Announcements

• Congratulations on surviving Ass5!
  • I hope it was a rewarding state-of-the-art learning experience
  • It was a new assignment: We look forward to feedback on it

• Final project proposals: Thank you! Lots of interesting stuff!
  • We’ll get them back to you tomorrow!
  • Do get started working on your projects now – talk to your mentor!

• We plan to get Ass4 back this week...

• Final project milestone is out, due Tuesday March 2 (next Tuesday)

• Second invited speaker – Colin Raffel, super exciting! – is this Thursday
  • Again, you need to write a reaction paragraph for it
Lecture Plan:

Lecture 16: Coreference Resolution

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)
1. What is Coreference Resolution?

- Identify all mentions that refer to the same entity in the word
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.
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
3. Mention Detection

- Mention: A 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.: Paris, Joe Biden, Nike

3. Noun phrases
   - “a dog,” “the big fluffy cat stuck in the tree”
Mention Detection

- Mention: A span of text referring to some entity

- For detection: traditionally, use a pipeline of other NLP systems

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

2. Named entities
   - Use a Named Entity Recognition system

3. Noun phrases
   - Use a parser (especially a constituency parser – next week!)
Mention Detection: Not Quite 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)
Avoiding a traditional pipeline system

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

• Or we can not even try to do mention detection explicitly:
  • We can build a model that begins with all spans and jointly does mention-detection and coreference resolution end-to-end in one model
    • Will cover later in this lecture!
4. 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 different-but-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
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)
Taking stock ...

- It’s often said that language is interpreted “in context”
- We’ve seen some examples, like word-sense disambiguation:
  - I took money out of the bank vs. The boat disembarked from the bank
- Coreference is another key example of this:
  - Obama was the president of the U.S. from 2008 to 2016. He was born in Hawaii.
- As we progress through an article, or dialogue, or webpage, we build up a (potentially very complex) **discourse model**, and we interpret new sentences/utterances with respect to our model of what’s come before.
- Coreference and anaphora are all we see in this class of whole-discourse meaning
Four Kinds of Coreference Models

- Rule-based (pronominal anaphora resolution)
- Mention Pair
- Mention Ranking
- Clustering [skipping this year; see Clark and Manning (2016)]
5. 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!
Hobbs Algorithm Example

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.
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.
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.
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](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
"I voted for Nader because he was most aligned with my values," she said.
• 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.
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
Mention Pair Test Time

• Coreference resolution is a clustering task, but we are only scoring pairs of mentions... what to do?
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

“I voted for Nader because he was most aligned with my values,” she said.
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
- 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 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
- Take the transitive closure to get the clustering

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

Relatively easy

almost impossible
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
7. 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)

![Diagram showing coreference ranks for mentions: NA, I, Nader, he, my, she. The question is: "best antecedent for she?"
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).

Positive examples: model has to assign a high probability to either one (but not necessarily both).
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).

- 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).

\[
\begin{align*}
p(\text{NA, she}) &= 0.1 \\
p(l, \text{she}) &= 0.5 \\
p(\text{Nader, she}) &= 0.1 \\
p(\text{he, she}) &= 0.1 \\
p(\text{my, she}) &= 0.2
\end{align*}
\]

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

![Diagram of coreference resolution with probabilities]

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

Only add highest scoring coreference link.

Apply a softmax over the scores for candidate antecedents so probabilities sum to 1.
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
How do we compute the probabilities?

A. Non-neural statistical classifier

B. Simple neural network

C. More advanced model using LSTMs, attention, transformers
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 to get a strongly performing model:
  • Distance
  • Document genre
  • Speaker information
7. Convolutional Neural Nets (very briefly)

- Main CNN/ConvNet idea:
- What if we compute vectors for every possible word subsequence of a certain length?
- Example: “tentative deal reached to keep government open” computes vectors for:
  - tentative deal reached, deal reached to, reached to keep, to keep government, keep government open
- Regardless of whether phrase is grammatical
- Not very linguistically or cognitively plausible???
- Then group them afterwards (more soon)
What is a convolution anyway?

- 1d discrete convolution definition:
  \[(f * g)[n] = \sum_{m=-M}^{M} f[n - m]g[m].\]

- Convolution is classically used to extract features from images
  - Models position-invariant identification
  - Go to cs231n!

