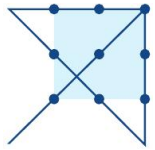
Heidelberg Institute for
Theoretical Studies

Latent Structures for Coreference Resolution

Sebastian Martschat and Michael Strube

Heidelberg Institute for Theoretical Studies gGmbH

Coreference Resolution



Coreference resolution is the task of determining which mentions in a text refer to the same entity.

An Example

Vicente del Bosque admits it will be difficult for him to select David de Gea in Spain's squad if the goalkeeper remains on the sidelines at Manchester United.

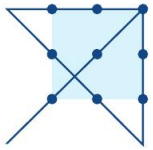
de Gea's long-anticipated transfer to Real Madrid fell through on Monday due to miscommunication between the Spanish club and United and he will stay at Old Trafford until at least January.

An Example

Vicente del Bosque admits it will be difficult for **him** to select **David de Gea** in Spain's squad if **the goalkeeper** remains on the sidelines at **Manchester United**.

de Gea's long-anticipated transfer to **Real Madrid** fell through on Monday due to miscommunication between **the Spanish club** and **United** and **he** will stay at Old Trafford until at least January.

Outline



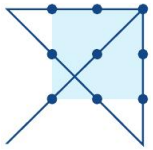
Motivation

Structures for Coreference Resolution

Experiments and Analysis

Conclusions and Future Work

Outline



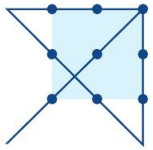
Motivation

Structures for Coreference Resolution

Experiments and Analysis

Conclusions and Future Work

General Paradigm

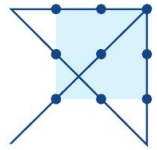


Consolidate pairwise decisions for anaphor-antecedent pairs

Vicente del Bosque admits it will be difficult for **him** to select **David de Gea** in Spain's squad if **the goalkeeper** remains on the sidelines at **Manchester United**.

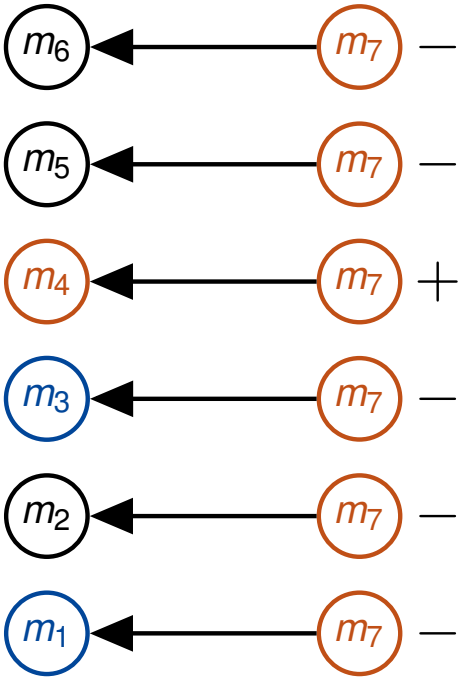
de Gea's long-anticipated transfer to **Real Madrid** fell through on Monday due to miscommunication between **the Spanish club** and **United** and **he** will stay at Old Trafford until at least January.

General Paradigm



Consolidate pairwise decisions for anaphor-antecedent pairs

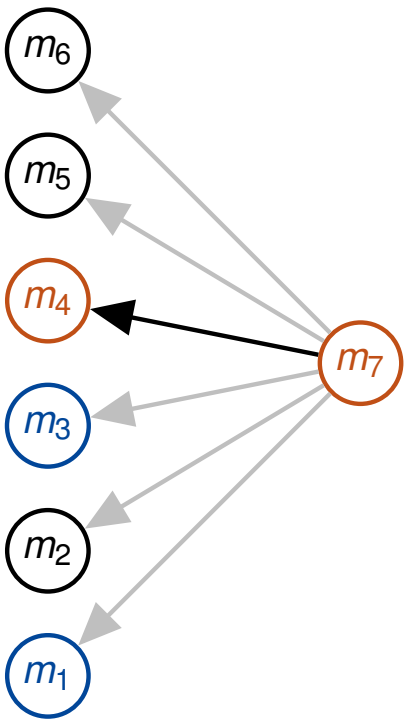
Mention Pairs



Vicente del Bosque admits it will be difficult for **him** to select **David de Gea** in Spain's squad if **the goalkeeper** remains on the sidelines at **Manchester United**.

de Gea's long-anticipated transfer to **Real Madrid** fell through on Monday due to miscommunication between **the Spanish club** and **United** and **he** will stay at Old Trafford until at least January.

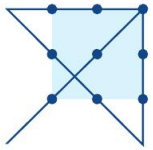
Mention Ranking

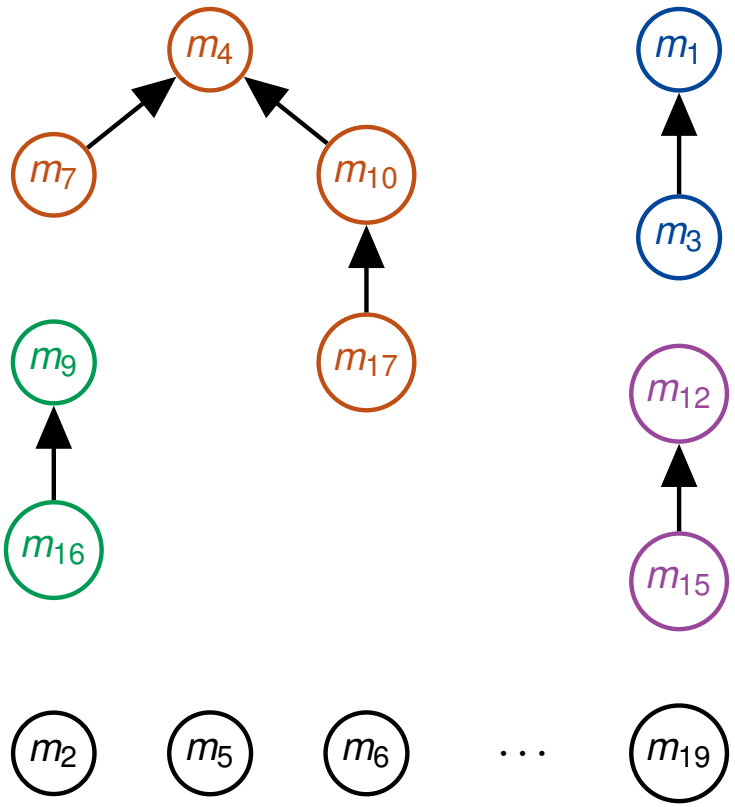


Vicente del Bosque admits it will be difficult for **him** to select **David de Gea** in Spain's squad if **the goalkeeper** remains on the sidelines at **Manchester United**.

de Gea's long-anticipated transfer to **Real Madrid** fell through on Monday due to miscommunication between **the Spanish club** and **United** and **he** will stay at Old Trafford until at least January.

Antecedent Trees

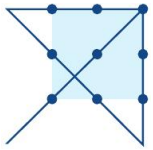




Vicente del Bosque admits it will be difficult for **him** to select **David de Gea** in Spain's squad if **the goalkeeper** remains on the sidelines at **Manchester United**.

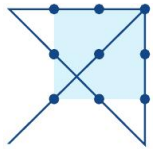
de Gea's long-anticipated transfer to **Real Madrid** fell through on Monday due to miscommunication between **the Spanish club** and **United** and **he** will stay at Old Trafford until at least January.

Unifying Approaches



- approaches operate on structures not annotated in training data
- we can view these structures as *latent structures*

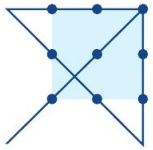
Unifying Approaches



- approaches operate on structures not annotated in training data
- we can view these structures as *latent structures*

$\rightarrow$ devise unified representation of approaches in terms of these structures

Outline



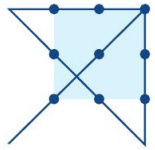
Motivation

Structures for Coreference Resolution

Experiments and Analysis

Conclusions and Future Work

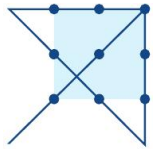
Final Goal



Learn a mapping

$$f\colon\ \mathcal{X} \to \mathcal{H} \times \mathcal{Z}$$

Final Goal

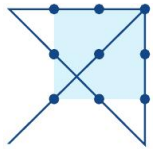


Learn a mapping

$$f: \mathcal{X} \rightarrow \mathcal{H} \times \mathcal{Z}$$

- $x \in \mathcal{X}$: structured input
- documents containing mentions and linguistic information

Final Goal

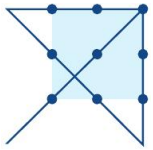


Learn a mapping

$$f\colon\ \mathcal{X} \rightarrow \mathcal{H} \times \mathcal{Z}$$

- $h \in \mathcal{H}$: document-level latent structure we actually predict
- mention pairs, antecedent trees, ...
- employ *graph-based* latent structures

