<http://dsba.korea.ac.kr/seminar/?mod=document&uid=42>

<https://gnoej671.tistory.com/category/NLP%20/CS224n>

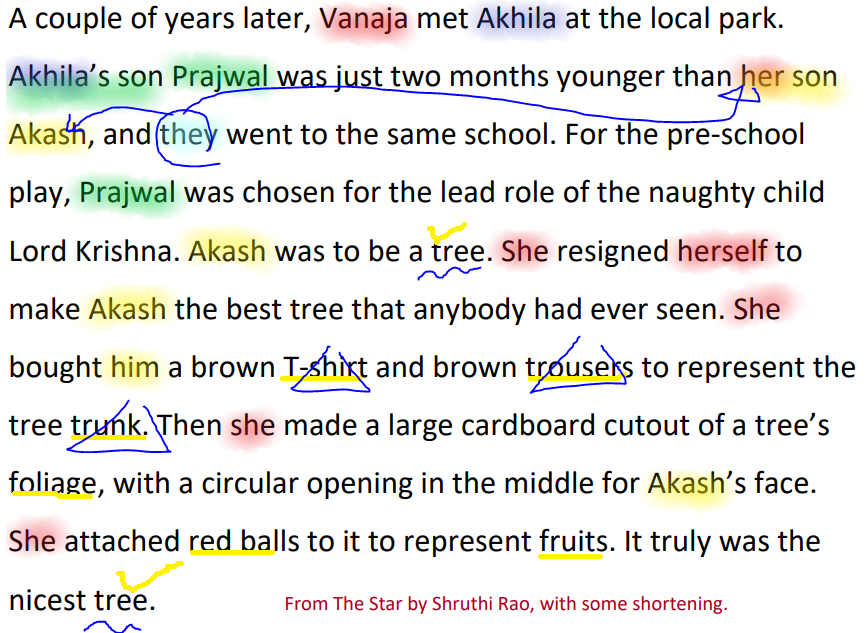
<https://happyzipsa.tistory.com/13?category=748723>

Coreference(동일 지시어) 란 텍스트에서 Real World Entities를 모두 찾아내는 것이다.

여기서 Entity란 고유명사가 될 수도 있고 고유 명사를 가리키는 대명사 or 일반 명사도 될 수 있다. 따라서 쉬운 문제는 아니다.(문맥을 알아야 이 대명사 or 일반 명사가 어떤 고유 명사를 가리키는지 알 수 있다.)

그리고 어떤 단어가 entity인지 구분하는 것도 사실 어렵다.

Coreference 풀 때 1. 고유 명사 찾기, 2. 고유 명사를 가리키는 일반명사 or 대명사 찾기 인데 실제로는 전자가 더 어렵다고 한다.(의외)



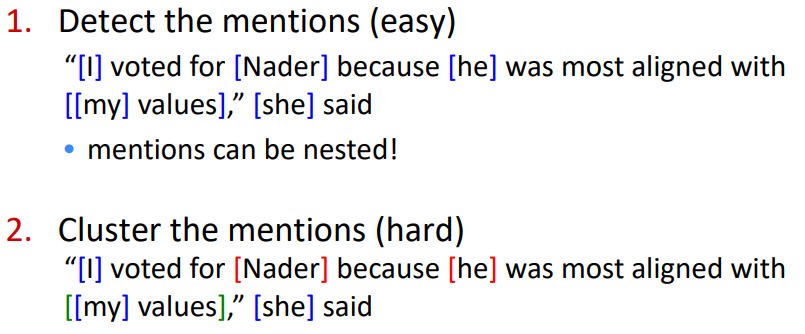
처럼 They는 앞의 2개 이상의 entity를 포함하고 있다. 이 경우 split antecedents라고 한다. NLP에서는 이러한 split antecedents를 처리할 수 있는 기술이 없다.

여기서 tree는 Akash를 나타내고 있는데 어떤 시스템은 같은 entity로 분류하고 그렇지 않은 시스템도 있다.

