Natural Language Processing with DeepLearning

week 14

The course

1. What is Coreference Resolution? (10 mins)
2. Applications of coreference resolution (5 mins)
3. Mention Detection (5 mins)
4. Some Linguistics: Types of Reference (5 mins) Four Kinds of Coreference Resolution Models
5. Rule-based (Hobbs Algorithm) (10 mins)
6. Mention-pair and mention-ranking models (15 mins)
7. Interlude: ConvNets for language (sequences) (10 mins)
8. Current state-of-the-art neural coreference systems (10 mins)
9. Evaluation and current results (10 mins)

< What is Coreference Resolution? >

그래서 coreference resolution에서 하고 싶은 것은 그럼 이 mentions of entities 에서 어떠한 mentions이 같은 entity를 나타내는가? 즉, 위의 예스에서 he 와 his는 무엇을 나타내는가? (여기서는 Barack Obama를 나타낸다)

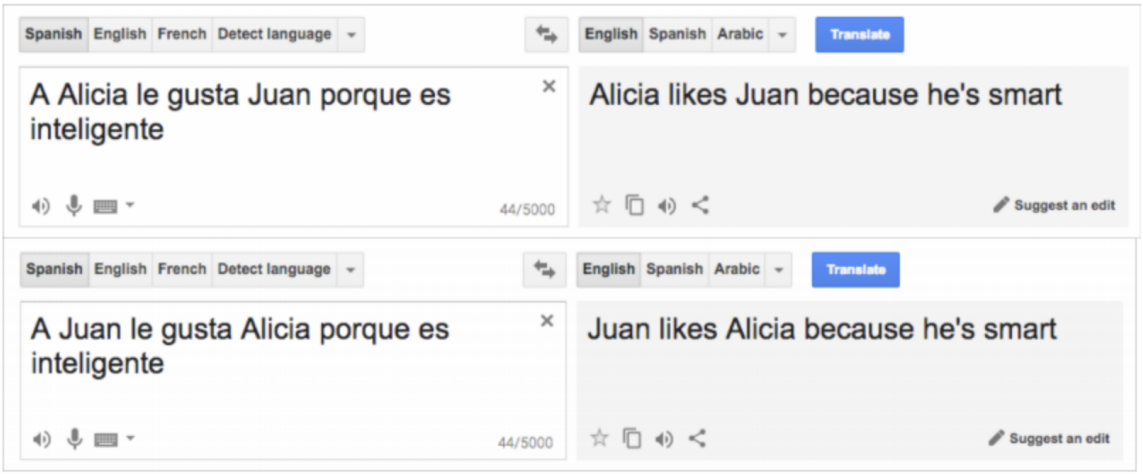
A couple of years later, Vanaja met Akhila at the local park. Akhila's son Prajwal was just two months younger than her son Akash, and they went to the same school. For the pre-school play, Prajwal was chosen for the lead role of the naughty child Lord Krishna. Akash was to be a tree. She resigned herself to make Akash the best tree that anybody had ever seen. She bought him a brown T-shirt and brown trousers to represent the tree trunk. Then she made a large cardboard cutout of a tree's foliage, with a circular opening in the middle for Akash's face. She attached red balls to it to represent fruits. It truly was the nicest tree. From The Star by Shruthi Rao, with some shortening.

* They와 같이, they 전에오는 2개이상의 entity를 포함하고 있는 경우, split antecedents라고 부른다.
* NLP에서 이러한 split antecedents를 처리할 수 있는 기술이 없다.
* 여기서 tree같은 경우는 Akash를 나타낸다고 할 수 있는데 그럼 tree와 Akash를 같은 entity로 분류 할 것인가? 어떤 시스템은 coreference가 있다고 판단하고, 다른 시스템은 없다고 판단할 것. 즉, 시스템 마다 다르다.
* 이곳에서 tree의 부분을 나타내는 entity로써 T-shirt, trousers, trunk 와 같은 단어들이 있는데 그럼 이 단어들도 tree와 같은 entity로 표시할 것인가 아니면 다른 entity로 표시할 것인가? (주로 다른 entity로 표시 되지만 그렇다고 이들이 아이에 관계가 없다고는 할 수 없다.)