Final Goal

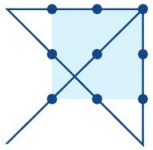


Learn a mapping

$$f: \mathcal{X} \to \mathcal{H} \times \mathcal{Z}$$

Latent structures: subclass of directed labeled graphs $G = (V, A, L)$

Final Goal



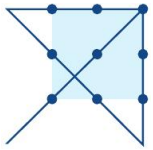
Learn a mapping

$$f\colon\ \mathcal{X} \to \mathcal{H} \times \mathcal{Z}$$

Latent structures: subclass of directed labeled graphs $G = (V, A, L)$

Nodes V: mentions plus dummy mention m_0 for anaphoricity detection

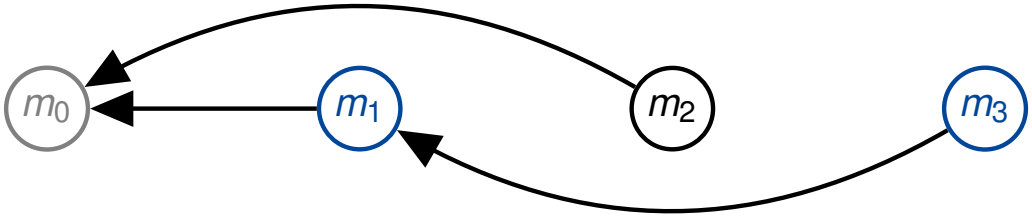
Final Goal



Learn a mapping

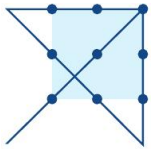
$$f\colon\ \mathcal{X} \to \mathcal{H} \times \mathcal{Z}$$

Latent structures: subclass of directed labeled graphs $G = (V, A, L)$



Arcs A: subset of all backward arcs

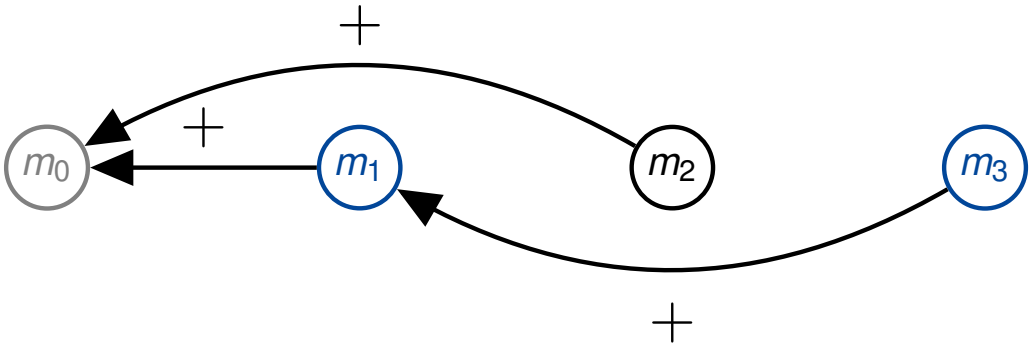
Final Goal



Learn a mapping

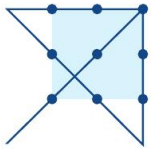
$$f: \mathcal{X} \to \mathcal{H} \times \mathcal{Z}$$

Latent structures: subclass of directed labeled graphs $G = (V, A, L)$



Labels L: labels for arcs

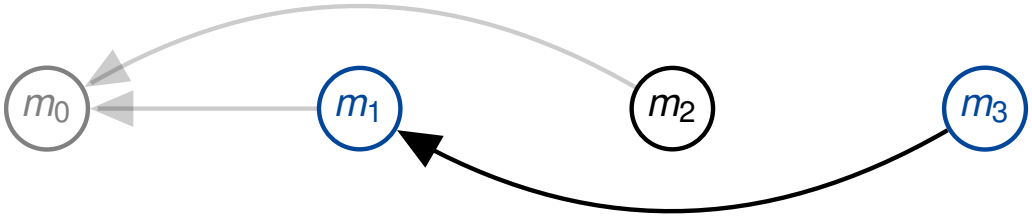
Final Goal



Learn a mapping

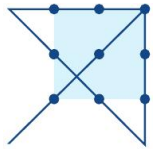
$$f\colon\ \mathcal{X} \to \mathcal{H} \times \mathcal{Z}$$

Latent structures: subclass of directed labeled graphs $G = (V, A, L)$



Graph can be split into *substructures* which are handled individually

Final Goal

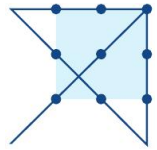


Learn a mapping

$$f\colon\ \mathcal{X} \to \mathcal{H} \times \mathcal{Z}$$

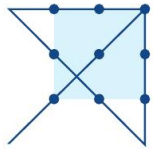
- $z \in \mathcal{Z}$: mapping of mentions to entity identifiers
- inferred via latent $h \in \mathcal{H}$

Linear Models



Employ an edge-factored linear model:

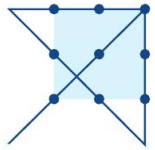
Linear Models



Employ an edge-factored linear model:

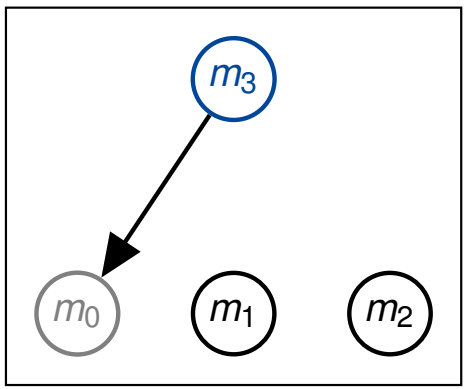
$$f(x) = \arg\max_{(h,z)\in\mathcal{H}_x\times\mathcal{Z}_x} \sum_{a\in h} \langle \theta, \phi(x,a,z) \rangle$$

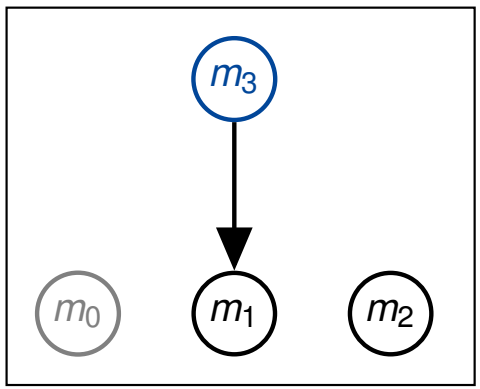
Linear Models

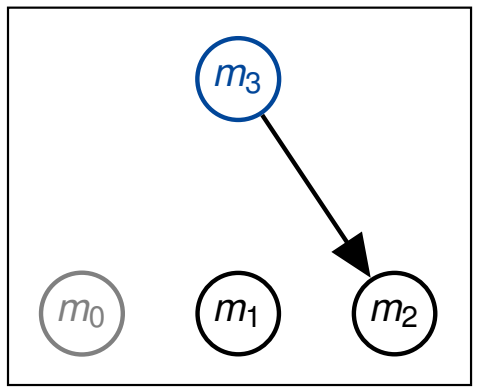


Employ an edge-factored linear model:

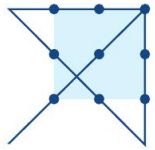
$$f(x) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \sum_{a \in h} \langle \theta, \phi(x, a, z) \rangle$$





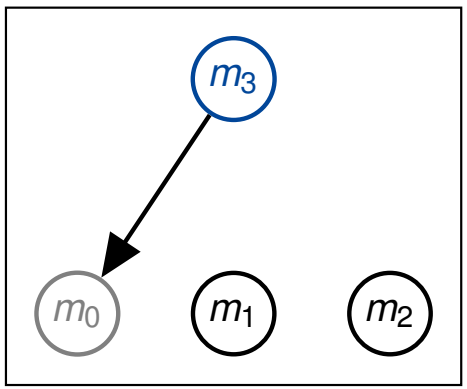


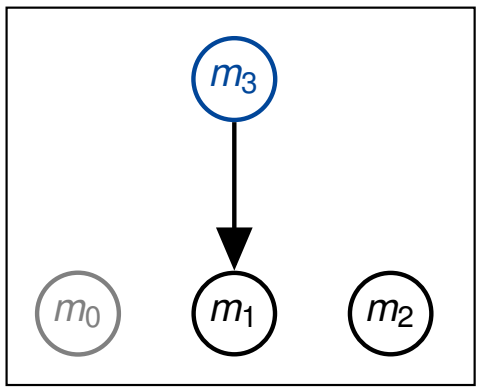
Linear Models

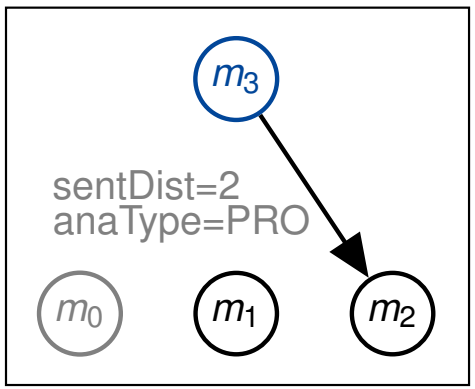


Employ an edge-factored linear model:

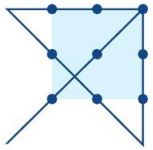
$$f(x) = \arg\max_{(h,z)\in\mathcal{H}_x\times\mathcal{Z}_x} \sum_{a\in h} \langle \theta, \phi(x,a,z) \rangle$$





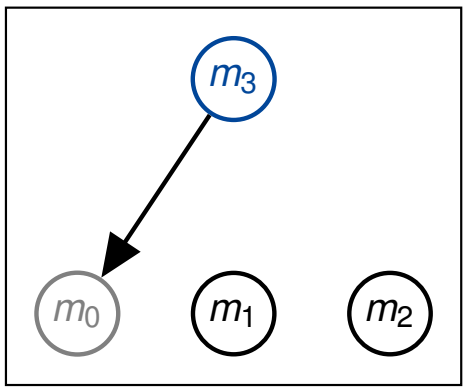


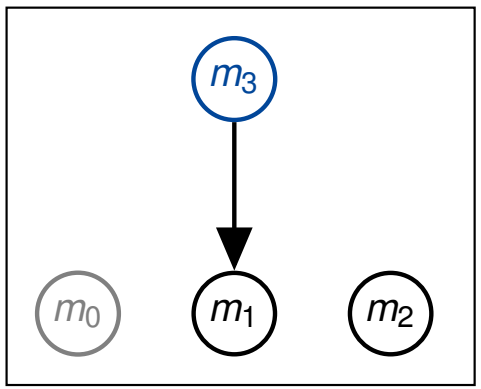
Linear Models

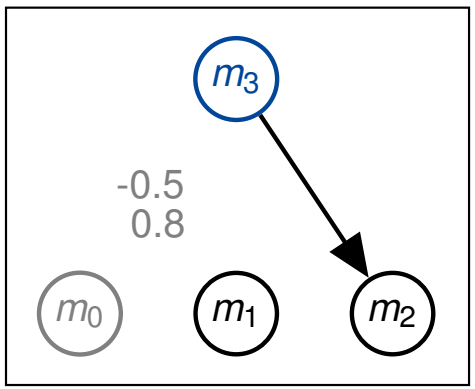


Employ an edge-factored linear model:

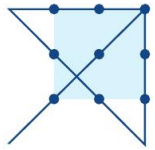
$$f(x) = \arg\max_{(h,z)\in\mathcal{H}_x\times\mathcal{Z}_x} \sum_{a\in h} \langle \theta, \phi(x,a,z) \rangle$$





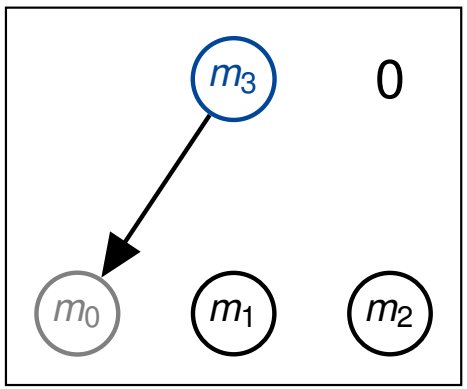


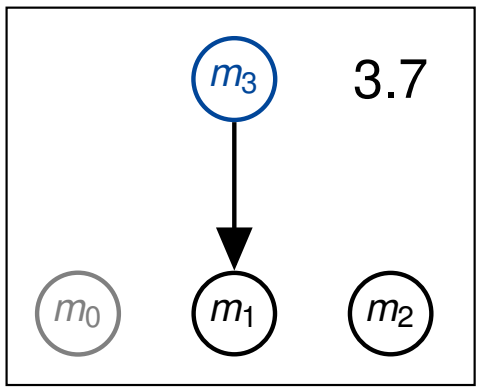
Linear Models

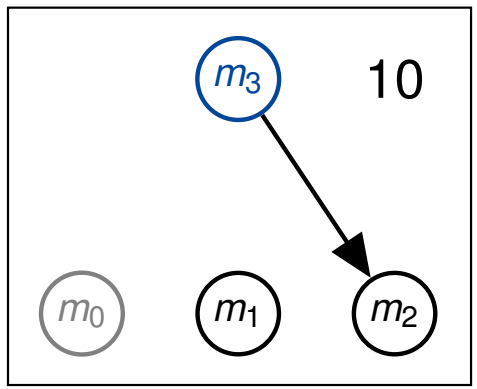


Employ an edge-factored linear model:

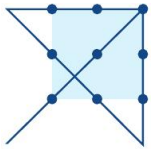
$$f(x) = \arg\max_{(h,z)\in\mathcal{H}_x\times\mathcal{Z}_x} \sum_{a\in h} \langle \theta, \phi(x,a,z) \rangle$$





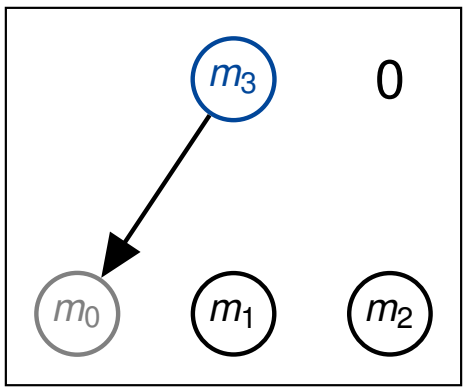


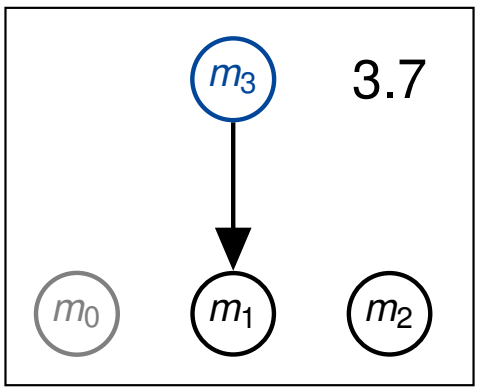
Linear Models

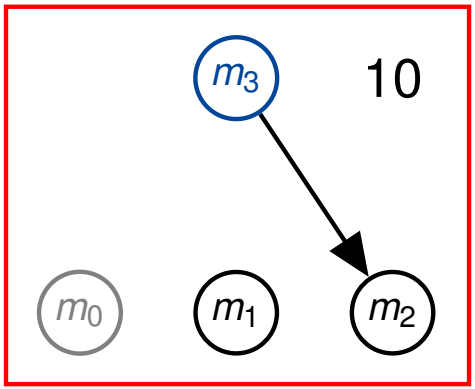


Employ an edge-factored linear model:

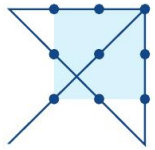
$$f(x) = \arg\max_{(h,z)\in\mathcal{H}_x\times\mathcal{Z}_x} \sum_{a\in h} \langle \theta, \phi(x,a,z) \rangle$$







Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\mathtt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

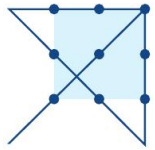
$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\mathtt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$

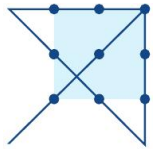
$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\mathrm{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$

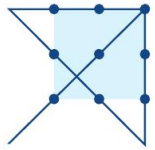
$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

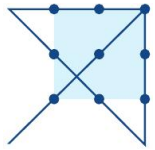
$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

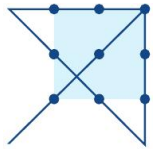
$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

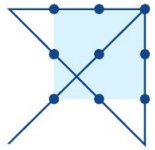
$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

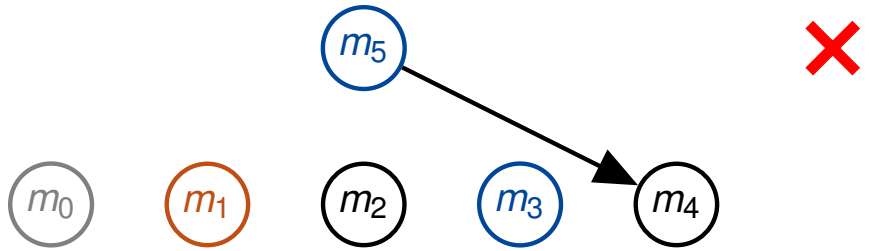
$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$

$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$

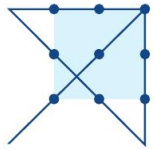
if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ



Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

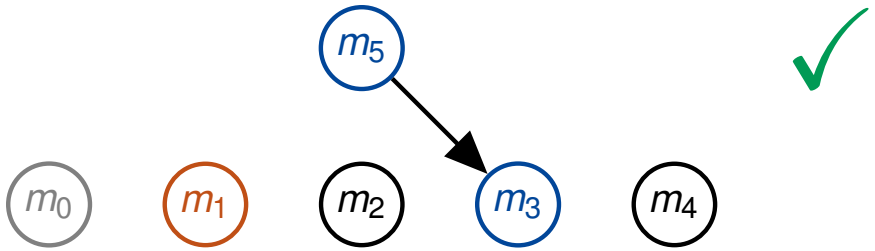
$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

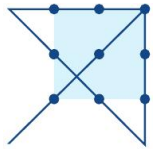
if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ



Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

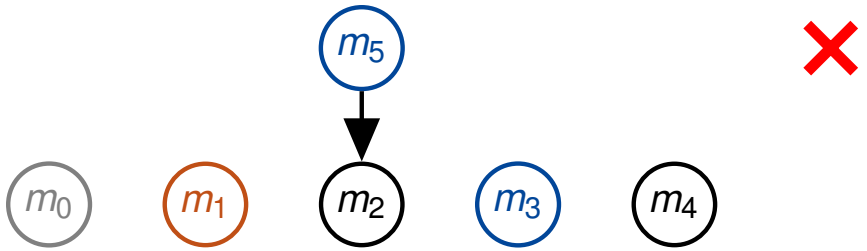
$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

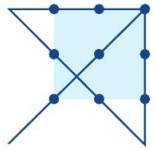
if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ



Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

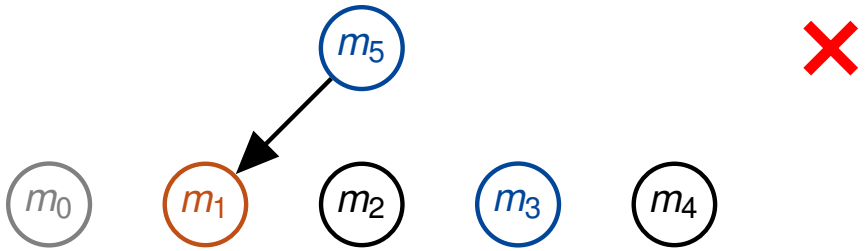
$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

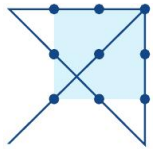
if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ



Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

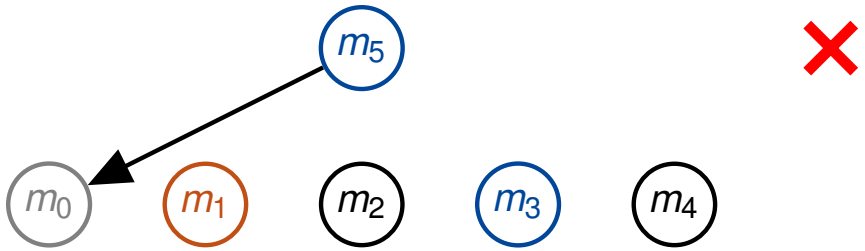
$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

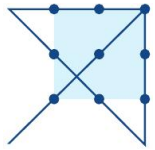
if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ



Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

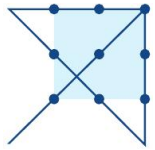
$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

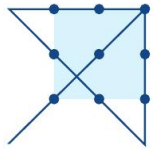
if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Reward solutions with high cost: large-margin approach

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$

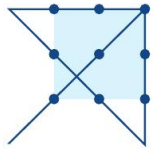
$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

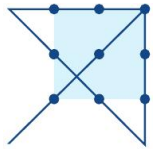
$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

Output: Weight vector θ

Learning: Perceptron



Input: Training set $\mathcal{D}$, cost function c, number of epochs n

function PERCEPTRON($\mathcal{D}$, c, n)

Set $\theta = (0, \ldots, 0)$

for $\texttt{epoch} = 1, \ldots, n$ **do**

for $(x, z) \in \mathcal{D}$ **do**

$$\hat{h}_{\text{opt}} = \arg\max_{h \in \mathcal{H}_{x,z}} \langle \theta, \phi(x, h, z) \rangle$$

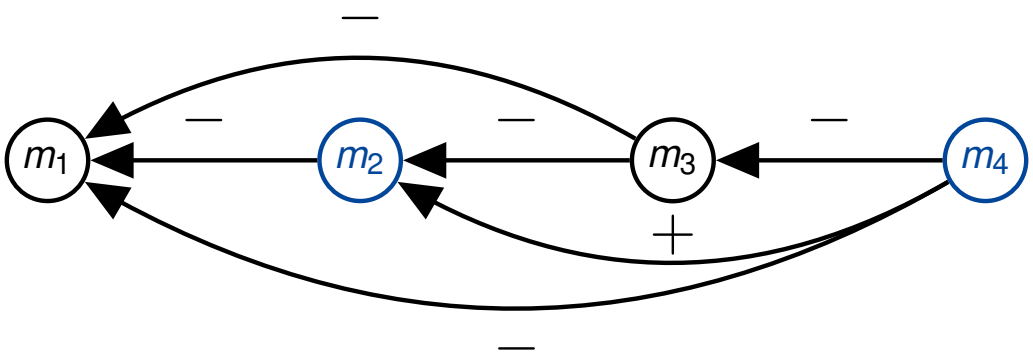
$$(\hat{h}, \hat{z}) = \arg\max_{(h,z) \in \mathcal{H}_x \times \mathcal{Z}_x} \langle \theta, \phi(x, h, z) \rangle + c(x, h, \hat{h}_{\text{opt}}, z)$$

if $\hat{h}$ does not encode z **then**

Set $\theta = \theta + \phi(x, \hat{h}_{\text{opt}}, z) - \phi(x, \hat{h}, \hat{z})$

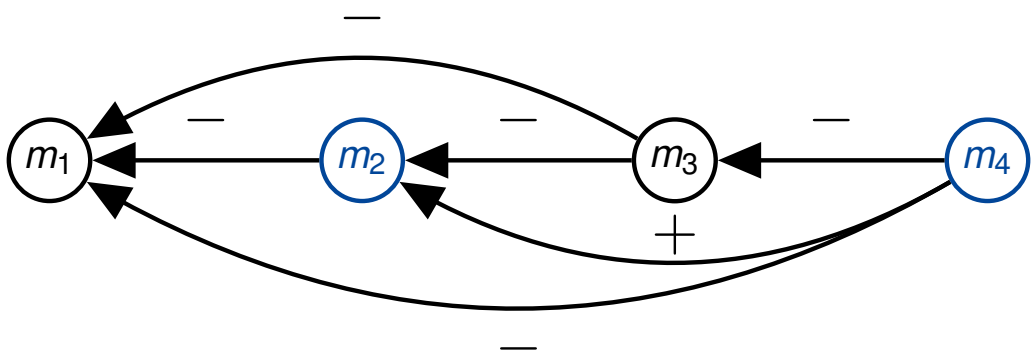
Output: Weight vector θ

Mention Pair



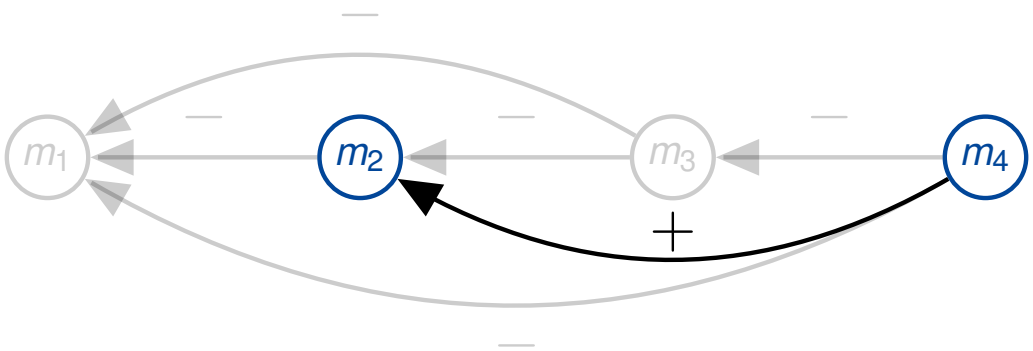
Soon et al. (2001), Ng and Cardie (2002), Bengtson and Roth (2008), ...