- 2d example
  - Yellow color and red numbers show filter (=kernel) weights
  - Green shows input
  - Pink shows output

From Stanford UFLDL wiki
A 1D convolution for text

Apply a **filter** (or **kernel**) of size 3

<table>
<thead>
<tr>
<th>tentative</th>
<th>0.2</th>
<th>0.1</th>
<th>-0.3</th>
<th>0.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>deal</td>
<td>0.5</td>
<td>0.2</td>
<td>-0.3</td>
<td>-0.1</td>
</tr>
<tr>
<td>reached</td>
<td>-0.1</td>
<td>-0.3</td>
<td>-0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>to</td>
<td>0.3</td>
<td>-0.3</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>keep</td>
<td>0.2</td>
<td>-0.3</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>government</td>
<td>0.1</td>
<td>0.2</td>
<td>-0.1</td>
<td>-0.1</td>
</tr>
<tr>
<td>open</td>
<td>-0.4</td>
<td>-0.4</td>
<td>0.2</td>
<td>0.3</td>
</tr>
</tbody>
</table>

| t,d,r     | -1.0| 0.0 | 0.50 |
| d,r,t     | -0.5| 0.5 | 0.38 |
| r,t,k     | -3.6| -2.6| 0.93 |
| t,k,g     | -0.2| 0.8 | 0.31 |
| k,g,o     | 0.3 | 1.3 | 0.21 |

+ **bias**

⇒ **non-linearity**
1D convolution for text with padding

<table>
<thead>
<tr>
<th></th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>tentative</td>
<td>0.2</td>
<td>0.1</td>
<td>-0.3</td>
<td>0.4</td>
</tr>
<tr>
<td>deal</td>
<td>0.5</td>
<td>0.2</td>
<td>-0.3</td>
<td>-0.1</td>
</tr>
<tr>
<td>reached</td>
<td>-0.1</td>
<td>-0.3</td>
<td>-0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>to</td>
<td>0.3</td>
<td>-0.3</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>keep</td>
<td>0.2</td>
<td>-0.3</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>government</td>
<td>0.1</td>
<td>0.2</td>
<td>-0.1</td>
<td>-0.1</td>
</tr>
<tr>
<td>open</td>
<td>-0.4</td>
<td>-0.4</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>Ø</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Apply a **filter** (or **kernel**) of size 3

<table>
<thead>
<tr>
<th>3</th>
<th>1</th>
<th>2</th>
<th>-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>2</td>
<td>1</td>
<td>-3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>
### 3 channel 1D convolution with padding = 1

<table>
<thead>
<tr>
<th>Word</th>
<th>Convolution Coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>0.0 0.0 0.0 0.0</td>
</tr>
<tr>
<td>tentative</td>
<td>0.2 0.1 −0.3 0.4</td>
</tr>
<tr>
<td>deal</td>
<td>0.5 0.2 −0.3 −0.1</td>
</tr>
<tr>
<td>reached</td>
<td>−0.1 −0.3 −0.2 0.4</td>
</tr>
<tr>
<td>to</td>
<td>0.3 −0.3 0.1 0.1</td>
</tr>
<tr>
<td>keep</td>
<td>0.2 −0.3 0.4 0.2</td>
</tr>
<tr>
<td>government</td>
<td>0.1 0.2 −0.1 −0.1</td>
</tr>
<tr>
<td>open</td>
<td>−0.4 −0.4 0.2 0.3</td>
</tr>
<tr>
<td>Ø</td>
<td>0.0 0.0 0.0 0.0</td>
</tr>
</tbody>
</table>

#### Apply 3 filters of size 3

<table>
<thead>
<tr>
<th>Filter</th>
<th>Coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø,t,d</td>
<td>−0.6 0.2 1.4</td>
</tr>
<tr>
<td>t,d,r</td>
<td>−1.0 1.6 −1.0</td>
</tr>
<tr>
<td>d,r,t</td>
<td>−0.5 −0.1 0.8</td>
</tr>
<tr>
<td>r,t,k</td>
<td>−3.6 0.3 0.3</td>
</tr>
<tr>
<td>t,k,g</td>
<td>−0.2 0.1 1.2</td>
</tr>
<tr>
<td>k,g,o</td>
<td>0.3 0.6 0.9</td>
</tr>
<tr>
<td>g,o,Ø</td>
<td>−0.5 −0.9 0.1</td>
</tr>
</tbody>
</table>

