Natural Language Processing with Deep Learning

Coreference Resolution
What is Coreference Resolution?

- Identify all mentions that refer to the same real world entity

Barack Obama nominated Hillary Rodham Clinton as his secretary of state on Monday. He chose her because she had foreign affairs experience as a former First Lady.
Barack Obama nominated **Hillary Rodham Clinton** as his **secretary of state** on Monday. He chose **her** because **she** had foreign affairs experience as a former **First Lady**.
Applications

• Full text understanding
  • information extraction, question answering, summarization, ...
  • “He was born in 1961” (Who?)
Applications

• Full text understanding
• Machine translation
  • languages have different features for gender, number, dropped pronouns, etc.
Applications

- Full text understanding
- Machine translation
- 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”
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
Mention Detection

- Mention: span of text referring to some entity
- Three kinds of mentions:

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”
Mention Detection

• Span of text referring to some entity
• For detection: use other NLP systems

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

2. Named entities
   • Use a NER system

3. Noun phrases
   • Use a parser (especially a constituency parser or NP chunker)
Mention Detection: Not so Simple

- Marking all pronouns, named entities, and NPs as mentions over-generates mentions
- Are these mentions?
  - *It is sunny*
  - *Every student*
  - *No student*
  - *The best donut in the world*
  - *100 miles*
How to deal with these bad mentions?

• Could train a classifier to filter out spurious mentions

• Much more common: keep all mentions as “candidate mentions”
  • After your coreference system is done running discard all singleton mentions (i.e., ones that have not been marked as coreference with anything else)
Can we avoid a pipelined system?

• We could instead train a classifier specifically for mention detection instead of using a POS tagger, NER system, and parser.

• Or even jointly do mention-detection and coreference resolution end-to-end instead of in two steps
  • Will cover later in this lecture!
On to Coreference! First, some linguistics

- **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.*

<table>
<thead>
<tr>
<th>antecedent</th>
<th>anaphor</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Barack Obama</em></td>
<td><em>he</em></td>
</tr>
<tr>
<td><em>would sign the bill.</em></td>
<td></td>
</tr>
</tbody>
</table>
Anaphora vs Coreference

• Coreference with named entities
  
  text

  Barack Obama

  Obama

  world

• Anaphora

  text

  Barack Obama

  he

  world
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.
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.
Anaphora vs. Cataphora

• Usually the antecedent comes before the anaphor (e.g., a pronoun), but not always
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...”

(Oscar Wilde – The Picture of Dorian Gray)
Four Kinds of Coreference Models

- Rule-based (pronominal anaphora resolution)
- Mention Pair
- Mention Ranking
- Clustering
Traditional pronominal anaphora resolution: Hobbs’ naive algorithm

1. Begin at the NP immediately dominating the pronoun
2. Go up tree to first NP or S. Call this X, and the path p.
3. Traverse all branches below X to the left of p, left-to-right, breadth-first. Propose as antecedent any NP that has a NP or S between it and X
4. If X is the highest S in the sentence, traverse the parse trees of the previous sentences in the order of recency. Traverse each tree left-to-right, breadth first. When an NP is encountered, propose as antecedent. If X not the highest node, go to step 5.
Hobbs’ naive algorithm (1976)

5. From node X, go up the tree to the first NP or S. Call it X, and the path p.

6. If X is an NP and the path p to X came from a non-head phrase of X (a specifier or adjunct, such as a possessive, PP, apposition, or relative clause), propose X as antecedent

   (The original said “did not pass through the N’ that X immediately dominates”, but the Penn Treebank grammar lacks N’ nodes....)

7. Traverse all branches below X to the left of the path, in a left-to-right, breadth first manner. Propose any NP encountered as the antecedent

8. If X is an S node, traverse all branches of X to the right of the path but do not go below any NP or S encountered. Propose any NP as the antecedent.

9. Go to step 4

   Until deep learning still often used as a feature in ML systems!
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”

- The city council refused the women a permit because they feared violence.
- The city council refused the women a permit because they advocated violence.
  - Winograd (1972)
- These are called Winograd Schema
  - Recently proposed as an alternative to the Turing test
    - See: Hector J. Levesque “On our best behaviour” IJCAI 2013
    - http://commonsensereasoning.org/winograd.html
- If you’ve fully solved coreference, arguably you’ve solved AI
“... the naïve approach is quite good. Computationally speaking, it will be a long time before a semantically based algorithm is sophisticated enough to perform as well, and these results set a very high standard for any other approach to aim for.

“Yet there is every reason to pursue a semantically based approach. The naïve algorithm does not work. Any one can think of examples where it fails. In these cases it not only fails; it gives no indication that it has failed and offers no help in finding the real antecedent.”

— Hobbs (1978), Lingua, p. 345
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.

Positive examples: want $p(m_i, m_j)$ to be near 1
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.