Mention Pair



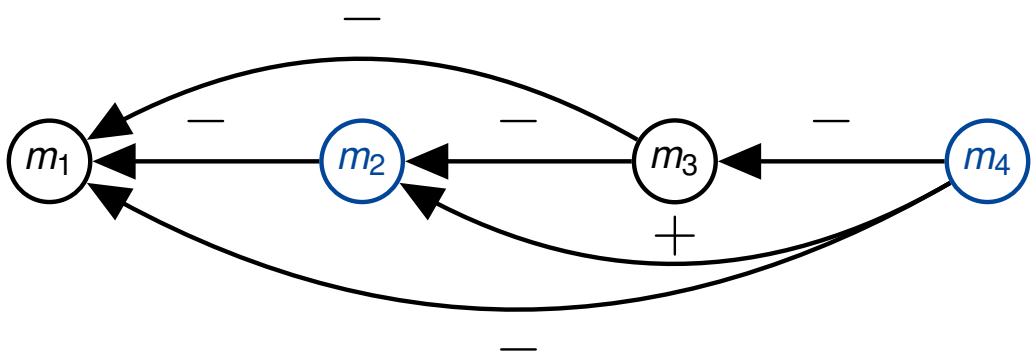
Latent structure

Mention Pair



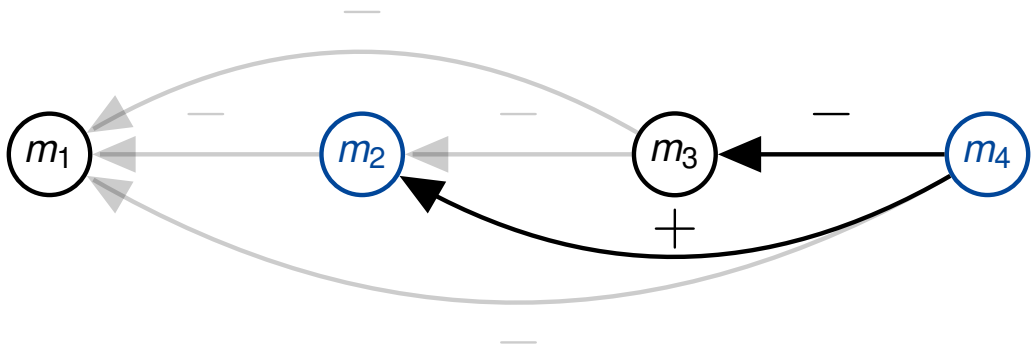
Substructure

Mention Pair



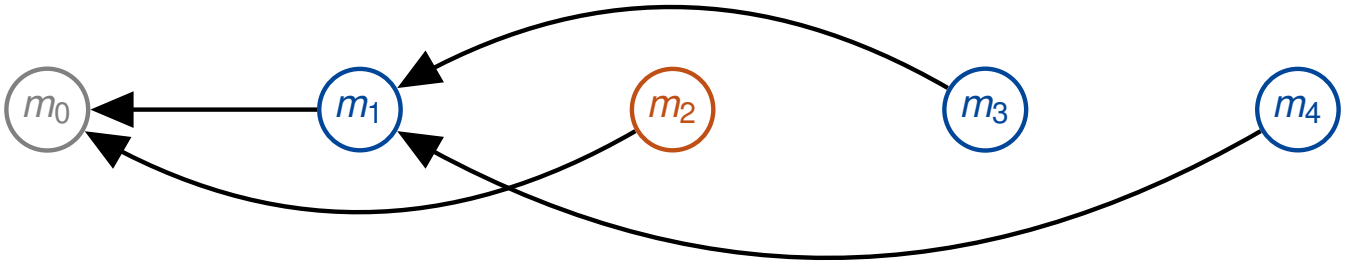
No costs (use training data resampling)

Mention Pair



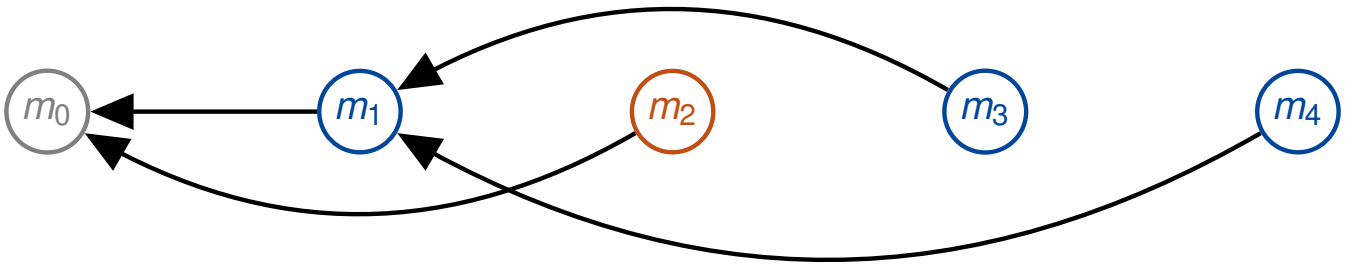
No costs (use training data resampling)

Mention Ranking



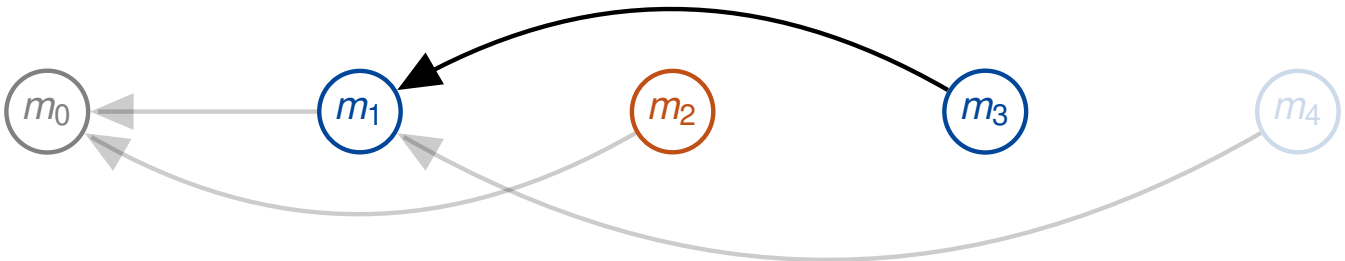
Denis and Baldridge (2008), Chang et al. (2013), ...

Mention Ranking



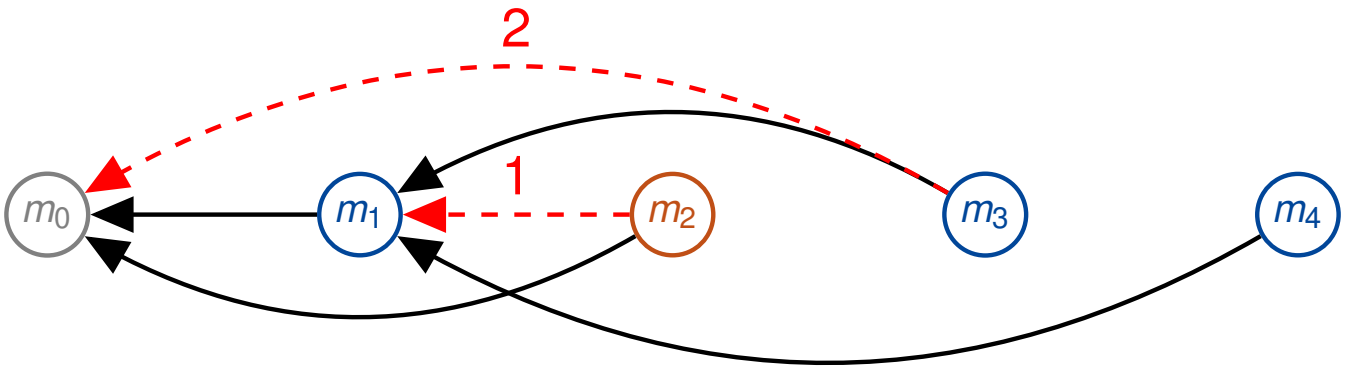
Latent structure

Mention Ranking



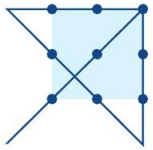
Substructure

Mention Ranking



Cost function (Durrett and Klein, 2013; Fernandes et al., 2014)