Could also use (zero) padding = 2

Also called “wide convolution”
**conv1d, padded with max pooling over time**

<table>
<thead>
<tr>
<th></th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>φ</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>tentative</td>
<td>0.2</td>
<td>0.1</td>
<td>−0.3</td>
<td>0.4</td>
</tr>
<tr>
<td>deal</td>
<td>0.5</td>
<td>0.2</td>
<td>−0.3</td>
<td>−0.1</td>
</tr>
<tr>
<td>reached</td>
<td>−0.1</td>
<td>−0.3</td>
<td>−0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>to</td>
<td>0.3</td>
<td>−0.3</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>keep</td>
<td>0.2</td>
<td>−0.3</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>government</td>
<td>0.1</td>
<td>0.2</td>
<td>−0.1</td>
<td>−0.1</td>
</tr>
<tr>
<td>open</td>
<td>−0.4</td>
<td>−0.4</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>φ</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>−0.6</th>
<th>0.2</th>
<th>1.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>φ,t,d</td>
<td>−0.6</td>
<td>0.2</td>
<td>1.4</td>
</tr>
<tr>
<td>t,d,r</td>
<td>−1.0</td>
<td>1.6</td>
<td>−1.0</td>
</tr>
<tr>
<td>d,r,t</td>
<td>−0.5</td>
<td>−0.1</td>
<td>0.8</td>
</tr>
<tr>
<td>r,t,k</td>
<td>−3.6</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>t,k,g</td>
<td>−0.2</td>
<td>0.1</td>
<td>1.2</td>
</tr>
<tr>
<td>k,g,o</td>
<td>0.3</td>
<td>0.6</td>
<td>0.9</td>
</tr>
<tr>
<td>g,o,φ</td>
<td>−0.5</td>
<td>−0.9</td>
<td>0.1</td>
</tr>
</tbody>
</table>

**max p**

<table>
<thead>
<tr>
<th></th>
<th>0.3</th>
<th>1.6</th>
<th>1.4</th>
</tr>
</thead>
</table>

Apply 3 **filters** of size 3

<table>
<thead>
<tr>
<th>3</th>
<th>1</th>
<th>2</th>
<th>−3</th>
</tr>
</thead>
<tbody>
<tr>
<td>−1</td>
<td>2</td>
<td>1</td>
<td>−3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>−1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>−1</td>
<td>−1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>−1</th>
<th>2</th>
<th>−1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>−1</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
conv1d, padded with ave pooling over time

<table>
<thead>
<tr>
<th></th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
<th>0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ø</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>tentative</td>
<td>0.2</td>
<td>0.1</td>
<td>−0.3</td>
<td>0.4</td>
</tr>
<tr>
<td>deal</td>
<td>0.5</td>
<td>0.2</td>
<td>−0.3</td>
<td>−0.1</td>
</tr>
<tr>
<td>reached</td>
<td>−0.1</td>
<td>−0.3</td>
<td>−0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>to</td>
<td>0.3</td>
<td>−0.3</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>keep</td>
<td>0.2</td>
<td>−0.3</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>government</td>
<td>0.1</td>
<td>0.2</td>
<td>−0.1</td>
<td>−0.1</td>
</tr>
<tr>
<td>open</td>
<td>−0.4</td>
<td>−0.4</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>ø</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>ø,t,d</td>
<td>−0.6</td>
<td>0.2</td>
<td>1.4</td>
<td></td>
</tr>
<tr>
<td>t,d,r</td>
<td>−1.0</td>
<td>1.6</td>
<td>−1.0</td>
<td></td>
</tr>
<tr>
<td>d,r,t</td>
<td>−0.5</td>
<td>−0.1</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>r,t,k</td>
<td>−3.6</td>
<td>0.3</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>t,k,g</td>
<td>−0.2</td>
<td>0.1</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>k,g,o</td>
<td>0.3</td>
<td>0.6</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>g,o,ø</td>
<td>−0.5</td>
<td>−0.9</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>ave p</td>
<td>−0.87</td>
<td>0.26</td>
<td>0.53</td>
<td></td>
</tr>
</tbody>
</table>

Apply 3 filters of size 3

<table>
<thead>
<tr>
<th>3</th>
<th>1</th>
<th>2</th>
<th>−3</th>
</tr>
</thead>
<tbody>
<tr>
<td>−1</td>
<td>2</td>
<td>1</td>
<td>−3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>−1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>−1</td>
<td>−1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>−1</th>
<th>2</th>
<th>−1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>−1</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
In PyTorch

batch_size = 16
word_embed_size = 4
seq_len = 7
input = torch.randn(batch_size, word_embed_size, seq_len)
conv1 = Conv1d(in_channels=word_embed_size, out_channels=3,
               kernel_size=3)  # can add: padding=1
hidden1 = conv1(input)
hidden2 = torch.max(hidden1, dim=2)  # max pool
Learning Character-level Representations for Part-of-Speech Tagging  Dos Santos and Zadrozny (2014)

- Convolution over characters to generate word embeddings
- Fixed window of word embeddings used for PoS tagging
8. End-to-end Neural Coref Model

- Current state-of-the-art models for coreference resolution
  - Kenton Lee et al. from UW (EMNLP 2017) et seq.
- 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

- First embed the words in the document using a word embedding matrix and a character-level CNN.
End-to-end Model

- Then run a bidirectional LSTM over the document
End-to-end Model

- Next, represent each span of text $i$ going from $\text{START}(i)$ to $\text{END}(i)$ as a vector.
End-to-end Model

- Next, represent each span of text $i$ going from START($i$) to END($i$) as a vector

- *General, General Electric, General Electric said, ..., Electric, Electric said*, ... will all get its own vector representation
End-to-end Model

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

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

Additional features:
- BILSTM hidden states for span’s start and end
- Attention-based representation (details next slide) of the words in the span
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

hidden states for span’s start and end

Attention-based representation

Additional features
End-to-end Model

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

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

Are spans \( i \) and \( j \) coreferent mentions?