Negative examples: want $p(m_i, m_j)$ to be near 0
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
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.

Take the transitive closure to get the clustering.

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

Even though the model did not predict this coreference link, I and my are coreferent due to transitivity.
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* ...

![Diagram of mention pair models](image)

- 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
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)

<table>
<thead>
<tr>
<th>NA</th>
<th>I</th>
<th>Nader</th>
<th>he</th>
<th>my</th>
<th>she</th>
</tr>
</thead>
</table>

- \( p(NA, \text{she}) = 0.1 \)
- \( p(I, \text{she}) = 0.5 \)
- \( p(\text{Nader}, \text{she}) = 0.1 \)
- \( p(\text{he}, \text{she}) = 0.1 \)
- \( p(\text{my}, \text{she}) = 0.2 \)

Apply a softmax over the scores for candidate antecedents so probabilities sum to 1

only add highest scoring coreference link
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:
  \[ \sum_{j=1}^{i-1} 1(y_{ij} = 1)p(m_j, m_i) \]

  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
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:

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

- 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
Mention Ranking Models: Test Time

- Pretty much the same as mention-pair model except each mention is assigned only one antecedent.
How do we compute the probabilities?

A. Non-neural statistical classifier

B. Simple neural network

C. More advanced model using LSTMs, attention
A. 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.
B. Neural Coref Model

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

• Embeddings
  • Previous two words, first word, last word, head word, ... of each mention
    • The **head** word is the “most important” word in the mention – you can find it using a parser. e.g., *The fluffy cat stuck in the tree*

• Still need some other features:
  • Distance
  • Document genre
  • Speaker information
C. End-to-end Model

- Current state-of-the-art model for coreference resolution (Kenton Lee et al. from UW, EMNLP 2017)
- Mention ranking model
- Improvements over simple feed-forward NN
  - Use an LSTM
  - Use attention
  - Do mention detection and coreference end-to-end
    - No mention detection step!
    - Instead consider every span of text (up to a certain length) as a candidate mention
      - a span is just a contiguous sequence of words
End-to-end Model

- Next, represent each span of text \(i\) going from \(\text{START}(i)\) to \(\text{END}(i)\) as a vector. For example, for “the postal service”

Span representation \((g)\)

Span head \((\hat{x})\)

Bidirectional LSTM \((x^*)\)

Word & character embedding \((x)\)

General Electric said the Postal Service contacted the company

Span representation: \(g_i = [x^*_{\text{START}(i)}, x^*_{\text{END}(i)}, \hat{x}_i, \phi(i)]\)

BILSTM hidden states for span’s start and end

Attention-based representation (details next slide) of the words in the span

Additional features
End-to-end Model

- \( \hat{x}_i \) is an attention-weighted average of the word embeddings in the span

\[
\alpha_t = \mathbf{w}_\alpha \cdot \text{FFNN}_\alpha(x^*_t)
\]

dot product of weight vector and transformed hidden state

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

just a softmax over attention scores for the span

\[
\hat{x}_i = \sum_{t=\text{START}(i)}^{\text{END}(i)} a_{i,t} \cdot x_t
\]

Attention-weighted sum of word embeddings
End-to-end Model

- Why include all these different terms in the span?

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

- Represents the context to the left and right of the span
- Represents the span itself
- Represents other information not in the text
End-to-end Model

- Lastly, score every pair of spans to decide if they are coreferent mentions

\[ s(i, j) = s_m(i) + s_m(j) + s_a(i, j) \]

- Scoring functions take the span representations as input

\[ s_m(i) = w_m \cdot \text{FFNN}_m(g_i) \]
\[ s_a(i, j) = w_a \cdot \text{FFNN}_a([g_i, g_j, g_i \odot g_j, \phi(i, j)]) \]
End-to-end Model

- Intractable to score every pair of spans
  - \(O(T^2)\) spans of text in a document (\(T\) is the number of words)
  - \(O(T^4)\) runtime!
  - So have to do lots of pruning to make work (only consider a few of the spans that are likely to be mentions)

- Attention learns which words are important in a mention (a bit like head words)
  
  (A fire in a Bangladeshi garment factory) has left at least 37 people dead and 100 hospitalized. Most of the deceased were killed in the crush as workers tried to flee (the blaze) in the four-story building.
Last Coreference Approach: Clustering-Based

• Coreference is a clustering task, let’s use a clustering algorithm!
  • In particular we will use agglomerative clustering

• Start with each mention in it’s own singleton cluster

• Merge a pair of clusters at each step
  • Use a model to score which cluster merges are good
Coreference Models: Clustering-Based

Google recently ... the company announced Google Plus ... the product features ...
Coreference Models: Clustering-Based

Mention-pair decision is difficult

Google

Google Plus

? coreferent

Cluster-pair decision is easier

Cluster 1

Google

the company

Cluster 2

Google Plus

the product

? coreferent
Clustering Model Architecture

From Clark & Manning, 2016

Merge clusters \( c_1 = \{ \text{Google, the company} \} \) and \( c_2 = \{ \text{Google Plus, the product} \} \)?