Outline



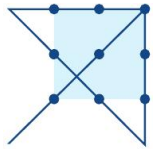
Motivation

Structures for Coreference Resolution

Experiments and Analysis

Conclusions and Future Work

Data



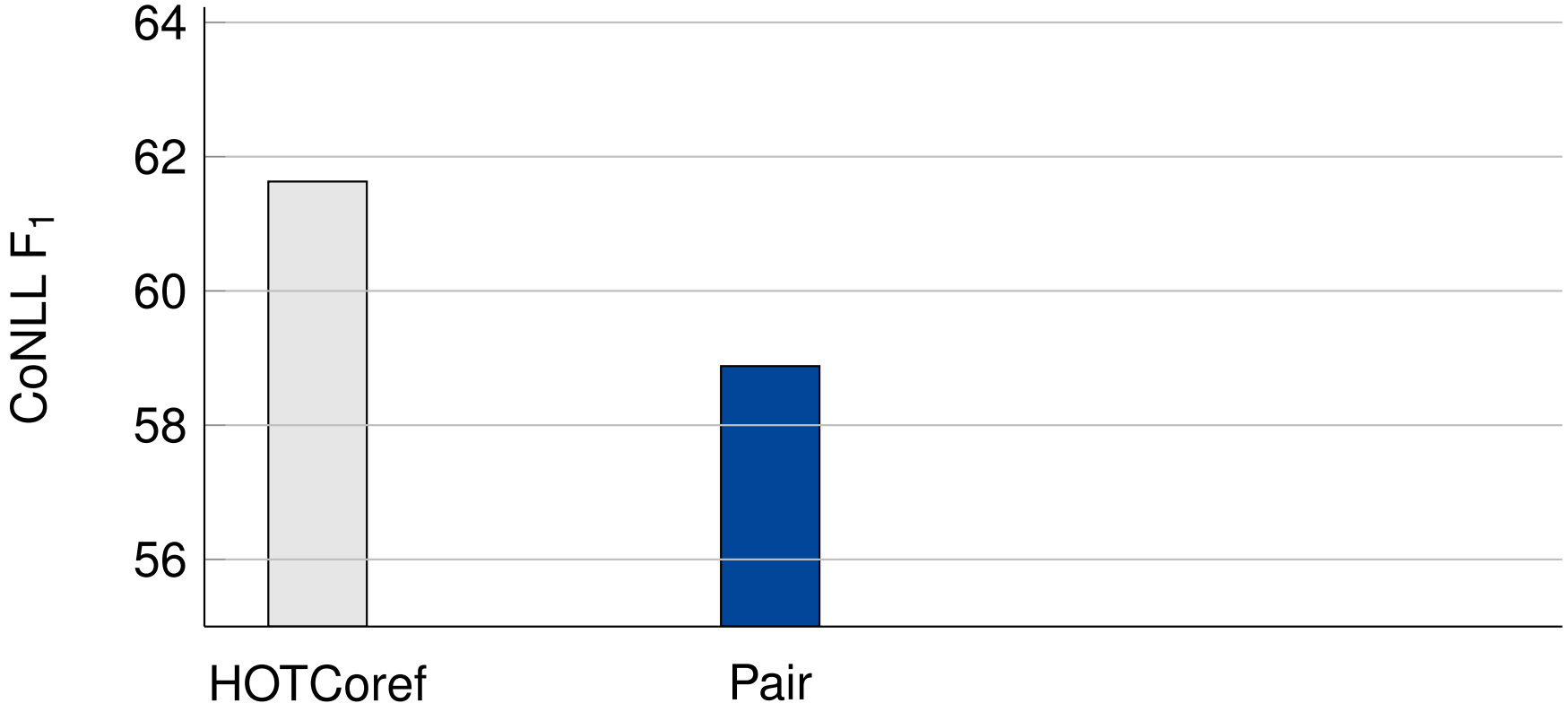
- conduct analysis and experiments on the English data from the CoNLL-2012 shared task on multilingual coreference resolution
- evaluate via CoNLL scorer (average of three widely used evaluation metrics)

Results on Test Data

Results on Test Data

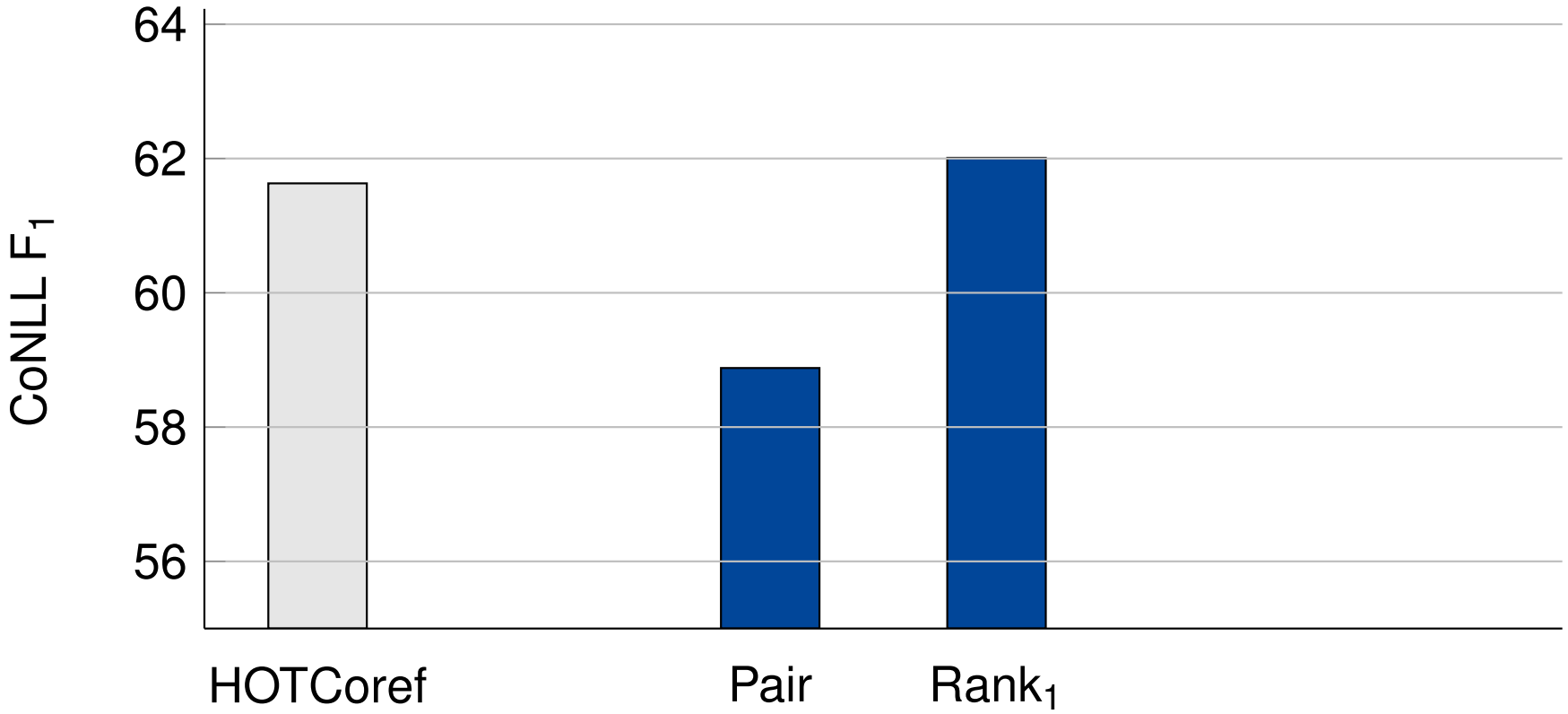
- state-of-the-art system based on antecedent trees with non-local features (Björkelund and Kuhn, 2014)

Results on Test Data



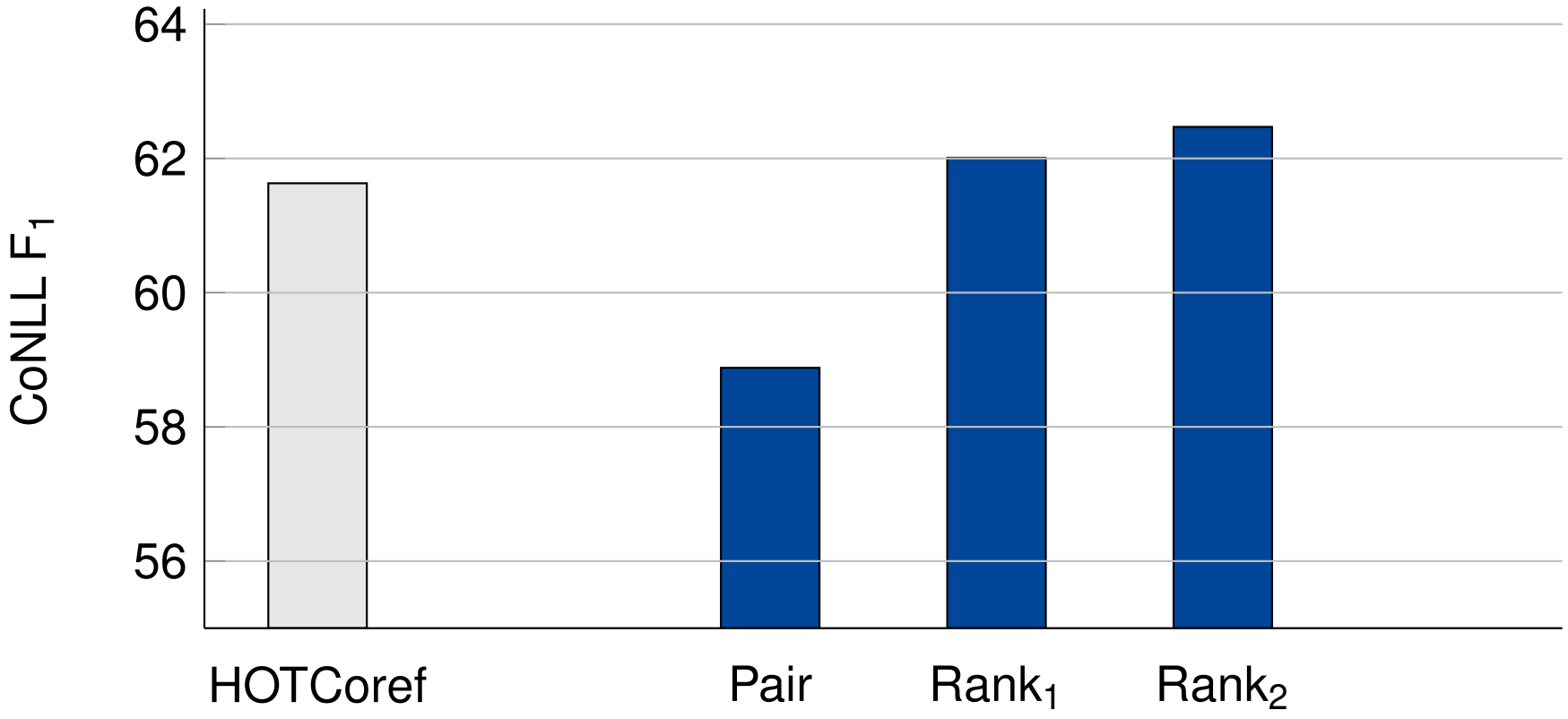
- mention pair model with standard strategy for training data balancing
- rich lexical feature set

