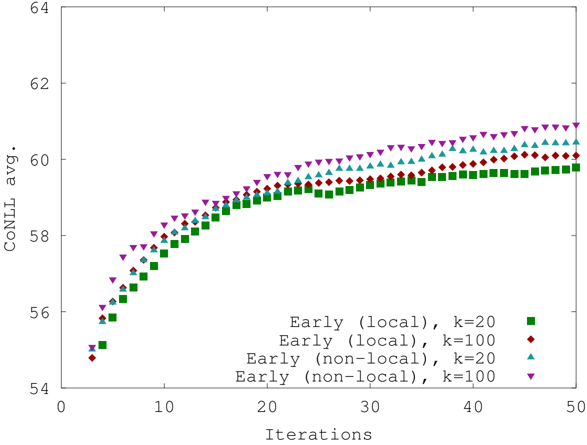
- Consider one beam item



- **Stop** and update weights (on partial structures)
- Move on to next document

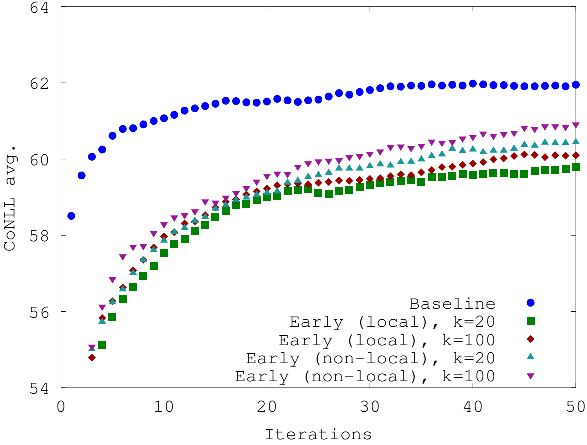
- **Ignores large amounts** of training data
- Two ways of dealing with this
  - More iterations
  - Larger beam size (k)

Early updates vs baseline



- On English development set

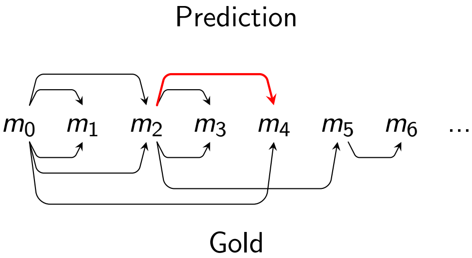
Early updates vs baseline



- On English development set

LaSO updates (Daumé III and Marcu, 2005)

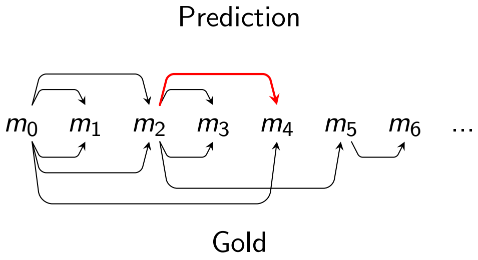
- Consider one beam item



- **Pause** and update weights (on partial structures)
- Revert to correct and continue
- Always reaches the end of the document, but...
- ...**skews the shape** of the latent tree

LaSO updates (Daumé III and Marcu, 2005)

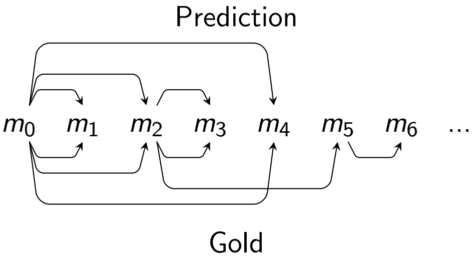
- Consider one beam item



- **Pause** and update weights (on partial structures)
- Revert to correct and continue
- Always reaches the end of the document, but...
- ...**skews the shape** of the latent tree

LaSO updates (Daumé III and Marcu, 2005)

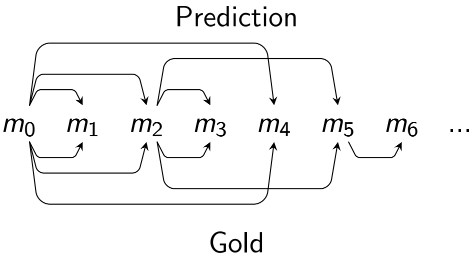
- Consider one beam item



- **Pause** and update weights (on partial structures)
- Revert to correct and continue
- Always reaches the end of the document, but...
- ...**skews the shape** of the latent tree

