ALGORITHMS FOR INFORMATION RETRIEVAL AND INTELLIGENT WEB

Mention Detection

Mention Detection

- The first stage of coreference is mention detection: finding the spans of text that constitute each mention.
- Mention detection algorithms typically take a liberal approach by proposing a wide range of candidate mentions, prioritizing high recall. The emphasis at this stage is on inclusivity, capturing a broad set of potential mentions.
- Subsequent stages in the coreference resolution process involve more refined filtering to enhance precision.

Mention Detection

- Using the previous sample sentence:
  Victoria Chen, CFO of Megabucks Banking, saw her pay jump to $2.3 million, as the 38-year-old also became the company's president. It is widely known that she came to Megabucks from rival Lotsabucks.
  It might result in the following list of 13 potential mentions:

Victoria Chen	$2.3 million	she
CFO of Megabucks Banking	the 38-year-old	Megabucks
Megabucks Banking	the company	Lotsabucks
her	the company's president	
her pay	It	

Mention Detection

- The span-based algorithm described first extracts all n-gram spans of words up to N=10. However, it's essential to note that many noun phrases (NPs) and the majority of random n-gram spans are not referring expressions. Consequently, all such mention detection systems need to eventually filter out pleonastic/expletive pronouns like "It," appositives such as "CFO of Megabucks Banking Inc," or predicate nominals like "the company's president" or "$2.3 million."

Mention Detection

- Mention-detection rules are sometimes designed specifically for particular evaluation campaigns. For OntoNotes, for example, mentions are not embedded within larger mentions, and while numeric quantities are annotated, they are rarely coreferential. Thus for OntoNotes tasks like CoNLL 2012 , a common first pass rule-based mention detection algorithm is:

1. Take all NPs, possessive pronouns, and named entities.
2. Remove numeric quantities (100 dollars, 8%), mentions embedded in larger mentions, adjectival forms of nations, and stop words (like *there*).
3. Remove pleonastic *it* based on regular expression patterns.

Coreference Resolution

- An essential aspect of language processing involves determining the entity being discussed in a text. Take, for example, the following passage:

  "Victoria Chen, CFO of Megabucks Banking, saw her pay jump to $2.3 million, as the 38-year-old became the company's president. It is widely known that she came to Megabucks from rival Lotsabucks."

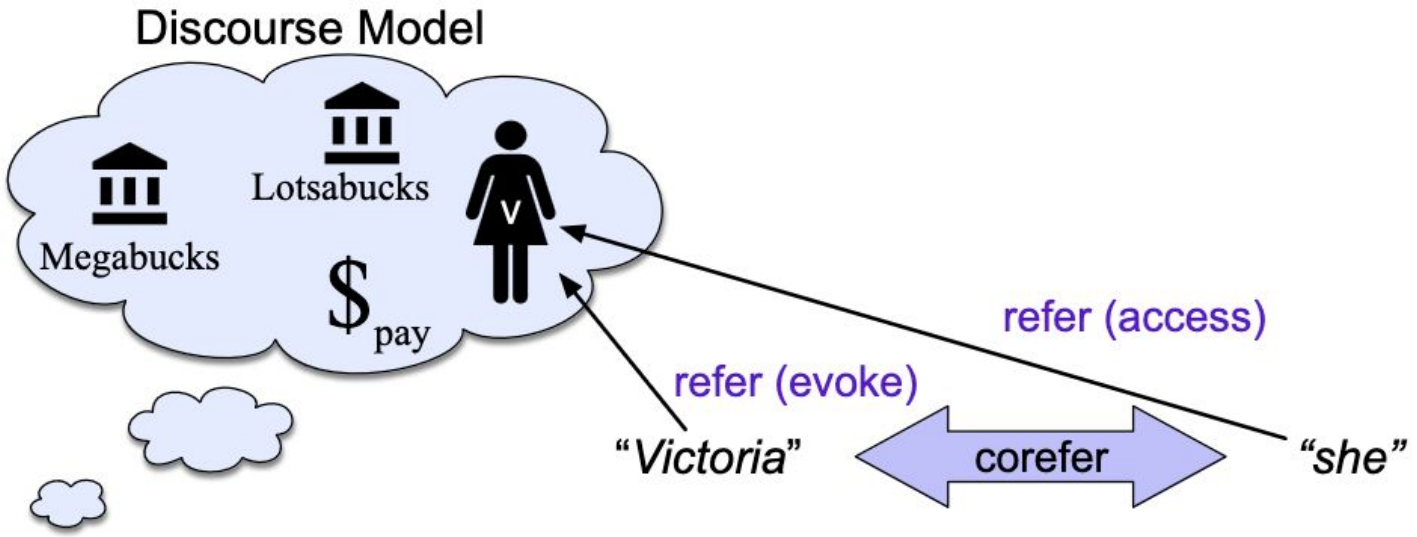
- In this passage, the underlined phrases serve as referring expressions used by the writer to identify the person named Victoria Chen. These linguistic expressions, such as "her" or "Victoria Chen," are referred to as mentions or referring expressions. The actual person being referred to (Victoria Chen) is called the referent, and to differentiate between referring expressions and their referents, the former is typically italicized.

