Higher-order Coreference Resolution with Coarse-to-fine Inference

Kenton Lee, Luheng He, Luke Zettlemoyer

NAACL
2018

정진우

Table of Contents

Coreference Resolution(상호참조해결)

- 하나의 개체(entity)를 표현하는 단어는 다양하다.

Ex) 별명, 약어, 대명사 등

- 동일한 의미를 가지지만 서로 다른 방식으로 표현되는 명사 및 명사구들을 찾아 서로 같은 개체로 연결해주는 자연어 처리 문제

- 참조 관계를 올바르게 찾아낼 수 있으면, 문서 내에서 언급하는 대상에 대한 정보를 일관성 있게 유지 및 전달 가능

Terminology

Terminology

1. Mention : 상호참조해결의 대상이 되는 모든 명사구(즉, 명사, 명사구 등)를 의미
2. Antecedent Mention : 특정 Mention과 동일한 의미를 가지면서 문서의 앞부분에 가장 가깝게 선행하는 Mention
3. Span : 한 개 이상의 단어로 이뤄진 Mention이 될 가능성이 있는 단어 또는 토큰들의 집합
4. Mention Cluster : 동일한 의미를 가진 명사, 명사구의 군집

유관순 열사가 의도적으로 띄워졌다는 견해도 있다. 하지만 이화학당 학당장 서리 월터는 "나는 그녀의 온전한 몸에다 수의를 입혔다"고 밝혔다.

Cluster : {(유관순, 그녀), (서리 월터, 나)}

Prior Model

Prior Model

Mention Detection -> Mention Clustering

위와 같은 과정을 생략하고 **End-to-End 딥러닝 모델로 해결**

End-to-end Neural Coreference Resolution(Kenton Lee et al, 2017) – EMNLP

Key idea :

1. 모든 span을 잠재적 mention으로 본다.
2. 각 mention에 대한 가능한 선행사(Antecedent)의 분포를 학습한다.

End-to-end Neural Coreference Resolution(Kenton Lee et al, 2017) – EMNLP

길이가 다른 모든 가능한 Span N개를 만든다.

하나의 다큐먼트에는 i개의 Span이 생성될 수 있다. (1 ≤ i ≤ N)

i번째 Span의 시작 단어인 START(i)에 근거하여 Span의 순서를 정한다.

동일한 START(i)를 가진 Span은 END(i)에 의해 순서가 정해지게 된다.

[[유관순] 열사]가 의도적으로 띄워졌다는 [견해]도 있다. 하지만 [[이화학당] [학당장] [서리 월터]]는 "[나]는 [[그녀]의 온전한 [몸]]에다 [수의]를 입혔다"고 밝혔다.

End-to-end Neural Coreference Resolution(Kenton Lee et al, 2017) – EMNLP

각 span i 에는 Antecedent span yi가 할당된다.

yi에 대해 가능한 Antecedent 집합은 $Y(i) = \{\epsilon , 1 , . . . i - 1\}$

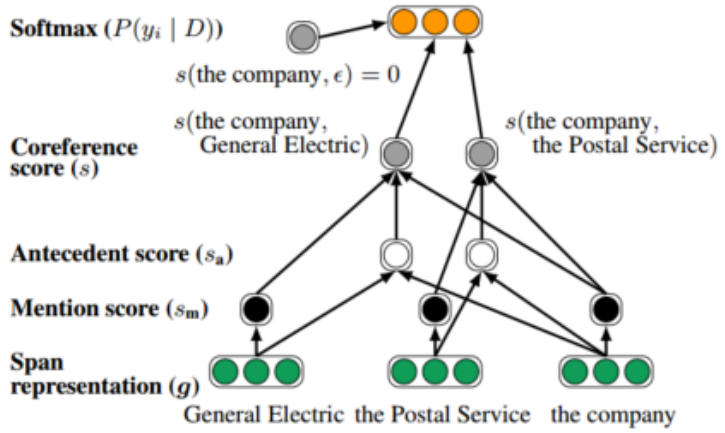
ϵ는 더미 Antecedent이다.