LaSO updates (Daumé III and Marcu, 2005)

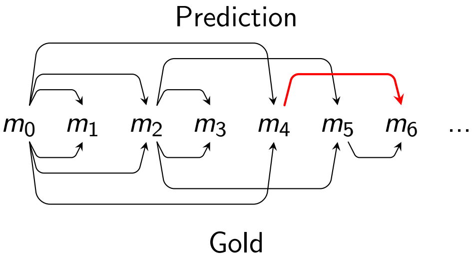
- Consider one beam item



- **Pause** and update weights (on partial structures)
- Revert to correct and continue
- Always reaches the end of the document, but...
- ...**skews the shape** of the latent tree

LaSO updates (Daumé III and Marcu, 2005)

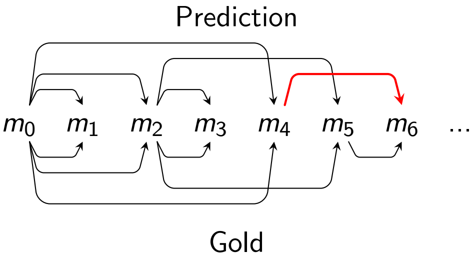
- Consider one beam item



- **Pause** and update weights (on partial structures)
- Revert to correct and continue
- Always reaches the end of the document, but...
- ...**skews the shape** of the latent tree

LaSO updates (Daumé III and Marcu, 2005)

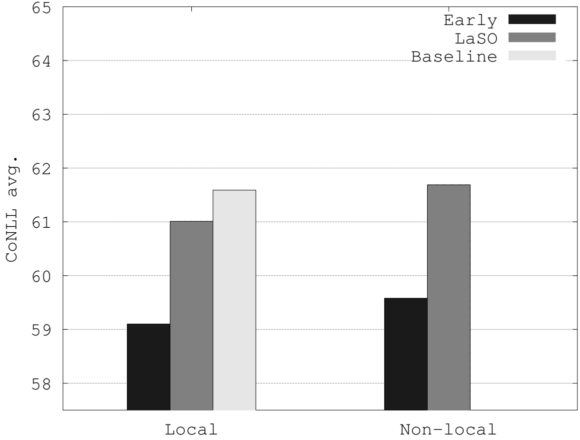
- Consider one beam item



- **Pause** and update weights (on partial structures)
- Revert to correct and continue

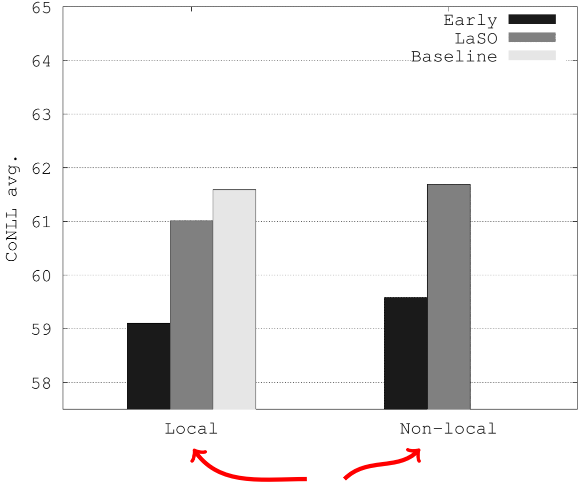
- Always reaches the end of the document, but...
- ...**skews the shape** of the latent tree