2. Applications

* coreference resolution은 다양한 분야에 사용될 수 있다. (question answering, summarization, extracting facts from texts...)
* 예를 들어 'he was born in 1961' 이라는 문장에서 coreference resolution 기술을 사용해서 he가 누구를 의미하는지 알아 낼 수 있을 것이다.

- Machine translation
  - languages have different features for gender, number, dropped pronouns, etc.



* Spanish에서는 verb의 subject를 drop할 수 있는데 이를 영어로 번역할 때는 subject를 채워 넣을 필요가 있다.
* 여기서는 coreference decision을 통해서 smart가 Juan을 나타냄을 알아내서 he라는 주어를 넣어야겠다 라고 판단할 수 있지만 사실 Google translate은 coreference에 대해서 알지 못한다. (그냥 he가 default 값이기 때문에 he로 표시되는 것)
* Google translate에서는 학습된 문장을 통해서 he와 she를 결정하게 되는데, 그 결과 각 gender에 대한 stereotype을 반영하기도 한다.
* 이러한 문제를 계선하고 싶다면 coreference resolution을 사용할 필요가 있다.

- Dialogue Systems

  "Book tickets to see James Bond"

  "Spectre is playing near you at 2:00 and 3:00 today. How many tickets would you like?"

  "Two tickets for the showing at three"

* 여기서 각 색깔로 표시된 단어들은 서로 연관이 깊지만 엄밀히 따지면 같지 않다.

3. Coreference Resolution in Two Steps

1. Detect the mentions (easy)

   "[I] voted for [Nader] because [he] was most aligned with [[my] values]," [she] said

   - mentions can be nested!

2. Cluster the mentions (hard)

   "[I] voted for [Nader] because [he] was most aligned with [[my] values]," [she] said

4. Mention Detection

* 어떠한 entity를 갖는 모든 단어를 찾아내고 싶다.
* 우리는 mention을 3가지로 나눌 수 있다.

1. Pronouns

   - I, your, it, she, him, etc.

2. Named entities

   - People, places, etc.

3. Noun phrases

   - "a dog," "the big fluffy cat stuck in the tree"

* mention을 찾아내기 위해서 우리는 다른 NLP시스템을 사용한다.

- For detection: use other NLP systems

1. Pronouns
   - Use a part-of-speech tagger

2. Named entities
   - Use a NER system (like hw3)

3. Noun phrases
   - Use a parser (especially a constituency parser – next week!)

5. Mention Detection: Not so Simple

- Are these mentions?
  - It is sunny
  - Every student
  - No student
  - The best donut in the world
  - 100 miles

* 위에 말한 3가지 조건을 만족하지만 mention이라고 말하기 어려운 경우도 있다.

6. How to deal with these bad mentions?
* 안좋은 mention을 거를 수 있도록 classifer를 학습 시칸다.
* 하지만 우리는 classifier를 학습시키는 단계를 건너 뛸 때가 많은데 그 이유는 그냥 진행시켜도 특정한 entity를 나타내지 않는 mention들은 혼자서 분류되기 때문에 시스템에 나쁜 영향을 끼치지 않는다.

7. Can we avoid a pipelined system?
* POS tagger, NER, paser를 하나하나 따로 사용하는 것이 아니라, 이러한 일을 한번에 할 수 있는, mention을 발견해내는 classifier를 학습시킬 수 있을 것인가?
= 한번에 할 수 있는 방법이 있다. mention detection과 coreference resolution을 2 단계로 나누지 않고 end-to-end로 한번에 진행한다.

8. On to Coreference! First, some linguistics
* 비슷해보여서 서로 햇갈리지만 사실은 다른 두 개념을 소개한다.

- **Coreference** is when two mentions refer to the same entity in the world
  - *Barack Obama traveled to … Obama*

- A related linguistic concept is **anaphora:** when a term (anaphor) refers to another term (antecedent)
  - the interpretation of the anaphor is in some way determined by the interpretation of the antecedent
  - *Barack Obama said he would sign the bill.*
    antecedent anaphor

* mention이 같은 entity를 나타낸다면 coreference라고 한다.
* 문장 속에서 앞에 나온 단어를 가리키는 것을 anaphora라고 한다. (뒤에 나오는 단어가 앞에 나오는 단어에 의해서 해석 될 때)
* 그럼 굳이 왜 coreference 와 anaphora를 나누는 걸까? 둘의 차이점은?