Coreference Resolution

- When two or more referring expressions are employed to denote the same entity in a discourse, it is termed coreference. In this passage, "Victoria Chen" and "she" are examples of referring expressions that corefer to the same individual. This understanding of coreference is crucial in comprehending and analyzing the relationships between different elements within a given text.
- Natural language processing systems (and humans) interpret linguistic expressions with respect to a discourse model. A discourse model is a mental model that the understander builds incrementally when interpreting a text, containing representations of the entities referred to in the text, as well as properties of the entities and relations among them.

Coreference Resolution

- When a referent is first mentioned in a discourse, we say that a representation for it is evoked into the evoked model. Upon subsequent mention, this representation is accessed from the model.



Coreference Resolution

- Reference in a text to an entity that has been previously introduced into the discourse is called **anaphora**, and the referring expression used is said to be **anaphor, or anaphoric.**
- The anaphor corefers with a prior mention (in this case Victoria Chen) that is called the **antecedent**. Not every referring expression is an antecedent. An entity that has only a single mention in a text is called a **singleton**.
- Coreference resolution is the task of determining whether two mentions corefer, meaning they refer to the same entity in the discourse model, also known as the same discourse entity. The collection of coreferring expressions is commonly referred to as a coreference chain or a cluster.

Coreference Resolution

- In the context of the provided example in processing , a coreference resolution algorithm would be tasked with identifying at least four coreference chains, corresponding to the four entities in the discourse model:

1. {Victoria Chen, her, the 38-year-old, She}
2. {Megabucks Banking, the company, Megabucks}
3. {her pay}
4. {Lotsabucks}

Note: her is syntactically part of another mention, her pay, referring to a completely different discourse entity.

Coreference Resolution

- Coreference resolution thus comprises two tasks (although they are often performed jointly):
  (1) identifying the mentions
  (2) clustering them into coreference chains/discourse entities.
  Beyond coreference resolution, there is often a need to delve deeper into determining which real-world entity is associated with a given discourse entity. For instance, the mention of "Washington" could refer to the U.S. state, the capital city, or the historical figure George Washington. The interpretation of the sentence varies significantly depending on the specific real-world entity being referenced.

Coreference Resolution

- The formulation of the coreference resolution task is as follows: When provided with a text T, the objective is to identify all entities within the text and determine the coreference links between these entities. The evaluation of the system's performance is conducted by comparing the generated coreference links produced by the system with those present in manually created gold coreference annotations for the same text T. Eg:

$[\text{Victoria Chen}]_a^1$, CFO of $[\text{Megabucks Banking}]_a^2$, saw $[[\text{her}]_b^1 \text{ pay}]_a^3$ jump to \$2.3 million, as $[\text{the 38-year-old}]_c^1$ also became $[[\text{the company}]_b^2$'s president. It is widely known that $[\text{she}]_d^1$ came to $[\text{Megabucks}]_c^2$ from rival $[\text{Lotsabucks}]_a^4$.