Baseline vs Early Updates vs LaSO



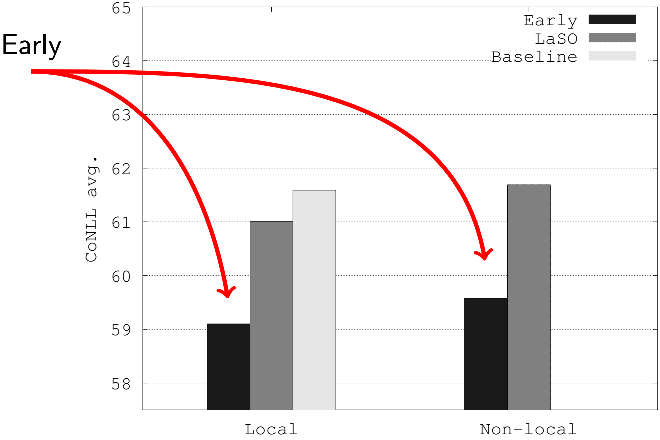
- On English development set

Baseline vs Early Updates vs LaSO



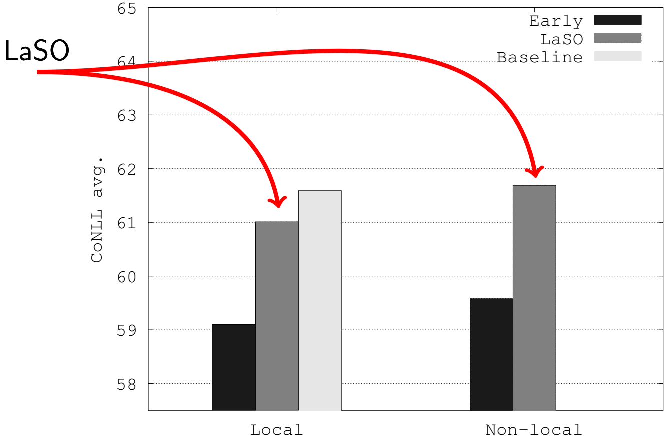
- On English development set

Baseline vs Early Updates vs LaSO



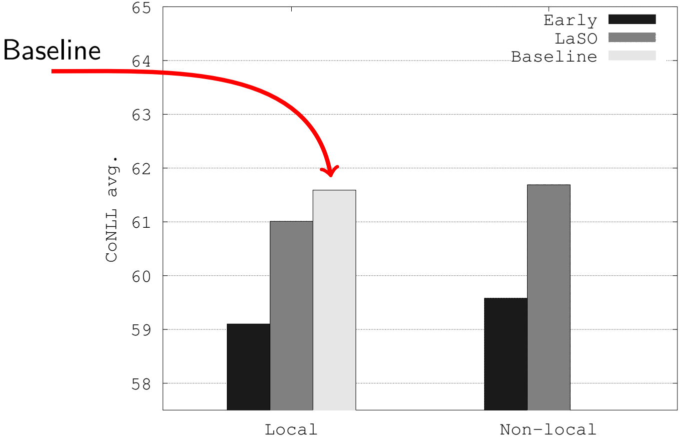
- On English development set

Baseline vs Early Updates vs LaSO



- On English development set

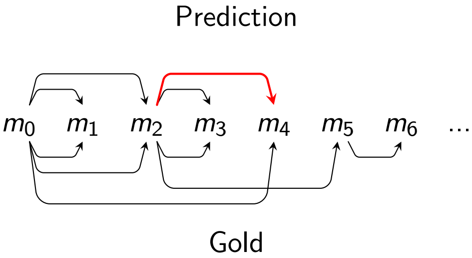
Baseline vs Early Updates vs LaSO



- On English development set

Delayed LaSO updates

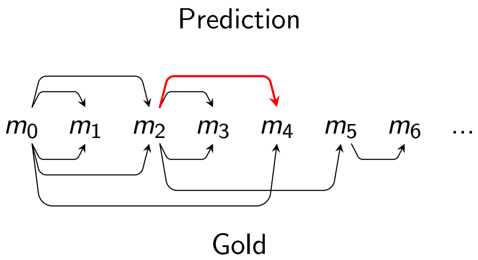
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Delayed LaSO updates

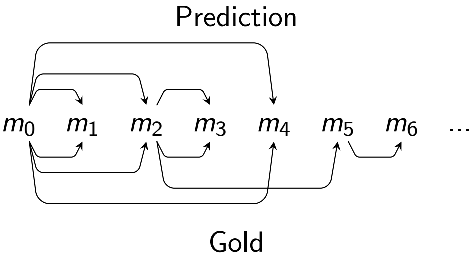
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Delayed LaSO updates