- Is \( i \) a mention?  Is \( j \) a mention?  Do they look coreferent?

- Scoring functions take the span representations as input

\[ s_m(i) = w_m \cdot \text{FFNN}_m(g_i) \]
\[ s_a(i, j) = w_a \cdot \text{FFNN}_a([g_i, g_j, g_i \circ g_j, \phi(i, j)]) \]

include multiplicative interactions between the representations again, we have some extra features
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.
BERT-based coref: Now has the best results!

- Pretrained transformers can learn long-distance semantic dependencies in text.
- Idea 1, SpanBERT: Pretrains BERT models to be better at span-based prediction tasks like coref and QA
- Idea 2, BERT-QA for coref: Treat Coreference like a deep QA task
  - “Point to” a mention, and ask “what is its antecedent”
  - Answer span is a coreference link
9. Coreference Evaluation

- Many different metrics: MUC, CEAF, LEA, B-CUBED, BLANC
  - People often report the average over a few different metrics
  - Essentially the metrics think of coreference as a clustering task and evaluate the quality of the clustering

Diagram:
- System Cluster 1
  - Gold Cluster 1
  - Gold Cluster 2
- System Cluster 2
Coreference Evaluation

- An example: B-cubed
  - For each mention, compute a precision and a recall

\[ P = \frac{4}{5}, \quad R = \frac{4}{6} \]
Coreference Evaluation

- An example: B-cubed
- For each mention, compute a precision and a recall

<table>
<thead>
<tr>
<th>System Cluster 1</th>
<th>System Cluster 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gold Cluster 1</td>
<td>Gold Cluster 2</td>
</tr>
</tbody>
</table>

P = 4/5  
R = 4/6

P = 1/5  
R = 1/3
Coreference Evaluation

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

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

System Cluster 1
- P = 4/5
- R = 4/6

System Cluster 2
- P = 1/5
- R = 1/3

Gold Cluster 1
- P = 2/4
- R = 2/6

Gold Cluster 2
- P = 2/4
- R = 2/3
Coreference Evaluation

100% Precision, 33% Recall

50% Precision, 100% Recall,
System Performance

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

- Standard evaluation: an F1 score averaged over 3 coreference metrics
## System Performance

<table>
<thead>
<tr>
<th>Model</th>
<th>English</th>
<th>Chinese</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lee et al. (2010)</td>
<td>~55</td>
<td>~50</td>
</tr>
<tr>
<td>Chen &amp; Ng (2012) [CoNLL 2012 Chinese winner]</td>
<td>54.5</td>
<td>57.6</td>
</tr>
<tr>
<td>Fernandes (2012) [CoNLL 2012 English winner]</td>
<td>60.7</td>
<td>51.6</td>
</tr>
<tr>
<td>Wiseman et al. (2015)</td>
<td>63.3</td>
<td>—</td>
</tr>
<tr>
<td>Clark &amp; Manning (2016)</td>
<td>65.4</td>
<td>63.7</td>
</tr>
<tr>
<td>Lee et al. (2017)</td>
<td>67.2</td>
<td>—</td>
</tr>
<tr>
<td>Joshi et al. (2019)</td>
<td>79.6</td>
<td>—</td>
</tr>
<tr>
<td>Wu et al. (2019) [CorefQA]</td>
<td>79.9</td>
<td>—</td>
</tr>
<tr>
<td>Xu et al. (2020)</td>
<td>80.2</td>
<td></td>
</tr>
<tr>
<td>Wu et al. (2020) [CorefQA + SpanBERT-large]</td>
<td>83.1</td>
<td></td>
</tr>
</tbody>
</table>

- **Rule-based system**, used to be state-of-the-art!
- Non-neural machine learning models
- **Neural mention ranker**
- **Neural clustering model**
- End-to-end neural ranker
- End-to-end neural mention ranker + **SpanBERT**
- CorefQA
- CorefQA + SpanBERT rulez
Where do neural scoring models help?

- Especially with NPs and named entities with no string matching. Neural vs non-neural scores:

  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 &quot;USS cole&quot;</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 most models still make many mistakes – OntoNotes coref is easy newswire case

- 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/)