또한 tree의 부분을 나타내는 단어로 T-shirt, trousers, trunk가 있는데 같은 entity로 표현하는가? 도 또한 이슈이다.(주로 다른 단어로 구분)

Coreference Resolution을 다른 task를 푸는데 같이 사용하면 다른 task에 대한 성능이 향상 된다.(QA, Summarization, extracting fact , MT, understanding, Dialogue ...)  
ex) “he was born in 1961” 에서 he가 누구를 나타내는지 알 수 있고 번역에 도움이 될 수 있다.  
ex) Spanish 에서는 subject가 가끔 생략되는데 Coreference decision을 이용해 subject를 알아서 끼워 넣었다.

Coreference Resolution은 두 단계로 나뉜다.



모든 mentions(entity라 생각되는 것)을 찾고 그것을 클러스터링 한다.

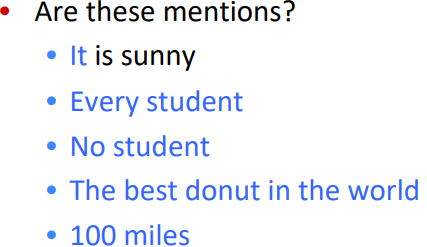
이때 mention은 크게 3가지로 분류하는데

1. Pronouns: I, your, it, she, him, etc
2. Named entities: People, places, etc (일반 명사와 더불어 고유 명사도 포함인듯)
3. Noun phrases: “a dog”, “the big fluffy cat stuck in the tree”(일반 명사들을 합친 구)

일단 mention을 찾기 위해서는 다른 NLP 시스템을 이용한다.

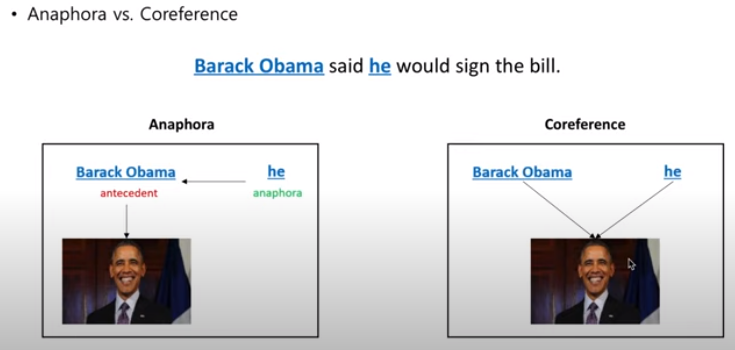
1. Pronouns: POS tagger
2. Named entities: NER(like hw3)
3. Noun phrases: parser(constituency parser)

아래처럼 비인칭 주어, 불특정 다수, 측정 단위 등 엔티티라고 명하기에 애매한 것이 있다.(위에 말한 3가지 조건을 만족하더라도) 이것을 다룰 수 있는 방법이 있나?



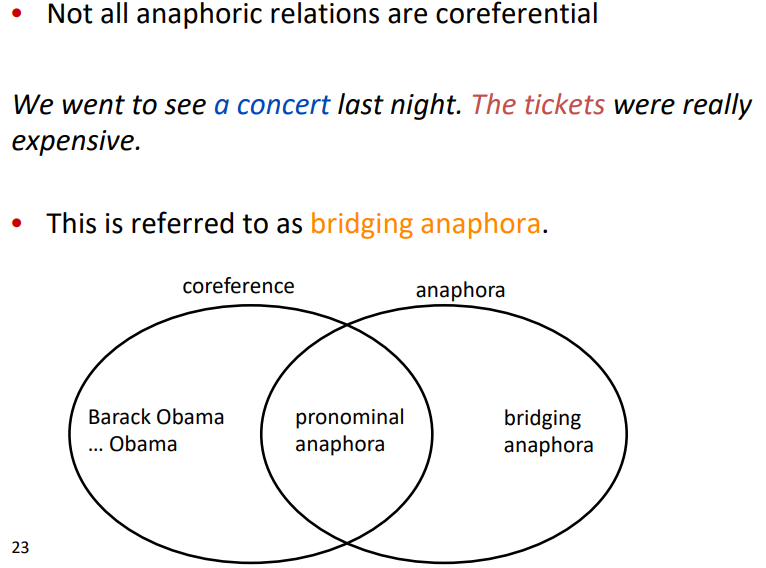
또한 지금까지는 2개의 pipe line을 거쳐서 진행했지만 최근에는(2016~) POS Tagger, NER, parser를 따로 이용하는 것이 아니라 detection + clustering(coreference resolution)을 한번에 할 수 있는 end-to-end classifier를 학습시키는 추세이다.