<table>
<thead>
<tr>
<th>Mention Pairs</th>
<th>Mention-Pair Representations</th>
<th>Cluster-Pair Representation</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Google, Google Plus)</td>
<td><img src="image" alt="Mention-Pair Representation" /></td>
<td><img src="image" alt="Cluster-Pair Representation" /></td>
<td>( s(\text{MERGE}[c_1, c_2]) )</td>
</tr>
<tr>
<td>(Google, the product)</td>
<td><img src="image" alt="Mention-Pair Representation" /></td>
<td><img src="image" alt="Cluster-Pair Representation" /></td>
<td></td>
</tr>
<tr>
<td>(the company, Google Plus)</td>
<td><img src="image" alt="Mention-Pair Representation" /></td>
<td><img src="image" alt="Cluster-Pair Representation" /></td>
<td></td>
</tr>
<tr>
<td>(the company, the product)</td>
<td><img src="image" alt="Mention-Pair Representation" /></td>
<td><img src="image" alt="Cluster-Pair Representation" /></td>
<td></td>
</tr>
</tbody>
</table>
Clustering Model Architecture

- First produce a vector for each pair of mentions
  - e.g., the output of the hidden layer in the feedforward neural network model
Clustering Model Architecture

• Score the candidate cluster merge by taking the dot product of the representation with a weight vector

\[ s(\text{MERGE}[c_1, c_2]) = u^T r_c(c_1, c_2) \]
Clustering Model: Training

• Current candidate cluster merges depend on previous ones it already made
  • So can’t use regular supervised learning
  • Instead use something like Reinforcement Learning to train the model
    • Reward for each merge: the change in a coreference evaluation metric
9. Coreference Evaluation

• Many different metrics: MUC, CEAF, LEA, B-CUBED, BLANC
  • Often report the average over a few different metrics
Coreference Evaluation

- An example: B-cubed
  - For each mention, compute a precision and a recall
  - Then average the individual Ps and Rs

\[
P = \frac{4(4/5) + 1(1/5) + 2(2/4) + 2(2/4)}{9} = 0.6
\]
Coreference Evaluation

100% Precision, 33% Recall

50% Precision, 100% Recall,
System Performance

- OntoNotes dataset: ~3000 documents labeled by humans
  - English and Chinese data

- Report an F1 score averaged over 3 coreference metrics
## System Performance

<table>
<thead>
<tr>
<th>Model</th>
<th>English</th>
<th>Chinese</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lee et al. (2010)</td>
<td>~55</td>
<td>~50</td>
<td>Rule-based system, used to be state-of-the-art!</td>
</tr>
<tr>
<td>Chen &amp; Ng (2012)</td>
<td>54.5</td>
<td>57.6</td>
<td>Non-neural machine learning models</td>
</tr>
<tr>
<td>[CoNLL 2012 Chinese winner]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fernandes (2012)</td>
<td>60.7</td>
<td>51.6</td>
<td>Neural mention ranker</td>
</tr>
<tr>
<td>[CoNLL 2012 English winner]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wiseman et al. (2015)</td>
<td>63.3</td>
<td>—</td>
<td>Neural clustering model</td>
</tr>
<tr>
<td>Clark &amp; Manning (2016)</td>
<td>65.4</td>
<td>63.7</td>
<td>End-to-end neural mention ranker</td>
</tr>
<tr>
<td>Lee et al. (2017)</td>
<td>67.2</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>
Where do neural scoring models help?

- Especially with NPs and named entities with no string matching. Neural vs non-neural scores:
  
  $18.9 \ F_1$ vs $10.7 \ F_1$ on this type compared to $68.7 \ F_1$ vs $66.1 \ F_1$

  These kinds of coreference are hard and the scores are still low!

**Example Wins**

<table>
<thead>
<tr>
<th>Anaphor</th>
<th>Antecedent</th>
</tr>
</thead>
<tbody>
<tr>
<td>the country’s leftist rebels</td>
<td>the guerillas</td>
</tr>
<tr>
<td>the company</td>
<td>the New York firm</td>
</tr>
<tr>
<td>216 sailors from the “USS cole”</td>
<td>the crew</td>
</tr>
<tr>
<td>the gun</td>
<td>the rifle</td>
</tr>
</tbody>
</table>
Conclusion

• Coreference is a useful, challenging, and linguistically interesting task
  • Many different kinds of coreference resolution systems
• Systems are getting better rapidly, largely due to better neural models
  • But overall, results are still not amazing
• Try out a coreference system yourself!
  • [http://corenlp.run/](http://corenlp.run/) (ask for coref in Annotations)
  • [https://huggingface.co/coref/](https://huggingface.co/coref/)