9. Not all anaphoric relations are coreferential

- Not all noun phrases have reference

- *Every dancer* twisted *her knee*.
- *No dancer* twisted *her knee.*

- There are three NPs in each of these sentences; because the first one is non-referential, the other two aren't either.

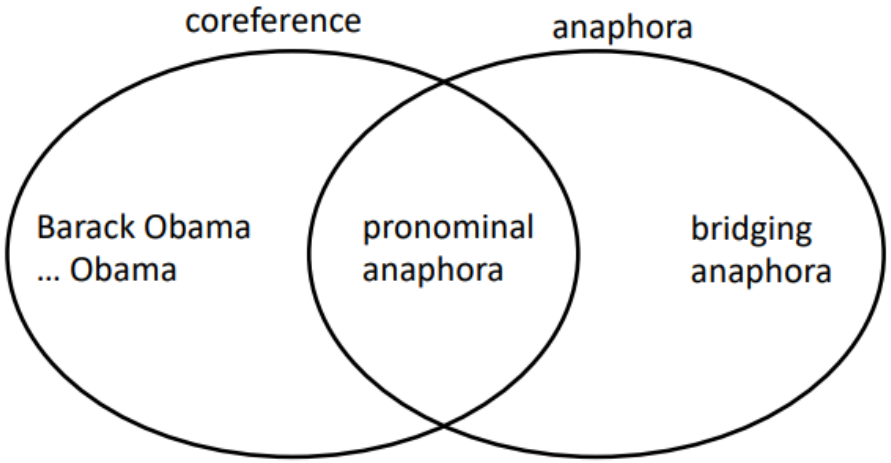
* 'No dancer twisted her knee' 라는 문장에서 her는 no dancer에 대한 anaphora라고 할 수 있는데 no dancer는 어떠한 reference도 갖지 않는다. 즉 no dancer는 특정한 entity를 가지고 있지 않는다. 이 문장에서는 anaphoric한 관계를 발견해낼 수 있지만 그렇다고해서 coreference를 발견할 수는 없다.

10. Anaphora vs Coreference

- Not all anaphoric relations are coreferential

We went to see a concert last night. The tickets were really expensive.

- This is referred to as bridging anaphora.



23

* 위 사진의 예시 문장에서 concert와 tickets은 서로 다른 entity를 이야기 하고 있기에 coreference하다고 이야기할 수 없지만, ticket은 concert의 ticket을 뜻하기 때문에(ticket의 뜻이 앞에 나오는 concert라는 단어에 의해서 해석되기 때문에), concert와 ticket은 anaphoric한 관계에 있다고할 수 있다.
* 이러한 경우를 bridging anaphora라고 한다.

11. Anaphora vs Cataphora
* Cataphora는 Anaphora의 반대 말이다. Anaphora는 뒤에 나오는 단어의 의미를 앞에 나오는 단어에서 찾는다면, cataphora는 앞에나오는 단어의 의미를 뒤에서 찾는다.

12. Cataphora

"From the corner of the divan of Persian saddle-bags on which he was lying, smoking, as was his custom, innumerable cigarettes, Lord Henry Wotton could just catch the gleam of the honey-sweet and honey-coloured blossoms of a laburnum..."

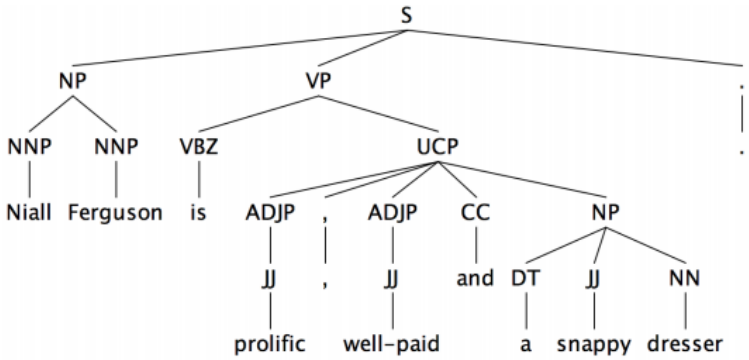
* He와 his가 무엇을 의미하는지 he 와 his 다음에 나오는 것을 볼 수 있다.