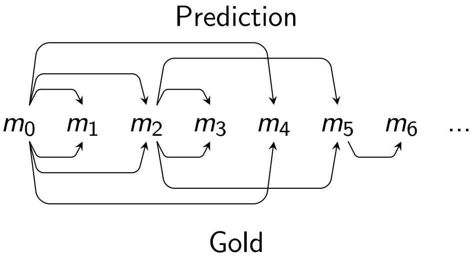
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Delayed LaSO updates

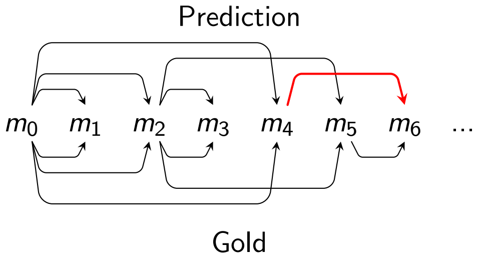
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Delayed LaSO updates

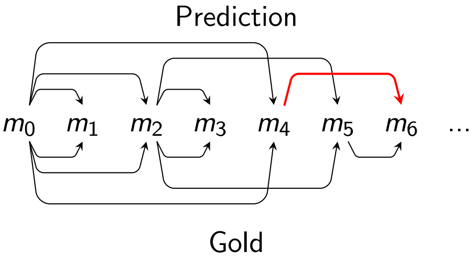
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Delayed LaSO updates

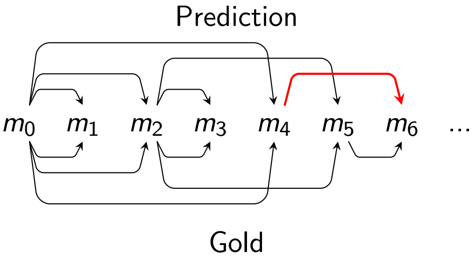
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Delayed LaSO updates

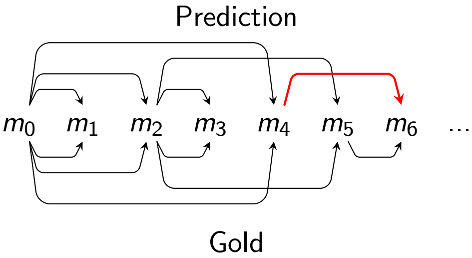
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Delayed LaSO updates

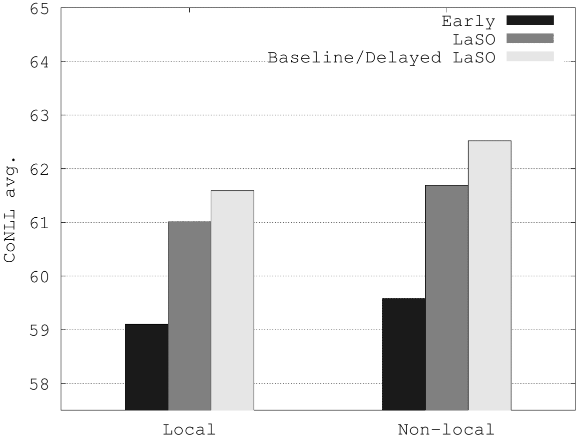
- Consider one beam item



- **Pause**, save the Δ vector that should be used for updates
- Revert to correct and continue
- At the end of the document, update with respect to all Δ's collected

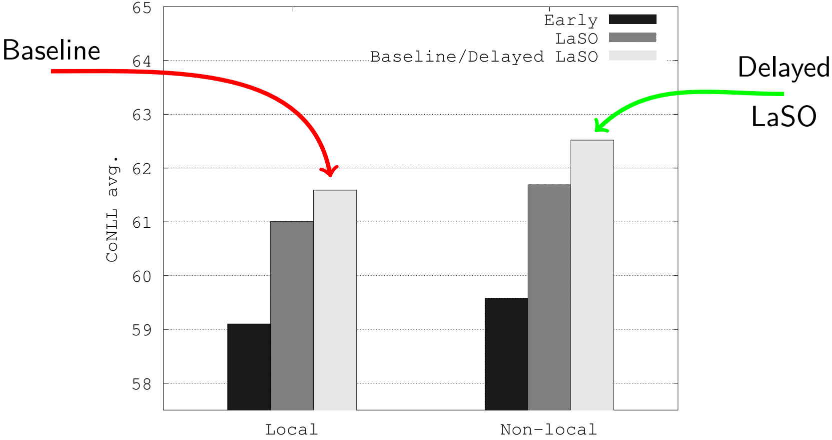
- Doesn't give the learner feedback within instances
- Without non-local features equivalent to baseline algorithm