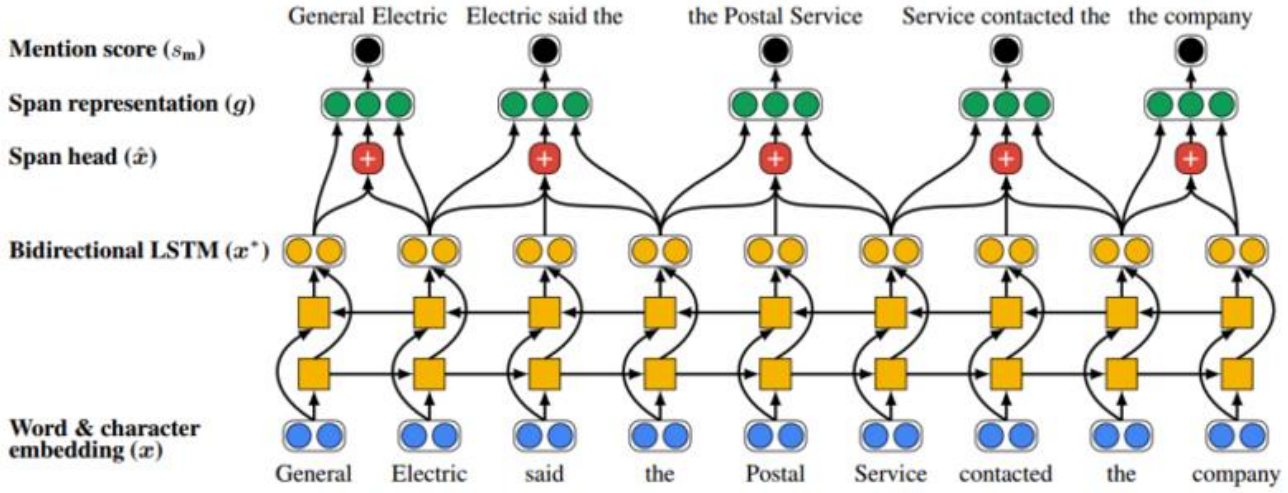
- 더미 Antecedent ϵ가 가능한 두 가지 시나리오는 :
  (1) span i는 mention이 아니다.
  (2) span i는 mention이지만 Antecedent가 없다.

span i의 TRUE Antecedent인 span j ($1 \leq j \leq i - 1$)는 i와 j 사이의 coreference link를 나타낸다.

[[유관순] 열사]가 의도적으로 띄워졌다는 [견해]도 있다. 하지만 [[이화학당] [학당장] [서리 월터]]는 "[나]는 [[그녀]의 온전한 [몸]]에다 [수의]를 입혔다"고 밝혔다.