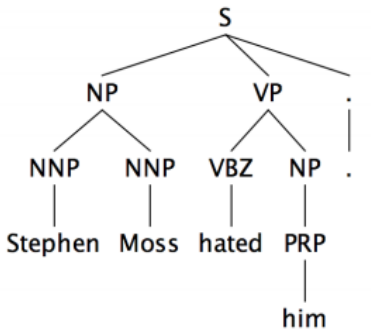
* 최근에는 cataphra라는 개념은 잘 사용되지 않는다. 실제로 coreference 시스템에서 mention을 발견하면 reference를 발견하기위해 mention 앞에 나오는 문장들을 살펴보지 그 뒤의 문장들을 살펴보지 않는다.

13. Four Kinds of Coreference Models
* Rule-based: 가장 기본
* Mention Pair
* Mention Ranking: easiest simple systems
* Clustering: 이론상 가장 좋아보이지만 실제로는 가장 좋은 성능을 발휘 하기 힘들다.

14. Traditional pronominal anaphora resolution: Hobb's naive algorithm
* 이 algorithm은 pronouns에 대한 reference를 찾기 위한 것이다. (대명사가 무엇을 가리키는지 찾는 행위)

15. Hobbs Algorithm Example





* 여기서 우리는 him에 대한 reference를 찾고 싶다.
* 이 것이 coreference resolution을 위한 완벽한 algorithm이라고는 할 수 없지만 이에대한 baseline을 구축했다고 ㅏㄹ 수 있다.
1. him에 해당하는 NP에서 시작한다.
2. tree를 거슬러 올라가서 NP나 S를 찾는다. (이 예시에서는 S를 찾을 수 있다)
3. 그 후 S를 기준으로 왼쪽에서 오른쪽으로 BFS식으로 tree를 훑어 내려가게 되는데, 이 때 NP를 발견하면 이 NP는 S와 NP 사이에 또 다른 NP나 S가 존재할 경우 him에 대한 reference 후보가 될 수 있다. (여기서는 NP와 S 사이에서 또 다른 NP나 S를 발견할 수 없기 때문에 him에대한 reference가 될 수 없다.)
4. 한 문장을 다 살폈다면, 그 전의 문장을 tree를 BFS로 훑는다. 이때 발견된 NP는 him에 대한 reference 후보가 된다.

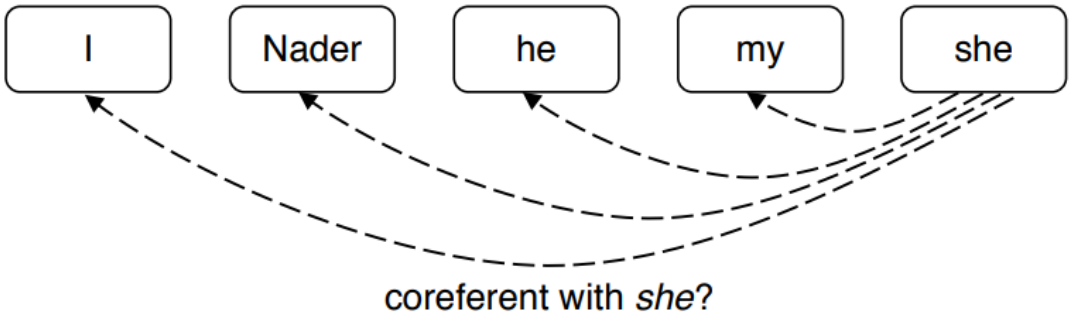
16. Knowledge-based Pronominal Coreference
- She poured water from the pitcher into the cup until it was full
- She poured water from the pitcher into the cup until it was empty”

* 이 두 문장은 같은 구조를 가지고 있지만 it이 가리키고 있는것은 다르다. 이러한 경우에는 위에 설명한 Hobb's algorithm을 사용할 수 없다.