Baseline vs Early Updates vs LaSO vs delayed LaSO



- On English development set

Baseline vs Early Updates vs LaSO vs delayed LaSO



- On English development set

Table of Contents

Results on benchmark data

	MUC			B^3			$CEAF_m$			$CEAF_e$			CoNLL
	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1	avg.
Arabic													
B&F	43.9	52.51	47.82	35.7	49.77	41.58	43.80	50.03	46.71	40.45	41.86	41.15	43.51
Fernandes	43.63	49.69	46.46	38.39	47.70	42.54	47.60	50.85	49.17	48.16	45.03	46.54	45.18
Our work	**47.53**	**53.3**	**50.25**	**44.14**	49.34	**46.60**	**50.94**	**55.19**	**52.98**	49.20	**49.45**	**49.33**	48.72
Chinese													
B&F	58.72	58.49	58.61	49.17	53.20	51.11	56.68	51.86	54.14	55.36	41.80	47.63	52.45
C&N	59.92	64.69	62.21	51.76	60.26	55.69	59.58	60.45	60.02	**58.84**	51.61	54.99	57.63
Our work	**62.57**	**69.39**	**65.80**	**53.87**	**61.64**	**57.49**	58.75	**64.76**	**61.61**	54.65	**59.33**	**56.89**	60.06
English													
B&F	65.23	70.10	67.58	49.51	60.69	54.47	56.93	59.51	58.19	51.34	49.14	59.21	57.42
D&K	66.58	74.94	70.51	53.20	**64.56**	58.33	59.19	66.23	62.51	52.90	58.06	55.36	61.40
Our work	**67.46**	74.30	**70.72**	**54.96**	62.71	**58.58**	**60.33**	**66.92**	**63.45**	52.27	**59.40**	55.61	61.63

- Evaluation on CoNLL 2012 test sets
- Comparison with best published previous results
- Bold numbers denote significant differences between two best

- B&F – (Björkelund and Farkas, 2012)
- Fernandes – (Fernandes et al., 2012)
- C&N – (Chen and Ng, 2012)
- D&K – (Durrett and Klein, 2013)

Results on benchmark data

	MUC			B^3			$CEAF_m$			$CEAF_e$			CoNLL
	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1	avg.
							Arabic						
B&F	43.9	52.51	47.82	35.7	49.77	41.58	43.80	50.03	46.71	40.45	41.86	41.15	43.51
Fernandes	43.63	49.69	46.46	38.39	47.70	42.54	47.60	50.85	49.17	48.16	45.03	46.54	45.18
Our work	**47.53**	**53.3**	**50.25**	**44.14**	49.34	**46.60**	**50.94**	**55.19**	**52.98**	49.20	**49.45**	**49.33**	48.72
							Chinese						
B&F	58.72	58.49	58.61	49.17	53.20	51.11	56.68	51.86	54.14	55.36	41.80	47.63	52.45
C&N	59.92	64.69	62.21	51.76	60.26	55.69	59.58	60.45	60.02	**58.84**	51.61	54.99	57.63
Our work	**62.57**	**69.39**	**65.80**	**53.87**	**61.64**	**57.49**	58.75	**64.76**	**61.61**	54.65	**59.33**	**56.89**	60.06
							English						
B&F	65.23	70.10	67.58	49.51	60.69	54.47	56.93	59.51	58.19	51.34	49.14	59.21	57.42
D&K	66.58	74.94	70.51	53.20	**64.56**	58.33	59.19	66.23	62.51	52.90	58.06	55.36	61.40
Our work	**67.46**	74.30	**70.72**	**54.96**	62.71	**58.58**	**60.33**	**66.92**	**63.45**	52.27	**59.40**	55.61	61.63

- Evaluation on CoNLL 2012 test sets
- Comparison with best published previous results
- Bold numbers denote significant differences between two best

- B&F – (Björkelund and Farkas, 2012)
- Fernandes – (Fernandes et al., 2012)
- C&N – (Chen and Ng, 2012)
- D&K – (Durrett and Klein, 2013)