End-to-end Neural Coreference Resolution(Kenton Lee et al, 2017) – EMNLP





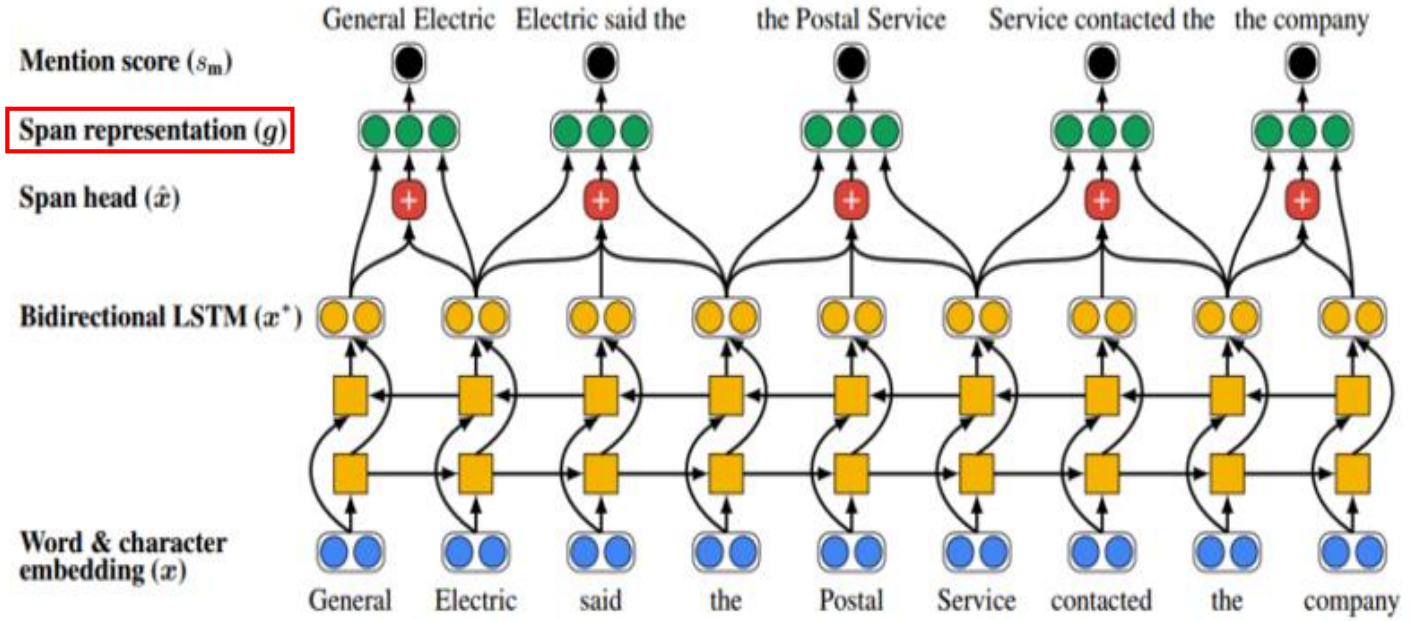
End-to-end Neural Coreference Resolution(Kenton Lee et al, 2017) – EMNLP

$$\alpha_t = \boldsymbol{w}_\alpha \cdot \text{FFNN}_\alpha(\boldsymbol{x}_t^*)$$

$$a_{i,t} = \frac{\exp(\alpha_t)}{\sum_{k=\text{START}(i)}^{\text{END}(i)} \exp(\alpha_k)}$$

$$\hat{\boldsymbol{x}}_i = \sum_{t=\text{START}(i)}^{\text{END}(i)} a_{i,t} \cdot \boldsymbol{x}_t$$

where $\hat{\boldsymbol{x}}_i$ is a weighted sum of word vectors in span i. The weights $a_{i,t}$ are automatically learned and correlate strongly with traditional definitions of head words as we will see in Section 9.2.