Coreference Resolution

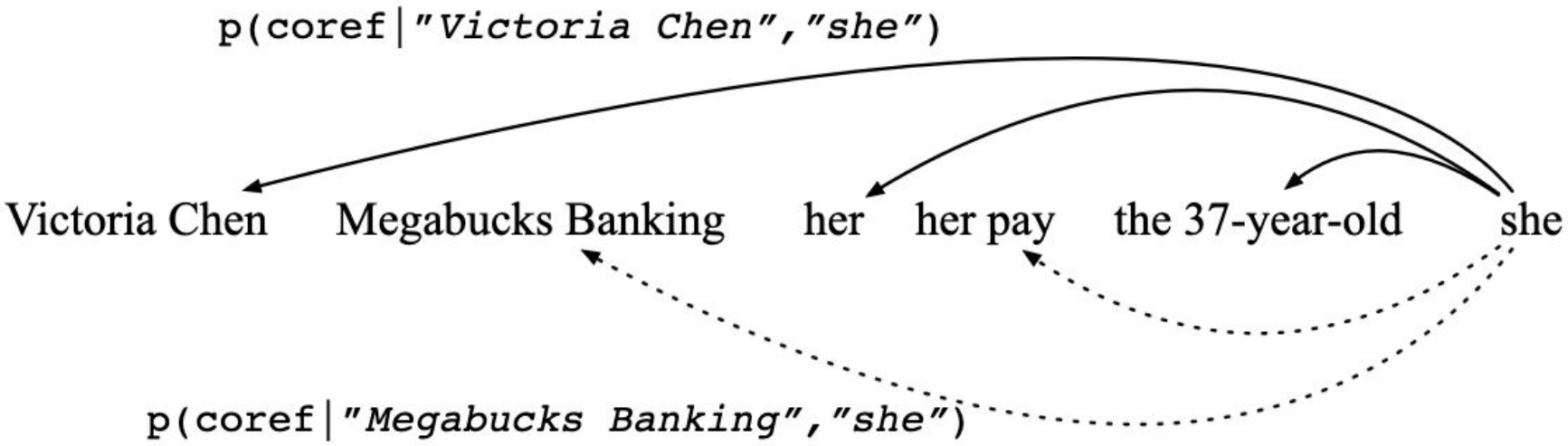
- Solving this task involves addressing challenges such as resolving pronominal anaphora (identifying that "her" refers to Victoria Chen), filtering out non-referential pronouns like the pleonastic "It" in phrases like "It has been ten years," handling definite noun phrases to establish coreference relationships (e.g., recognizing that "the 38-year-old" is coreferent with Victoria Chen), and deciphering references within names (e.g., realizing that "Megabucks" is the same entity as "Megabucks Banking"). This multifaceted process is essential for accurately identifying and linking mentions into coherent coreference clusters.

Coreference Resolution

- Exactly what counts as a mention and what links are annotated differs from task to task and dataset to dataset. For example some coreference datasets do not label singletons, making the task much simpler. Resolvers can achieve much higher scores on corpora without singletons, since singletons constitute the majority of mentions in running text, and they are often hard to distinguish from non-referential NPs.
- A few popular coreference datasets include OntoNotes, ISNotes, LitBank coreference corpus, ARRAU corpus, etc.

Mention-Pair Algorithm

- The mention-pair architecture is based around a classifier that— as its name suggests—is given a pair of mentions, a candidate anaphor and a candidate antecedent, and makes a binary classification decision: coreferring or not.
- Lets refer to the previous example for the pronoun ‘she’:



Mention-Pair Algorithm

- For each prior mention (Victoria Chen, Megabucks Banking, her, etc.), the binary classifier computes a probability: whether or not the mention is the antecedent of she. We want this probability to be high for actual antecedents (Victoria Chen, her, the 38-year-old) and low for non-antecedents (Megabucks Banking, her pay).
- For training, we need a heuristic for selecting training samples; since most pairs of mentions in a document are not coreferent, selecting every pair would lead to a massive overabundance of negative samples. The most common heuristic, from is to choose the closest antecedent as a positive example, and all pairs in between as the negative examples.

Mention-Pair Algorithm

- More formally, for each anaphor mention m_i we create:
  - one positive instance (m_i, m_j) where m_j is the closest antecedent to m_i, and a negative instance (m_i, m_k) for each m_k between m_j and m_i
- Thus for the anaphor she, we would choose (she, her) as the positive example and no negative examples. Similarly, for the anaphor the company we would choose (the company, Megabucks) as the positive example and (the company, she) (the company, the 38-year-old) (the company, her pay) and (the company, her) as negative examples.