Results on benchmark data

	MUC			B^3			$CEAF_m$			$CEAF_e$			CoNLL
	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1	Rec	Prec	F_1	avg.
Arabic													
B&F	43.9	52.51	47.82	35.7	49.77	41.58	43.80	50.03	46.71	40.45	41.86	41.15	43.51
Fernandes	43.63	49.69	46.46	38.39	47.70	42.54	47.60	50.85	49.17	48.16	45.03	46.54	45.18
Our work	**47.53**	**53.3**	**50.25**	**44.14**	49.34	**46.60**	**50.94**	**55.19**	**52.98**	49.20	**49.45**	**49.33**	48.72
Chinese													
B&F	58.72	58.49	58.61	49.17	53.20	51.11	56.68	51.86	54.14	55.36	41.80	47.63	52.45
C&N	59.92	64.69	62.21	51.76	60.26	55.69	59.58	60.45	60.02	**58.84**	51.61	54.99	57.63
Our work	**62.57**	**69.39**	**65.80**	**53.87**	**61.64**	**57.49**	58.75	**64.76**	**61.61**	54.65	**59.33**	**56.89**	60.06
English													
B&F	65.23	70.10	67.58	49.51	60.69	54.47	56.93	59.51	58.19	51.34	49.14	59.21	57.42
D&K	66.58	74.94	70.51	53.20	**64.56**	58.33	59.19	66.23	62.51	52.90	58.06	55.36	61.40
Our work	**67.46**	74.30	**70.72**	**54.96**	62.71	**58.58**	**60.33**	**66.92**	**63.45**	52.27	**59.40**	55.61	61.63

- Evaluation on CoNLL 2012 test sets
- Comparison with best published previous results
- Bold numbers denote significant differences between two best

- B&F – (Björkelund and Farkas, 2012)
- Fernandes – (Fernandes et al., 2012)
- C&N – (Chen and Ng, 2012)
- D&K – (Durrett and Klein, 2013)

Table of Contents

Conclusion

- Experiments on how to train structured perceptrons with latent antecedents and non-local features
- Beam Search and
  - – Early updates
  - – LaSO
  - + Delayed LaSO
- Significant improvements over baseline
- Significant improvements over current state of the art
- Sources available online[1]

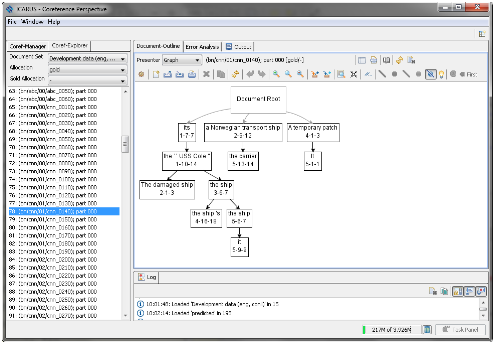
- Delayed LaSO is a general technique applicable to other similar problems

[1] `http://www.ims.uni-stuttgart.de/~anders/coref.html`

Teaser

- Want to look at some of the trees?

⇒ Come see our demo tonight! (Ballroom, starts at 18.50)



Questions

Thank you.
Questions?

References I

Björkelund, A. and Farkas, R. (2012). Data-driven multilingual coreference resolution using resolver stacking. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 49–55, Jeju Island, Korea. Association for Computational Linguistics.

Chen, C. and Ng, V. (2012). Combining the best of two worlds: A hybrid approach to multilingual coreference resolution. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 56–63, Jeju Island, Korea. Association for Computational Linguistics.

Collins, M. and Roark, B. (2004). Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 111–118, Barcelona, Spain.

Daumé III, H. and Marcu, D. (2005). Learning as search optimization: approximate large margin methods for structured prediction. In *ICML*, pages 169–176.

Durrett, G. and Klein, D. (2013). Easy victories and uphill battles in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1971–1982, Seattle, Washington, USA. Association for Computational Linguistics.

Fernandes, E., dos Santos, C., and Milidiú, R. (2012). Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 41–48, Jeju Island, Korea. Association for Computational Linguistics.

References II

Huang, L., Fayong, S., and Guo, Y. (2012). Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–151, Montréal, Canada. Association for Computational Linguistics.

Soon, W. M., Ng, H. T., and Lim, D. C. Y. (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.