“Barack Obama said he would sign the bill” 에서 anaphora 는 he 이고 antecedent 는 선행사로 Barack Obama가 된다. Anaphora는 선행사가 없다면 해석을 잘 할 수 없다.



이렇게 Anaphora로 나뉘었을 때 생각하면 는 오바마를 가리키지 않지만 Coreference로 여긴다면 오바마를 가리키게 된다.

즉, 문장을 이해하기 위해서는 Coreference 방식, Anaphora 방식이 있는 것이고 Coreference는 mention들이 독립적으로 같은 엔티티를 가리키며, anaphora는 독립적으로 다른 엔티티를 가리키며 선행 단어에 의존하거나(bridging anaphora) 의존적으로 같은 엔티티를 가리키며 앞에 antecedent에만 의존할 수도 있다.



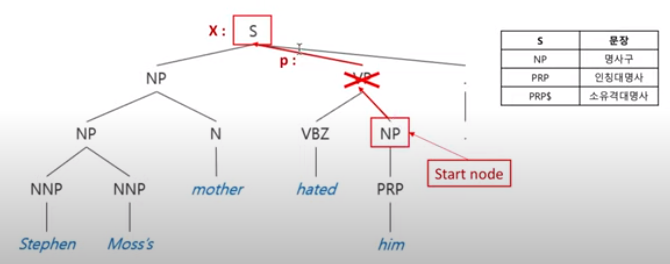
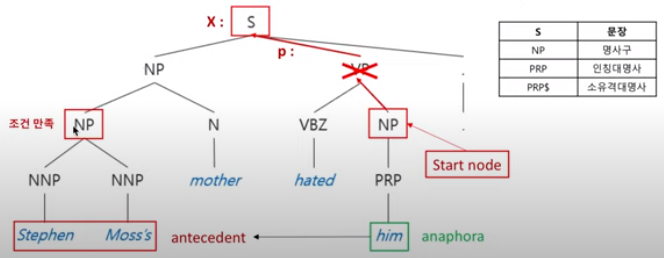
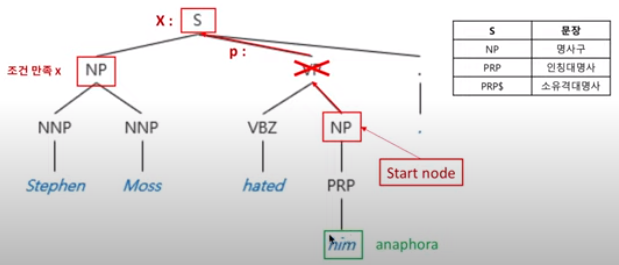
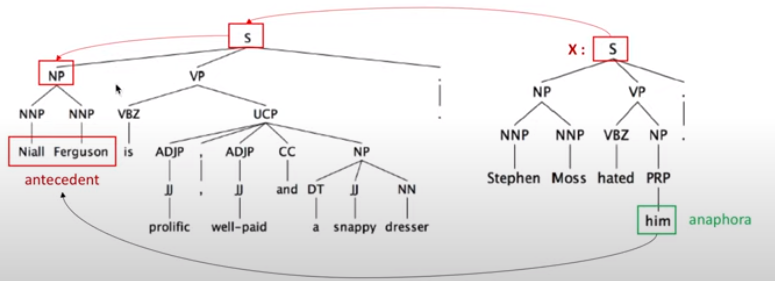
Cataphora는 뒤의 antecedent를 참조해서 그 의미를 파악하는 것이다.

“After he had received his orders, the soldier ~”

여기까지 서론이였고 이제 모델 및 알고리즘 알아볼 것이다.