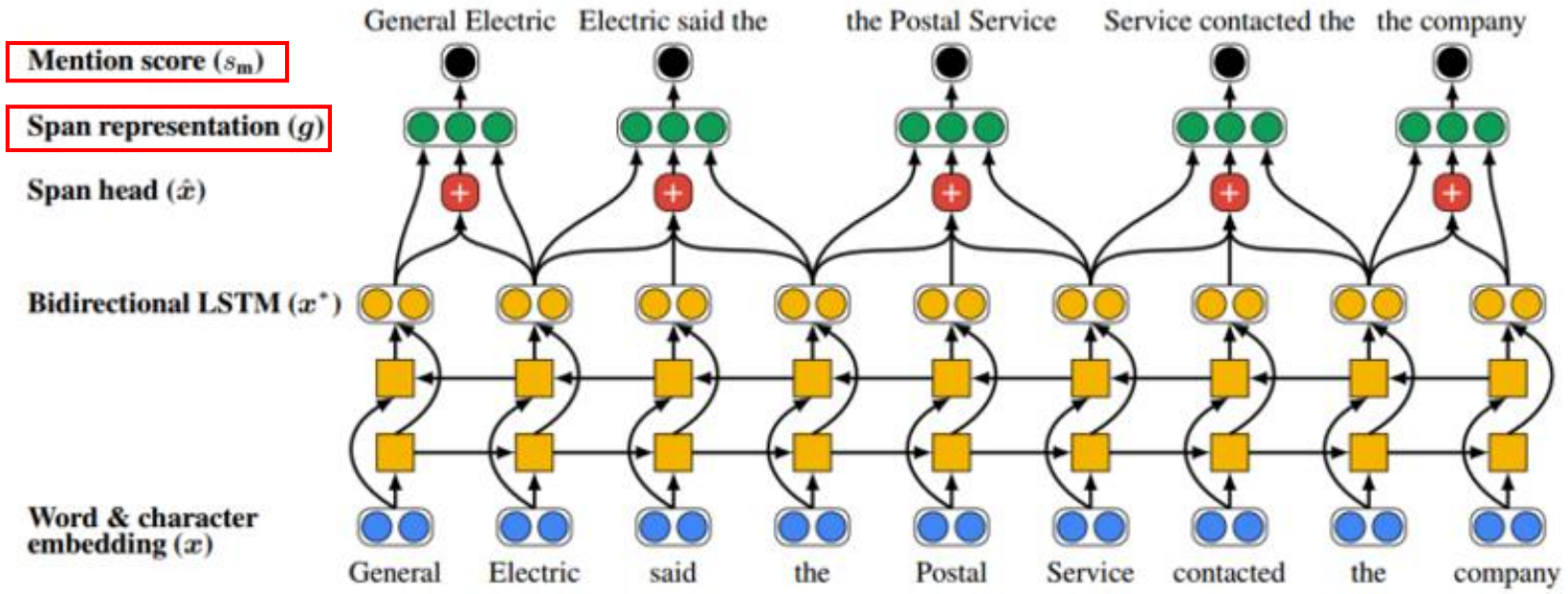


$$\boldsymbol{g}_i = [\boldsymbol{x}^*_{\text{START}(i)}, \boldsymbol{x}^*_{\text{END}(i)}, \hat{\boldsymbol{x}}_i, \phi(i)]$$

End-to-end Neural Coreference Resolution(Kenton Lee et al, 2017) – EMNLP

$$s_{\mathrm{m}}(i) = \boldsymbol{w}_{\mathrm{m}} \cdot \mathrm{FFNN}_{\mathrm{m}}(\boldsymbol{g}_i)$$

$$\boldsymbol{g}_i = [\boldsymbol{x}^*_{\mathrm{START}(i)}, \boldsymbol{x}^*_{\mathrm{END}(i)}, \hat{\boldsymbol{x}}_i, \phi(i)]$$