17. Coreference Models: Mention Pair

- Train a binary classifier that assigns every pair of mentions a probability of being coreferent: $p(m_i, m_j)$
  - e.g., for "she" look at all **candidate antecedents** (previously occurring mentions) and decide which are coreferent with it

*"**I** voted for **Nader** because **he** was most aligned with **my** values," **she** said.*



* We're going to take paris of mentions, and we're going to train a binary classifier that says 'is coreferent of isn't coreferent'.
* 문장을 왼쪽에서 오른쪽으로 훑을 것인데, 이 때 새로운 mention을 발견할 때 마다 we're going to evaluate our classifier with respect to every proceding mention, are they coreferent and it's gonna say yes or no.

18. Mention Pair Training

- N mentions in a document
- y_{ij} = 1 if mentions m_i and m_j are coreferent, -1 if otherwise
- Just train with regular cross-entropy loss (looks a bit different because it is binary classification)

$$J = -\sum_{i=2}^{N}\sum_{j=1}^{i} y_{ij} \log p(m_j, m_i)$$

Iterate through mentions

Iterate through candidate antecedents (previously occurring mentions)

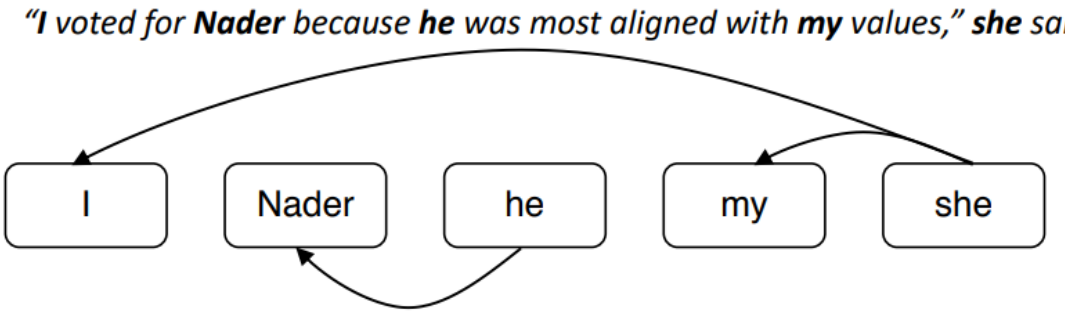
Coreferent mentions pairs should get high probability, others should get low probability

* Classifier의 결과값은, 두 mention이 coreferent 하다면 1을, coreferent 하지 않다면 0의 값을 내보낸다.

* P(mj, mi) = Coreference model that predicts the probability of them being coreferent.
* 그리고 이 model을, 다른 곳에서도 많이 사용한, cross entropy loss를 사용해서 학습시킨다.

19. Mention Pair Test Time

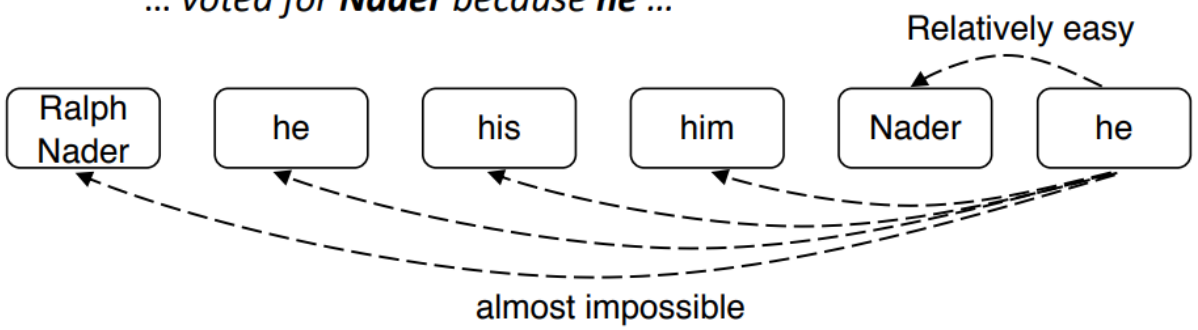
- Coreference resolution is a clustering task, but we are only scoring pairs of mentions… what to do?
- Pick some threshold (e.g., 0.5) and add **coreference links** between mention pairs where $p(m_i, m_j)$ is above the threshold