1. Rule-based Model – Hobb’s naïve algorithm(1976)

전통적인 pronominal anaphora resolution! 딥러닝 전 ML 시스템의 피처, 9개의 스텝을 반복하면서 antecedent를 찾는 과정 반복. Coreference resolution을 위한 완벽한 알고리즘은 아니지만 baseline을 구축

1. 대명사를 포함하고 있는 NP를 시작 노드로 설정  
   
2. 시작 노드에서부터 위쪽으로 이동하며 가장 먼저 등장하는 NP or S로 이동. 이때 그 노드는 X로 칭하고, path를 p라 한다.
3. X를 기준으로 BFS 방식으로 X와 어떤 NP 노드 사이에 S OR NP 노드를 부모로 가진다면 어떤 NP 노드를 선행사로 판단하는 수학적 알고리즘이다.  
   
4. 만약 트리가 더 크다면 이런 구조가 여러 번 있을 터이니 이 로직을 반복해준다.(위 예시와 달리 한 번에 가장 상위 S노드에 도달하지 못했을 때)
5. 하지만 아래와 같은 경우 선행사가 없을 수 있다. -> 이전 문장에서 1)~3)을 실행 한다.  
   아래 그림처럼 하면 된다. 만약 근데 X가 최상위 노드가 아니면 다시 4) 과정을 해주면 된다.  
     
   

그리고 역시나 rule-based다 보니까 허점이 매우 많다.  
에서 it이 가리키는게 다르지만 모두 다 컵을 향하게 된다. (Hobb’s algorithm이 해결하지 못하는 문제)

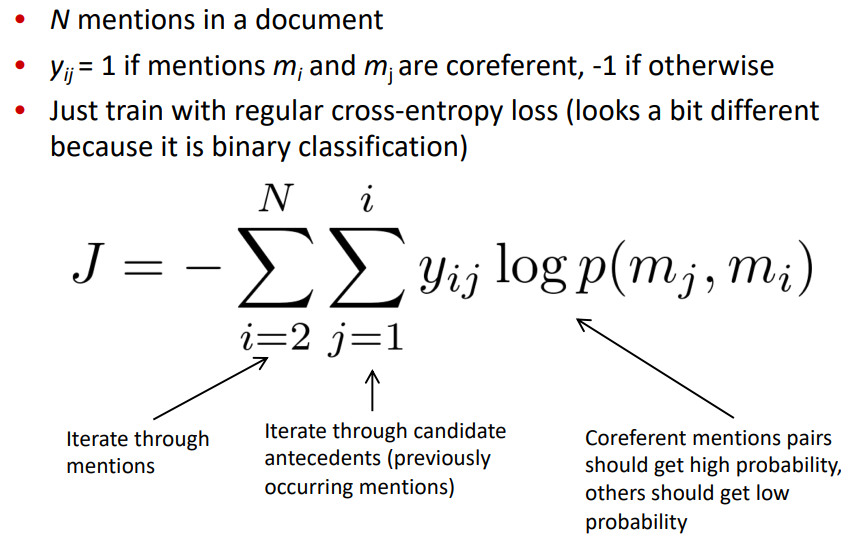
2. Mention Pair Model

문장을 왼쪽에서 오른쪽으로 훑되 새로운 mention을 발견할 때마다 앞의 이미 발견된 mention들과 비교를 하게 된다.

아래 그림처럼 i=2 인 이유는 맨 처음 mention에 대해서는 훑을 필요가 없기 때문이다.

p(m-j, m-i): 두 mention이 coreference일 확률

y-ij: coreferent 라면 1 아니라면 0 이다.

만약 coreferent 하지 않다면 p(m-j, m-i)가 0에 가까워 지도록 학습하고 coreferent 하다면 1에 가까워지도록 학습한다.  


삼단 법칙 같은 것(trasitive closure)을 이용해서 classifier에 all mention pairs를 넣어서 threshold 값 이상이면 coreference link로 연결한다. 이렇게 clustering이 가능해지며 mention이 어떤 단어와도 link가 없다면 singleton mention이 된다.

만약 성능이 안좋아 잘못 link가 이어질 경우 clustering이 급격히 커져 안 좋은 결과가 있을 수 있다.