Results on Test Data



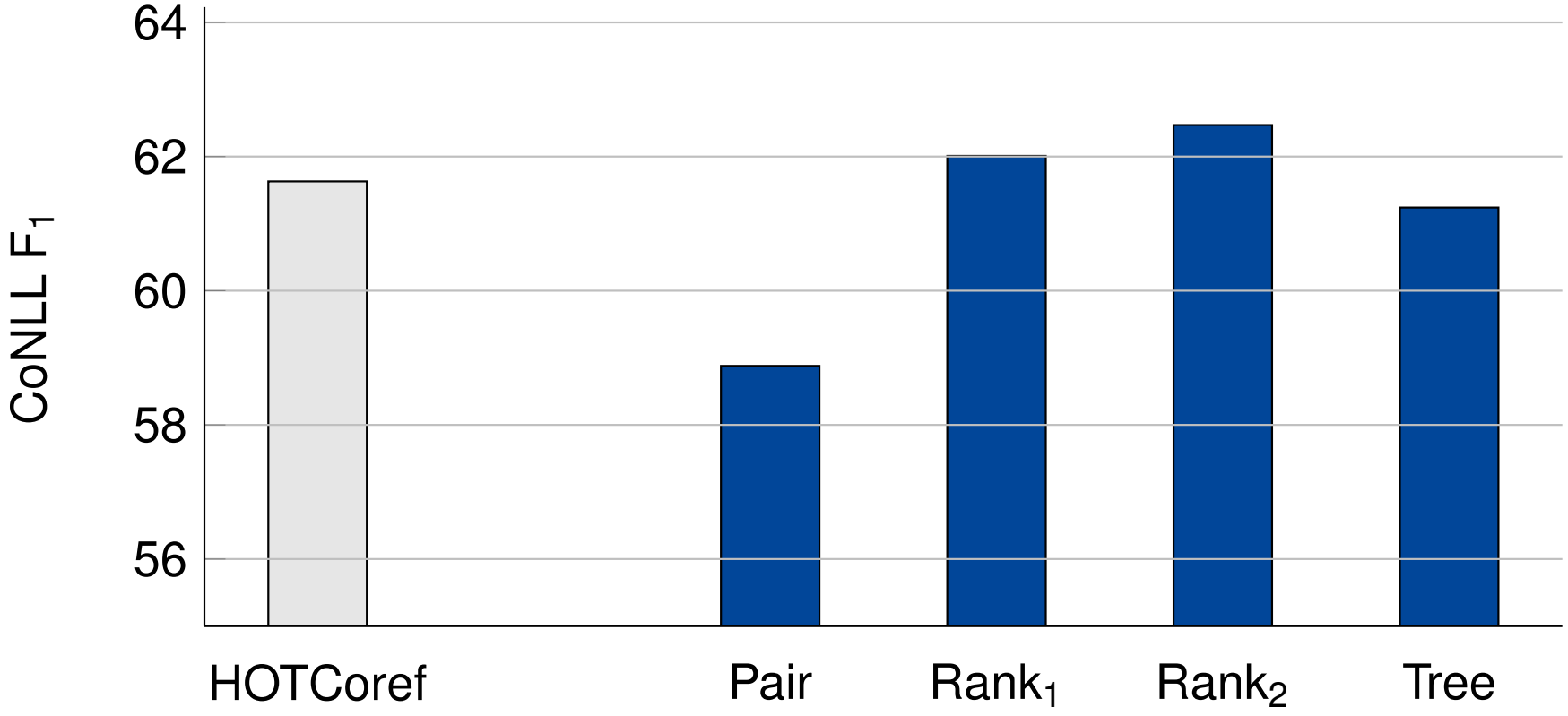
- ranking with closest antecedent as gold antecedent
- mainly gains in precision

Results on Test Data



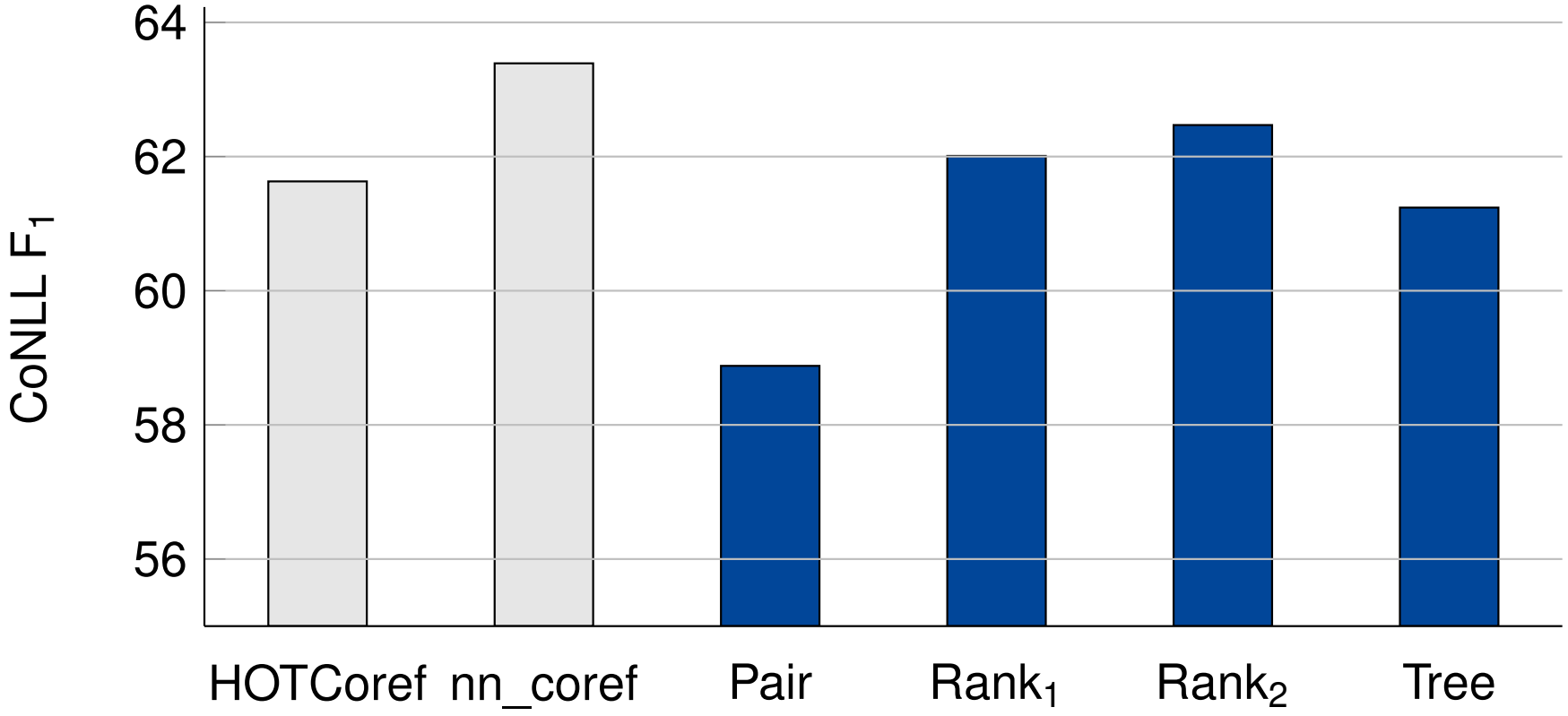
- ranking with latent antecedent as gold antecedent
- slight gains in precision and recall