* mention들을 pair로 묶어서 classifer에 집어 넣는다.
* Classifier의 결과가 어떠한 확률을 기준으로 yes 나 no 가 나올 것이다. 예를들어 0.5가 기준값이라면 0.5를 기준으로 coreference link를 결성할 것이다.
* 이 mention들을 clustering 하기 위해서 transitive closure 방법을 사용할 것이다. Trasitive closure란 A가 B와 coreferent 하고 B가 C랑 coreferent하다면 A와 C는 coreferent하다 라는 것을 의미한다.
* transitive closure를 사용하기 때문에 혹시나 실수가 일어나면 모든 mention이 하나로 묶일 수 있는 (over cluster) 위험성도 존재한다.
* coreferent를 이루지 않는 mention도 존재한다. 그렇다면 classifier에서는 모든 다른 mention에 대해서 no 값을 배출할 것이고 이 mention은 singleton mention이 될 것이다.
* Mention Pair는 coreference resolution을 위한 최선의 방법이 아닐 때가 많다. 그 이유는 we have this phenomenon of anaphora where we have textural dependence. We're not really want to make this all coreference decisions. We'd like to make the anaphora decisions of textural dependence. We'd like to choose one example of what is this anaphora relationship.

20. Mention Pair Models: Disadvantage

- Suppose we have a long document with the following mentions
  - **Ralph Nader** … **he** … **his** … **him** … <several paragraphs>
    … *voted for* ***Nader*** *because* ***he*** …



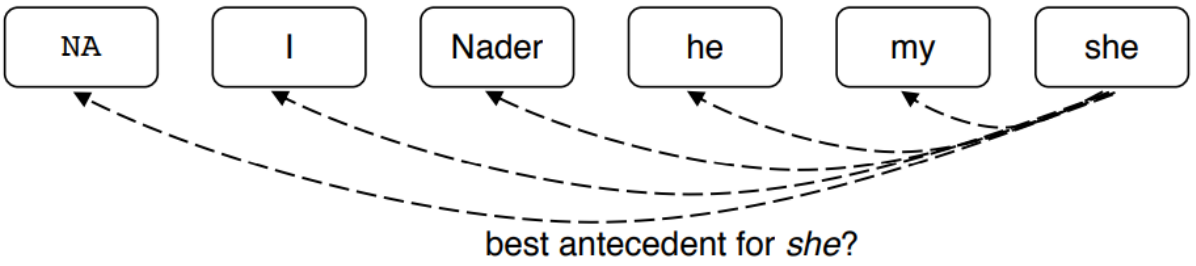
- Many mentions only have one clear antecedent
  - But we are asking the model to predict all of them
- Solution: instead train the model to predict only one antecedent for each mention
  - More linguistically plausible

42

* 만약 mention을 많이 포함하는 긴 문서가 있으면, 해당하는 mention을 모두 찾아내서 yes라는 값을 배출하는 것이 아니라 한 mention을 잘 표현하는 특정한 mention 하나를 찾아내고 싶다.