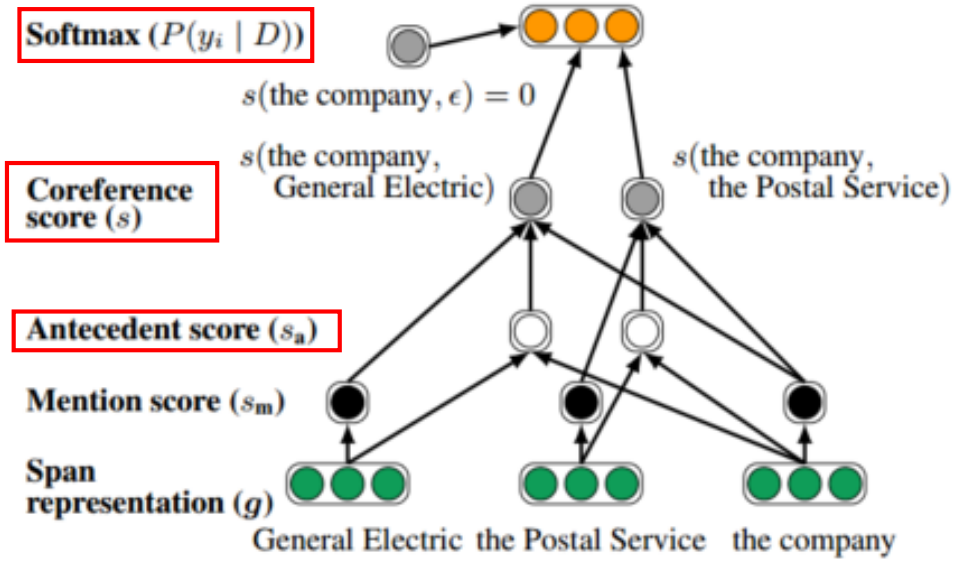
mention score를 계산하고, 이 score가 낮은 span에 대한 프루닝을 진행
(top λ*N을 이용하는데, 여기서 람다는 하이퍼 파라미터)

End-to-end Neural Coreference Resolution(Kenton Lee et al, 2017) – EMNLP

$$s(i,j) = \begin{cases} 0 & j = \epsilon \\ s_{\mathrm{m}}(i) + s_{\mathrm{m}}(j) + s_{\mathrm{a}}(i,j) & j \neq \epsilon \end{cases}$$

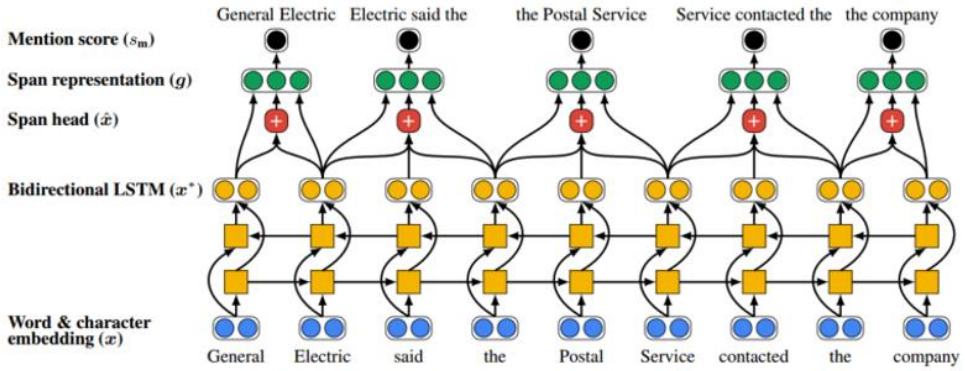
$$s_{\mathrm{a}}(i,j) = \boldsymbol{w}_{\mathrm{a}} \cdot \mathrm{FFNN}_{\mathrm{a}}([\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_i \circ \boldsymbol{g}_j, \phi(i,j)])$$

Antecedent score 계산 전 거리에 따른 프루닝 진행



$$P(y_1, \ldots, y_N \mid D) = \prod_{i=1}^{N} P(y_i \mid D) = \prod_{i=1}^{N} \frac{\exp(s(i, y_i))}{\sum_{y' \in \mathcal{Y}(i)} \exp(s(i, y'))}$$

$$\log \prod_{i=1}^{N} \sum_{\hat{y} \in \mathcal{Y}(i) \cap \mathrm{GOLD}(i)} P(\hat{y})$$



GOLD(i) = {ε, i-3, i-2, i-1}

*Gold Cluster = 실제 Antecedent 집합

Model Framework

Problem of Prior Model

이전 모델에서 mention pair 사이의 coreference link는 다른 link들의 영향을 받지 않고 독립적으로 결정되었다.