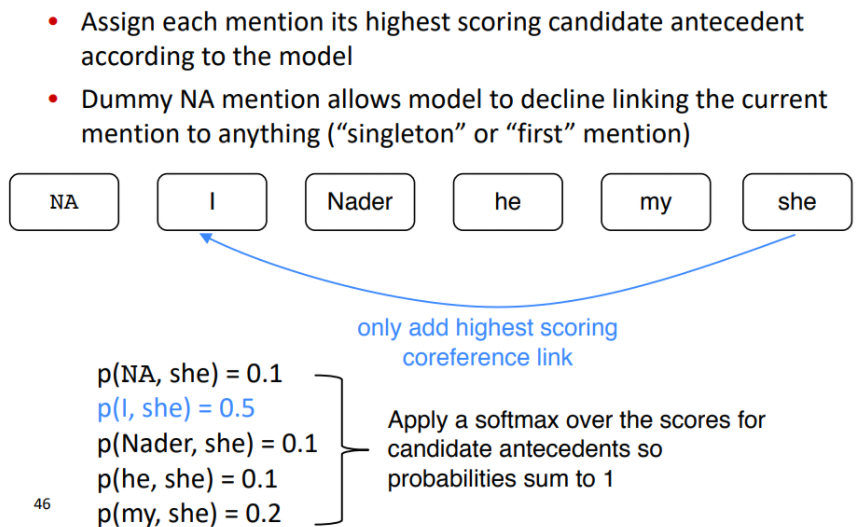
그런데 mention detecting은 어떻게 하는지 설명 안 나와있다.

하지만 단점으로 주로 antecedent는 앞이라 해도 가까운 앞에 있을 확률이 높다. 하지만 이 경우에서는 모든 pair에 대해 coreference를 계산하기 때문에 쓸모 없을 수 있는 계산이 많을 수 있다. -> 따라서 mention ranking model 을 하면은 한 개의 best 선행사를 찾으려 한다.

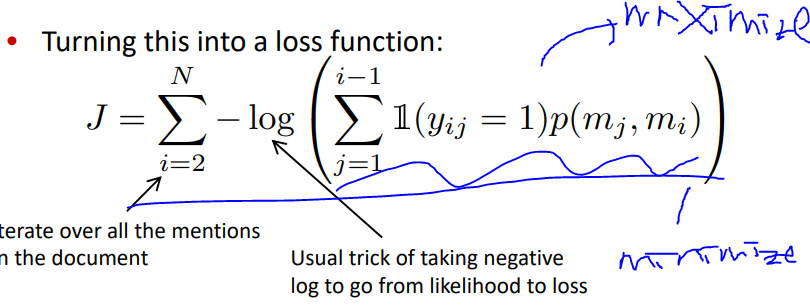
3. Mention Ranking Model

아래 그림의 경우 (she, i)와 (she, my)에 대한 coreferent 점수가 모두 높게 모델을 학습 시켜야 했지만 여기서는 softmax를 활용했기 때문에 1가지만 높으면 된다. 따라서 classifier는 가장 높은 값을 가진 mention 1개에 대해서만 coreference link를 가지게 된다.

참고로 NA는 singleton mention인 경우 필요한 가상의 선행사이다.



앞서 2. 의 모델의 loss function과 다르게 -log가 각 iteration에 붙는다. (앞서는 각 pair step마다 붙어 각 pair에서의 binary 문제를 해결할려 했다.) -> 각 iteration에서 앞 mentions 중 1개를 고르려는 컨셉과 맞다. 아래 항등함수가 추가되었다.(1 or 0 똑같음)



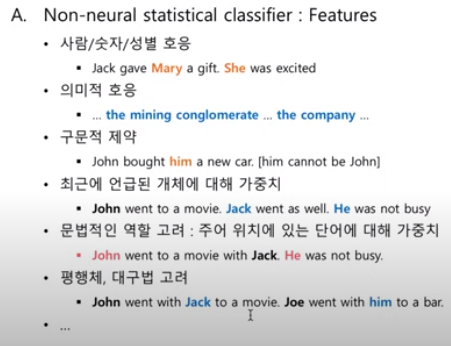
Test 시에는 softmax를 사용한다.(training 때는 사용 안하는 듯?) one best 선행사만 연결해줬는데도 mention-pair와 같은 결과 + 계산량 줄음 + 별도의 clustering이 없고 자연스럽게 진행