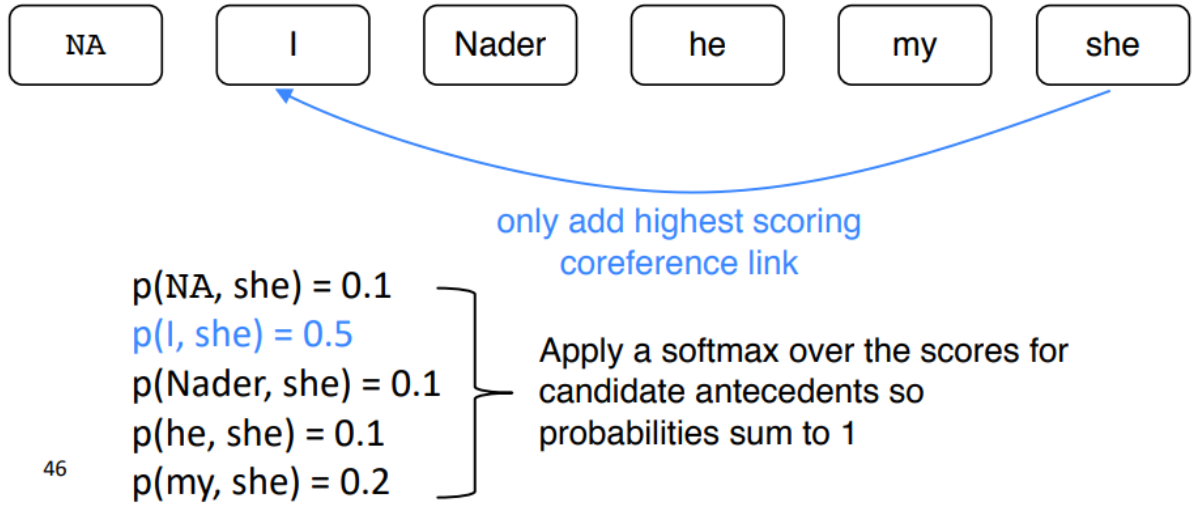
21. Coreference Models: Mention Ranking

- Assign each mention its highest scoring candidate antecedent according to the model
- Dummy NA mention allows model to decline linking the current mention to anything ("singleton" or "first" mention)



* For mention ranking, for each mention, we're going to find an antecedent that comes before it in the text, that it is coreferent with, and we're going to make a one of N decision.
* 예를들어 she가 어떤 mention과 coreferent 한지가 궁금하다고 하면 우리는 이 she 라는 단어가 다른 많은 mention과 coreferent 하더라도 하나의 mention만 고르게 될 것이다.
* 전에 coreferent한 mention이 없을 경우가 있을 텐데, 이 때를 위해서 NA값을 포함시킨다.

- Assign each mention its highest scoring candidate antecedent according to the model
- Dummy NA mention allows model to decline linking the current mention to anything (“singleton” or “first” mention)



46

* Mention Pair model 같은 경우는, model을 학습시킬 때 she와 my, she와 I 모두 높은 점수를 얻었어야 하지만 이제는 하나만 높은 점수를 얻으면 된다.
* She와 다른 mention에 대한 pair에 softmax를 적용시킨다. 이 때 softmax를 적용시켜서 나온 값들의 합은 1이 될 것이다. 이때 우리가 원하는 것은 어떠한 antecedents에 대해서 높은 확률값을 가지게 되는 것이다. (prior reference 가 있다면 antecedent에 대해서 높은 확률값을 얻을 것이고, 없다면 NA와 높은 확률값을 얻을 것이다.)
* 그리고 가장 높은 값을 가진 것에 한하여 coreference link를 만든다.

22. Coreference Models: Training

- We want the current mention m_j to be linked to *any one* of the candidate antecedents it's coreferent with.
- Mathematically, we want to maximize this probability:

$$\sum_{j=1}^{i-1} \mathbb{1}(y_{ij} = 1) p(m_j, m_i)$$

Iterate through candidate antecedents (previously occurring mentions)

For ones that are coreferent to m_j...

...we want the model to assign a high probability

- The model could produce 0.9 probability for one of the correct antecedents and low probability for everything else, and the sum will still be large

* 우리는 전에 나오는 다양한 antecednets 중에 적어도 한개의 높은 conference score를 얻고 싶다.

- Turning this into a loss function:

$$J = \sum_{i=2}^{N} -\log \left(\sum_{j=1}^{i-1} \mathbb{1}(y_{ij} = 1) p(m_j, m_i) \right)$$