-> 모델이 반환하는 cluster가 locally consistent 하지만 globally inconsistent하다는 문제 발생

** 학습할 때에는 GOLD CLUSTER에 맞춰 학습하지만 실제 클러스터를 뽑아보면 이러한 문제가 발생

Speaker 1: Um and **[I]** think that is what's - Go ahead Linda.
Speaker 2: Well and uh thanks goes to **[you]** and to the media to help us... So our hat is off to **[all of you]** as well.

Figure 1: Example of consistency errors to which first-order span-ranking models are susceptible. Span pairs (**I**, **you**) and (**you**, **all of you**) are locally consistent, but the span triplet (**I**, **you**, **all of you**) is globally inconsistent. Avoiding this error requires modeling higher-order structures.

[you] 와 [all of you]를 묶어야 하는가? Locally Yes / Globally No

이는[I]와 [you]의 링크를 보면 알 수 있다.

Problem of Prior Model

이전 모델에서는 휴리스틱하게 span i와 가장 가까운 K개의 Antecedent만 고려하여 계산량 줄임

하지만 이는 매우 긴 문서에 대해 성능 저하

Model

1. 이전 논문의 모델을 베이스라인으로 이용하되 이러한 문제를 해결하기 위한 higher-order inference(고차 추론)를 제시한다.

- 매 iteration 마다 antecedent distribution을 어텐션으로 이용하고, 이번의 coreference decision은 다음 번의 coreference decision에 영향을 주도록 하는 것이다. 예를 들어 [I, you] 의 coreference link가 [you, all of you] 의 coreference link에 영향을 미치는 것이다.

2. 모델의 computational challenge를 줄이기 위해 coarse-to-fine 방식을 제시하며 이는 end-to-end 모델을 학습하는 과정에서 같이 학습될 것이다. 이를 이용하면 프루닝이 더 많아지고 계산할 Antecedents의 수를 줄일 수 있다.

Model

Higher Order inference의 핵심

이전의 결정이 다음의 결정에 영향을 미치도록 만들자!

n번의 iteration을 통해 Span i 와 Antecedents 사이의 확률분포를 생성하고 이를 attention으로 이용하는 것

* 이 논문에서는 n = 2 일 때, 가장 성능이 좋다고 한다.

Model

Prior model : $$P(y_1, \ldots, y_N \mid D) = \prod_{i=1}^{N} P(y_i \mid D)$$

$$= \prod_{i=1}^{N} \frac{\exp(s(i, y_i))}{\sum_{y' \in \mathcal{Y}(i)} \exp(s(i, y'))}$$

attention mechanism은 아래와 같다:

$$P_n(y_i) = \frac{e^{s(\boldsymbol{g}_i^n, \boldsymbol{g}_{y_i}^n)}}{\sum_{y \in \mathcal{Y}(i)} e^{s(\boldsymbol{g}_i^n, \boldsymbol{g}_y^n))}}$$

$$\boldsymbol{a}_i^n = \sum_{y_i \in \mathcal{Y}(i)} P_n(y_i) \cdot \boldsymbol{g}_{y_i}^n$$

attention mechanism에서 사용될 값이 확률 분포에 기반하여 생성된다는 것을 확인할 수 있다.

Model

- span representation은 다음과 같이 업데이트 된다.

The current span representation $\boldsymbol{g}_i^n$ is then updated via interpolation with its expected antecedent representation $\boldsymbol{a}_i^n$:

$$\boldsymbol{f}_i^n = \sigma(\mathbf{W}_{\mathrm{f}}[\boldsymbol{g}_i^n, \boldsymbol{a}_i^n]) \quad (7)$$

$$\boldsymbol{g}_i^{n+1} = \boldsymbol{f}_i^n \circ \boldsymbol{g}_i^n + (\mathbf{1} - \boldsymbol{f}_i^n) \circ \boldsymbol{a}_i^n \quad (8)$$

- 이 때 학습된 gate vector fni는 현재 span information을 얼마나 유지하고, 얼마나 새로운 정보를 융합할지를 의미한다.