4. Clustering Model

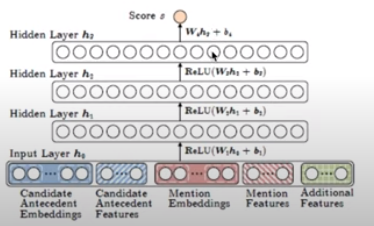
그럼 이제 여기서 중요한게 어떤 classifier를 이용해서 확률을 구할 것인가 가 이슈이다.

Non-neural statistical classifier

아래와 같은 부분을 피처로 해서 사용한다. 하지만 거의 사용하지 않는다. 뒤에 모델에 additional features 까지는 뭐 작동하긴 한다.(옛날 방법)



그 다음은 standard FFNN



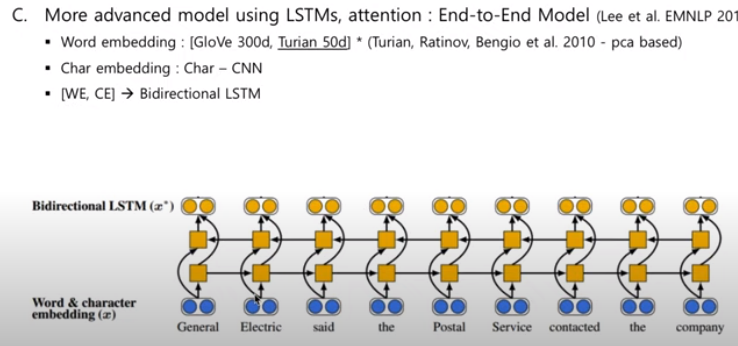
여기에다 추가적 피쳐(A에서 설명), 이전 두 단어(현재 mention 기준), 첫 번째 단어, 마지막 단어, 중심 단어 등등을 추가적으로 인풋에 넣는다.

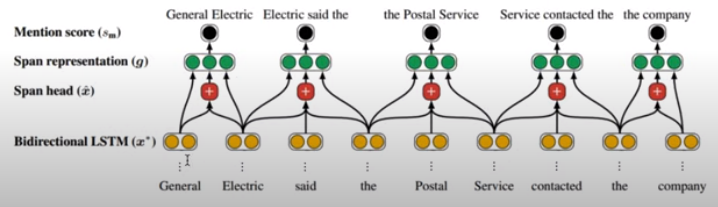
현재 SOTA는 BERT를 활용한 모델이며 그 직전에는 Kenton Lee(EMNLP 2017)을 소개했었다.

Mention ranking을 사용하고 LSTM & attention을 사용했다. Mention detection step이 따로 없는 end-to-end 모델이다.

특정 길이 이상의 text span은 모두 mention의 후보로 가정한다.(a, the는 독자적 span이 아닌 다른데 포함된다.) -> 뭔가 허점이 많을 것으로 예상된다.

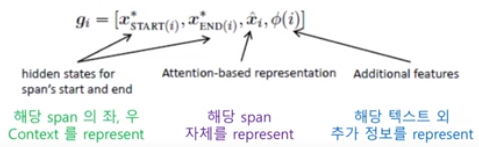
플로우는 다음과 같다.





이제 span representation을 구할 차례인데, 여기서 N(N+1)/2 개의 가능한 span이 존재한다.(N=sequence length)(처음, 마지막만 2 다리이고 나머지는 3다리 인듯?)

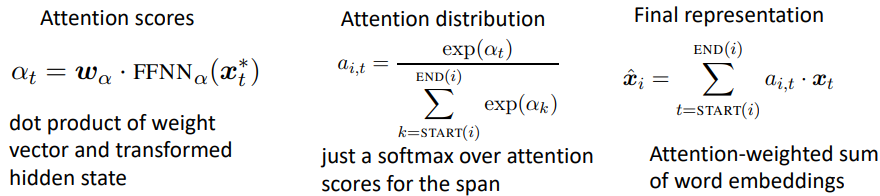
Span은 다음과 같다. [General], [General Electric], ..., [the], [the company], [company] 근데 어떤 기준으로 이렇게 나누는지 모르겠다..