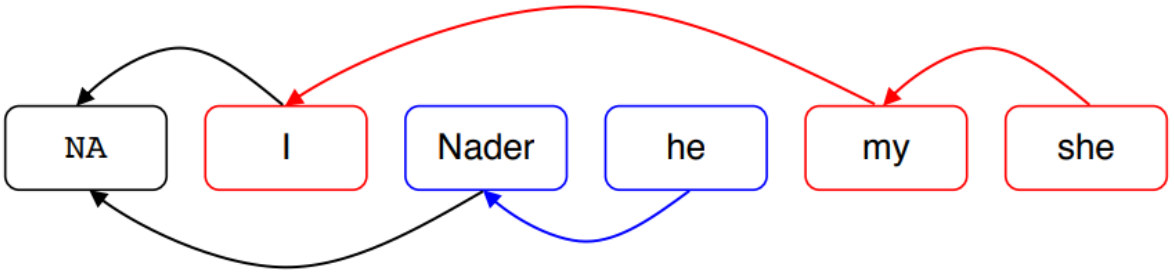
Iterate over all the mentions in the document

Usual trick of taking negative log to go from likelihood to loss

* We want to minimize the loss.

23. Mention Ranking Models: Test Time

- Pretty much the same as mention-pair model except each mention is assigned only one antecedent



* Softmax classifier is going to assign one antecedent for each mention. 각 mention 당 하나의 mention만을 제공한다.

24. How do we compute the probabilites?
- Non-neural statistical classifier, simple neural network, more advanced model using LSTMs + Attention...

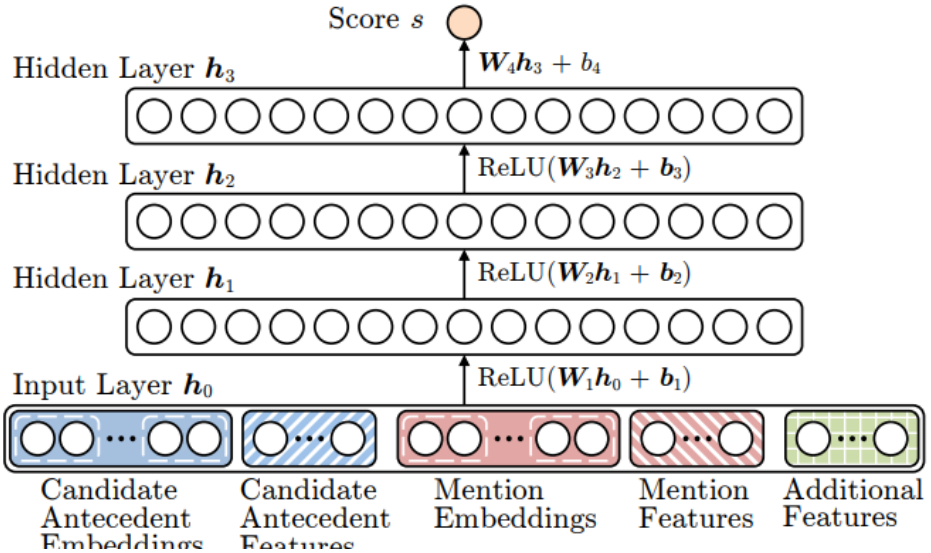
25. Non-Neural Coref Model: Features

- Person/Number/Gender agreement
  - Jack gave Mary a gift. She was excited.
- Semantic compatibility
  - ... the mining conglomerate ... the company ...
- Certain syntactic constraints
  - John bought him a new car. [him can not be John]
- More recently mentioned entities preferred for referenced
  - John went to a movie. Jack went as well. He was not busy.
- Grammatical Role: Prefer entities in the subject position
  - John went to a movie with Jack. He was not busy.
- Parallelism:
  - John went with Jack to a movie. Joe went with him to a bar.
- ...

* 가장 classic한 방법
* You have whole bunch of features and you had a feature based statistical classifier which gave a score. And the above are the features that we could use.
* Throw these all features into a statistical classifier and that's sort of 2000s decade coref systems as to how they're built.

26. Neural Coref Model

- Standard feed-forward neural network
  - Input layer: word embeddings and a few categorical features



* Word embeddings와 categorical features를 input값으로 하는 neural network를 구성해서 score값을 구한다.
* 여기서 나오는 feature들은 feature-based statistical classifier에서 나오는 것들을 몇몇 포함한다. 예를들어서 mention의 문법적 관계 (이게 주어인지 목적어인지)
* Closer things are more likely to be coreferent. So you might have additional features which record how far apart dimensions are.