  즉, span i에 대해 가질 수 있는 antecedent들에 대한 ratio를 학습하는 것이다. 이 때 ratio는 antecedent가 높은 확률(점수)를 가질 경우 높게 계산되어 해당 span representation에 더해지게 된다. 이후, 점차 antecedent들의 정보와 현재 span의 정보들이 융합되어 새로운 span representation을 만드는 것이다.

Model

Coarse-to-fine의 핵심

- 이러한 모델의 문제점은 바로 학습시간
- Antecedent들을 pruning하는 방법이 필요

- 본 논문에서는 이 문제를 coarse-to-fine 방식을 통해 해결
  우선, span i 와 이의 antecedent인 span j에 대해 간단한 score function를 만듦

$$s_c(i,j) = \boldsymbol{g}_i^\top \mathbf{W}_c \, \boldsymbol{g}_j$$

- Wc는 학습된 weight matrix로, 이는 span i와 span j를 단순 비교할 수 있도록 하는 함수

Similarity를 이용해 pruning을 하는 것이라고 생각하면 편할 것

Model

Sc(i,j)는 likely antecedent들을 계산하고 three stage pruning step으로서 이용

1) Sm(i)에 기반하여 top M개의 span을 추출

$$s_{\mathrm{m}}(i) = \boldsymbol{w}_{\mathrm{m}} \cdot \mathrm{FFNN}_{\mathrm{m}}(\boldsymbol{g}_i)$$

2) Sm(i) + Sm(j) + Sc(i,j)에 기반하여 span i에 대해 top K개의 Antecedent들을 추출한다. 주의할 점은 Sa(i,j)가 아니라 간단한 상호참조점수 함수를 통해 추출한다는 것

$$s_{\mathrm{a}}(i,j) = \boldsymbol{w}_{\mathrm{a}} \cdot \mathrm{FFNN}_{\mathrm{a}}([\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_i \circ \boldsymbol{g}_j, \phi(i,j)])$$

$$s_{\mathrm{c}}(i,j) = \boldsymbol{g}_i^{\top} \mathbf{W}_{\mathrm{c}}\, \boldsymbol{g}_j$$

3) 이후 남아있는 span pairs에 대해 S(i,j)를 계산한다.

$$s(i,j) = s_{\mathrm{m}}(i) + s_{\mathrm{m}}(j) + s_{\mathrm{c}}(i,j) + s_{\mathrm{a}}(i,j)$$

Experiment & Result

	MUC			B^3			$CEAF_{\phi_4}$			
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Avg. F1
Martschat and Strube (2015)	76.7	68.1	72.2	66.1	54.2	59.6	59.5	52.3	55.7	62.5
Clark and Manning (2015)	76.1	69.4	72.6	65.6	56.0	60.4	59.4	53.0	56.0	63.0
Wiseman et al. (2015)	76.2	69.3	72.6	66.2	55.8	60.5	59.4	54.9	57.1	63.4
Wiseman et al. (2016)	77.5	69.8	73.4	66.8	57.0	61.5	62.1	53.9	57.7	64.2
Clark and Manning (2016b)	79.9	69.3	74.2	71.0	56.5	63.0	63.8	54.3	58.7	65.3
Clark and Manning (2016a)	79.2	70.4	74.6	69.9	58.0	63.4	63.5	55.5	59.2	65.7
Lee et al. (2017)	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
+ ELMo (Peters et al., 2018)	80.1	77.2	78.6	69.8	66.5	68.1	66.4	62.9	64.6	70.4
+ hyperparameter tuning	80.7	78.8	79.8	71.7	68.7	70.2	67.2	66.8	67.0	72.3
+ coarse-to-fine inference	80.4	**79.9**	80.1	71.0	**70.0**	70.5	67.5	**67.2**	67.3	72.6
+ second-order inference	**81.4**	79.5	**80.4**	**72.2**	69.5	**70.8**	**68.2**	67.1	**67.6**	**73.0**