Results on Test Data



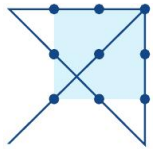
- antecedent trees
- higher precision, but lower recall

Results on Test Data



- mention ranking, feature combinations learned via neural networks (Wiseman et al., 2015)

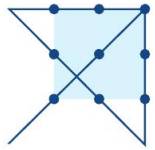
Analysis Tools



Employ our coreference resolution error analysis framework (Martschat and Strube, EMNLP 2014)

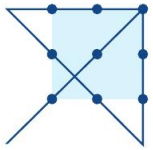
- extract precision and recall errors on development data
- compare errors made to assess strengths and weaknesses of approaches

Analysis



- ranking vs mention pair
  - mainly better anaphoricity determination
  - antecedent competition useful for pronouns
- latent ranking vs ranking with closest antecedents
  - mainly less precision errors for hard cases
- antecedent trees vs ranking
  - document-level modeling: more cautious updates $\rightarrow$ higher precision at expense of recall

Outline



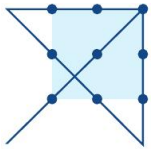
Motivation

Structures for Coreference Resolution

Experiments and Analysis

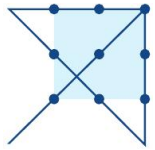
Conclusions and Future Work

Conclusions



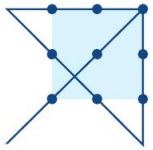
- coreference resolution approaches can be represented by latent structures they operate on
- devised a framework and implemented mention pair, mention ranking, antecedent trees
- mention ranking performs best, mainly due to anaphoricity modeling

Future Work



- apply framework to entity-centric approaches
- analyze more approaches
- devise new models in the framework

Thanks!

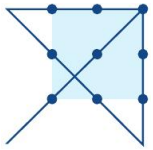


Python implementation, state-of-the-art models, tutorials available at:

`http://github.com/smartschat/cort`

This work has been funded by the Klaus Tschira Foundation.

Thanks!



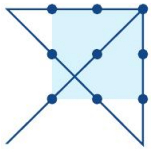
Python implementation, state-of-the-art models, tutorials available at:

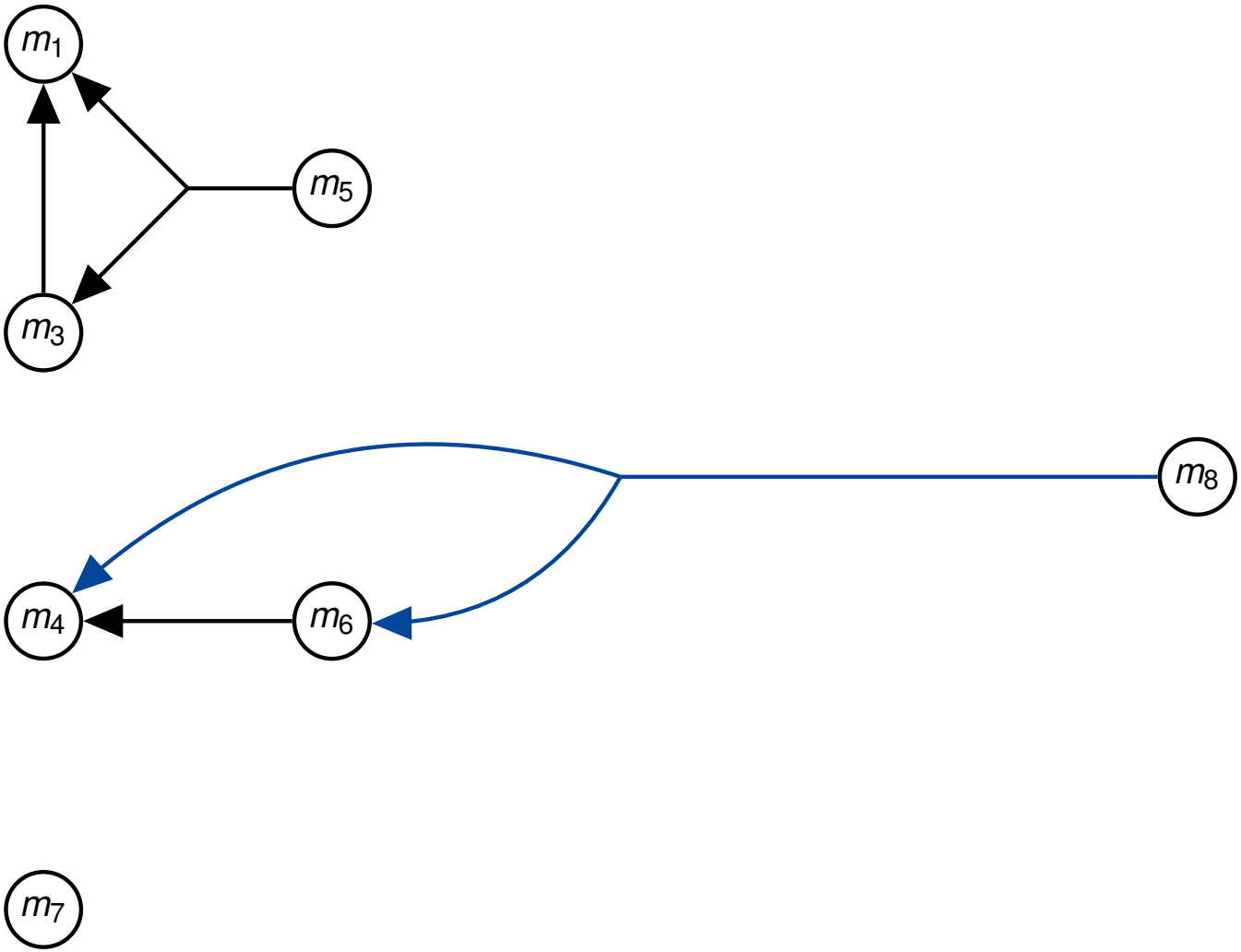
`http://github.com/smartschat/cort`

This work has been funded by the Klaus Tschira Foundation.

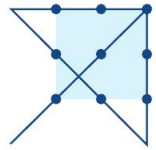
Thank you for your attention!

Entity-centric Approaches



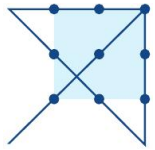


All Results



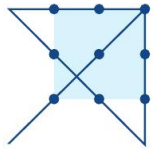
	MUC			B^3			$CEAF_e$			
Model	**R**	**P**	**F_1**	**R**	**P**	**F_1**	**R**	**P**	**F_1**	**Avg**
CoNLL-2012 English development data										
Pair	66.68	71.71	69.10	53.57	62.44	57.67	52.56	53.87	53.21	59.99
$Rank_1$	67.85	76.66	71.99*	55.33	65.45	59.97*	53.16	61.28	56.93*	62.96
$Rank_2$	68.02	76.73	**72.11**$^{\diamond\times}$	55.61	66.91	**60.74**$^{\dagger\diamond}$	54.48	61.36	**57.72**$^{\dagger\diamond\times}$	**63.52**
Tree	65.91	77.92	71.41	52.72	67.98	59.39	52.13	60.82	56.14	62.31
CoNLL-2012 English test data										
Pair	67.16	71.48	69.25	51.97	60.55	55.93	51.02	51.89	51.45	58.88
$Rank_1$	67.96	76.61	72.03*	54.07	64.98	59.03*	51.45	59.02	54.97*	62.01
$Rank_2$	68.13	76.72	**72.17**$^{\diamond}$	54.22	66.12	**59.58**$^{\dagger\diamond}$	52.33	59.47	**55.67**$^{\dagger\diamond}$	**62.47**
Tree	65.79	78.04	71.39	50.92	67.76	58.15	50.55	58.34	54.17	61.24

Analysis: Recall Errors



	Name/noun			Anaphor pronoun			
Model	**Both name**	**Mixed**	**Both noun**	**I/you/we**	**he/she**	**it/they**	**Rem.**
Max	3579	948	2063	2967	1990	2471	591
Pair	815	657	1074	394	373	1005	549
Rank_1	879	637	1221	348	247	806	557
Rank_2	857	647	1158	370	251	822	566
Tree	911	686	1258	441	247	863	572

Analysis: Precision Errors



	Name/noun			Anaphor pronoun			
Model	**Both name**	**Mixed**	**Both noun**	**I/you/we**	**he/she**	**it/they**	**Rem.**
Pair	885	83	1055	836	289	864	175
	2673	79	1098	2479	1546	1408	115
$Rank_1$	587	93	494	873	324	844	121
	2620	96	960	2521	1692	1510	97
$Rank_2$	640	92	567	862	318	835	42
	2664	102	1038	2461	1692	1594	43
Tree	595	57	442	836	318	757	37
	2628	82	924	2398	1691	1557	36