뭐 여튼간  처럼 span representation이 가능하다.

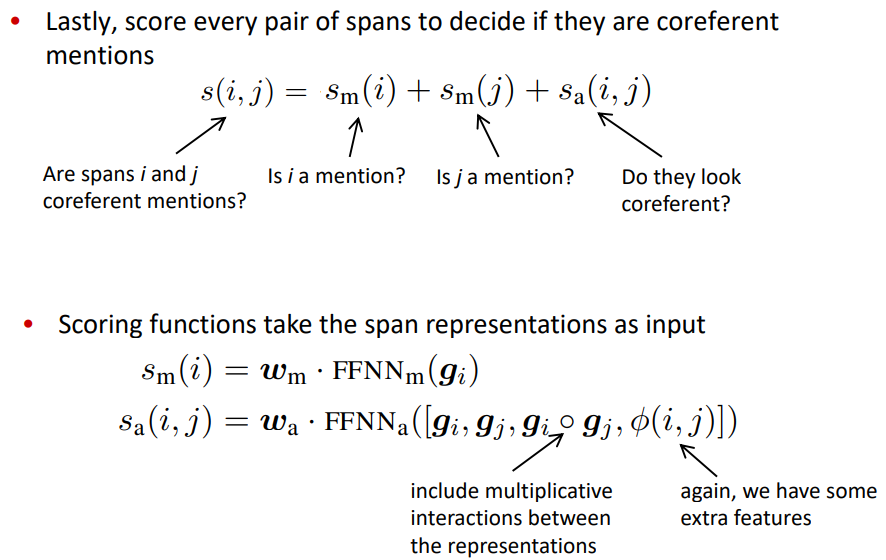
첫 단어와 마지막 단어, 그리고 해당 span 단어 자체를 represent 한것, 그리고 추가적인 피쳐(위에서 봤듯이)

이때 ^x-i에서 어텐션이 생기는데 아래처럼 위에서 언급한 것과 다른 그냥 어떤 FFNN을 거쳐(아마 span 내 단어들과 어텐션 하는듯?) 각 span의 어텐션 스코어를 획득하면 각 span 마다 어텐션 스코어가 생기고 이것을 softmax하면 attention distribution이 되며 final representation은 다음과 같다.

여기서 x-t는 span이다.



최종적으로 span i와 j가 coreferent mention인지 스코어를 매긴다.



가장 높은 span을 하나 선택해 coreference link에 추가

근데 이 방법이 좀 비효율적이다. T= number of words => T^2개의 span 존재 O(T^2) => 따라서 모든 span 끼리 또 비교 => O(T^4) => 모든 spans에 대해서 coreferent score 계산은 힘듬.

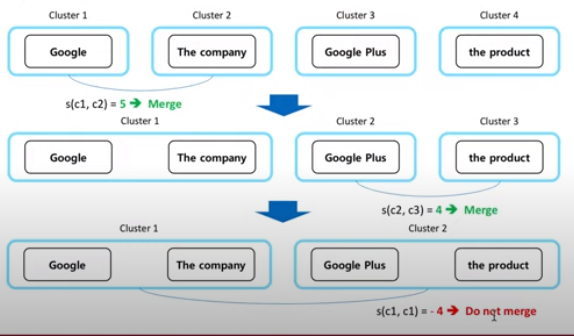
=> mention이 될 수 있는 span들만 고려하는 pruning 작업이 필요

그리고 의문인게 이렇게 대면 딱 coreference link 1개만 추가되는 것 아닌가..?

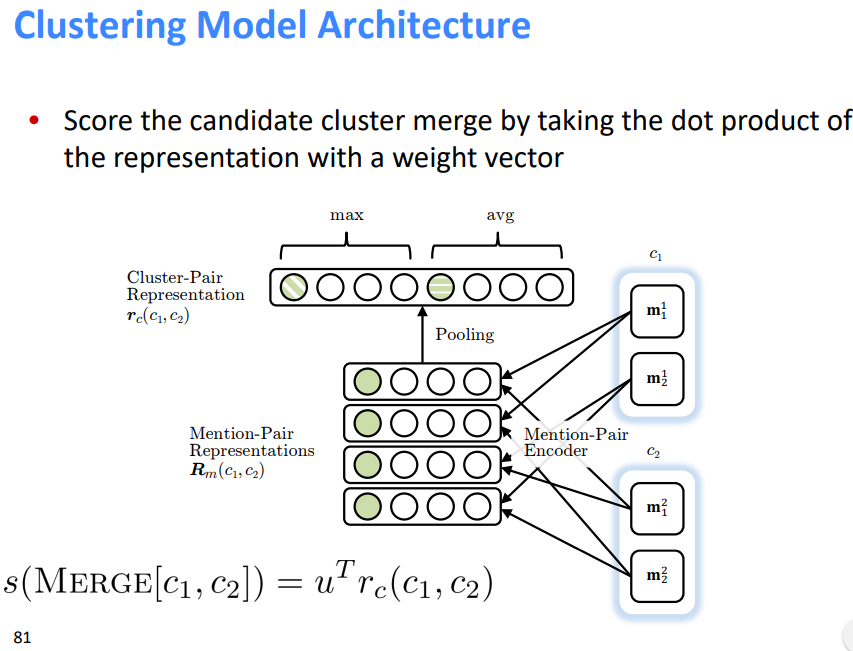
또한 어텐션은 해당 span(여기서는 이게 mention) 내에서 어떤 단어가 key word인지 알게 된다.

마지막은 clustering 모델!

각 mention이 독자적인 클러스터를 구성(근데 mention은 어떤 기준으로 나누는지 안알려줌)



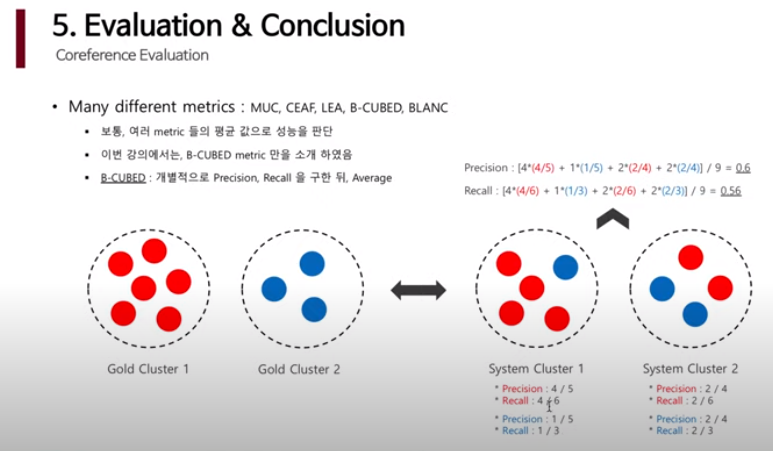
즉 mention들이 한 클러스터 내에 포함된다. 그래서 mention pair decision 단위가 아니라 cluster pair decision 단위로 생각한다. 그 로직은 다음과 같다.



현재 merge할 cluster pair는 이전 step에 만들어진 cluster들에 의존적이다. 일반적인 supervised learning은 사용할 수 없다. => 강화학습 => 매 step마다 바뀌는 머지 결과의 변화에 따라 coreference evaluation metric 변화가 reward로 사용됨.

Coreference metric evaluation: MUC, CEAF, LEA, B-CUBED, BLANC( 여러 metric들으 ㅣ평균 값으로 성능을 판단.)

여기서는 일단 B-CUBED만 설명



방식으로 precision & recall을 구할 수 있다.

또한 under cluster -> precision은 높고 recall이 줄어든다.(cluster가 여러 개)

Over cluster -> recall은 높아지지만 precision이 준다.(cluster가 1개)

* 균형 있는 지점을 찾아야 한다.

아직 이 Task 결과가 놀랍도록 성능이 좋진 않다. (어려운 문제)