Mention-Pair Algorithm

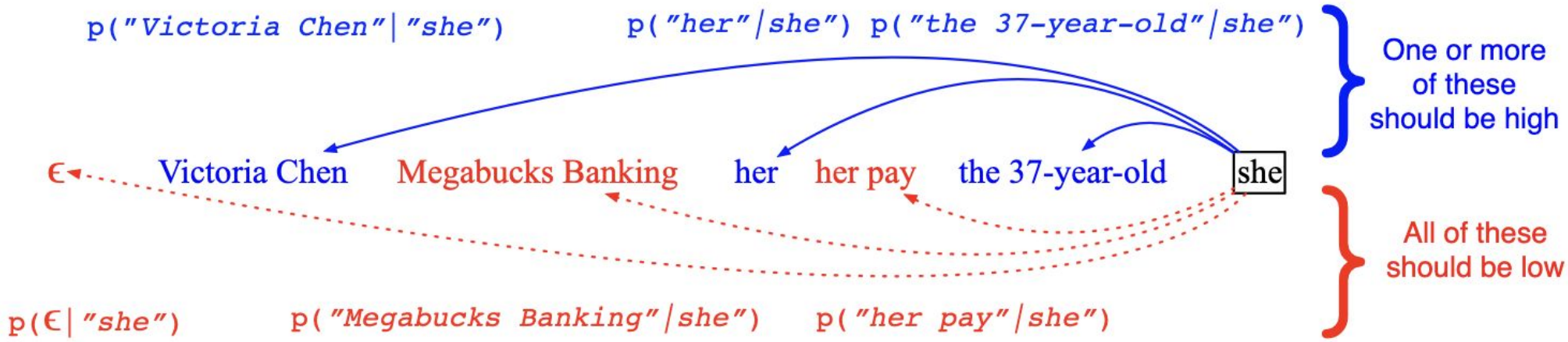
- While the mention-pair model has the advantage of simplicity, it has two main problems. First, the classifier doesn't directly compare candidate antecedents to each other, so it's not trained to decide, between two likely antecedents, which one is in fact better.
- Second, it ignores the discourse model, looking only at mentions, not entities. Each classifier decision is made completely locally to the pair, without being able to take into account other mentions of the same entity.

Mention Rank Algorithm

- The mention ranking model directly compares candidate antecedents to each other, choosing the highest-scoring antecedent for each anaphor.
- In early formulations, for a mention indexed as i, the classifier was responsible for determining which of the prior mentions {1, ..., i − 1} served as the antecedent . However, consider the case where mention i is not actually anaphoric, and none of the antecedents should be selected. In such situations, a model would need to run a separate anaphoricity classifier on i.
- Instead, a more effective approach is to jointly learn anaphoricity detection and coreference together with a single loss. This integrated approach avoids the need for explicit separate classifiers, allowing the model to simultaneously capture anaphoricity and coreference information.

Mention Rank Algorithm

- In modern mention-ranking systems, for the ith mention (anaphor), we have an associated random variable y_i ranging over the values Y (i) = {1, ..., i − 1, ϵ}. The value ϵ is a special dummy mention meaning that i does not have an antecedent (i.e., is either discourse-new and starts a new coref chain, or is non-anaphoric).



Mention Rank Algorithm

- At test time, for a given mention i the model computes one softmax over all the antecedents (plus ϵ) giving a probability for each candidate antecedent (or none).
- Once the antecedent is classified for each anaphor, transitive closure can be run over the pairwise decisions to get a complete clustering.
- Training in the mention-ranking model poses challenges compared to the mention-pair model. In this model, determining which of all possible gold antecedents to use for training each anaphor is complex. The best antecedent for each mention is latent, meaning that for each mention, there exists an entire cluster of legal gold antecedents to choose from.

Mention Rank Algorithm

- Early approaches utilized heuristics to select an antecedent. For instance, one heuristic involved choosing the closest antecedent as the gold antecedent, and considering all non-antecedents within a window of two sentences as negative examples.
- The simplest approach involves giving credit to any legal antecedent by summing over all of them, employing a loss function that optimizes the likelihood of all correct antecedents from the gold clustering.

Entity-Based Model

- Both the mention-pair and mention-ranking models make their decisions about mentions. By contrast, entity-based models link each mention not to a previous mention but to a previous discourse entity (cluster of mentions).
- A mention-ranking model can be turned into an entity-ranking model simply by having the classifier make its decisions over clusters of mentions rather than individual mentions
- For traditional feature-based models, this can be done by extracting features over clusters. The size of a cluster is a useful feature, as is its ‘shape’, which is the list of types of the mentions in the cluster i.e., sequences of the tokens (P)roper, (D)efinite, (I)ndefinite, (Pr)onoun, so that a cluster composed of {Victoria, her, the 38-year-old} would have the shape P-Pr-D

Entity-Based Method

- An entity-based model that includes a mention-pair classifier can use as features aggregates of mention-pair probabilities, for example computing the average probability of coreference over all mention-pairs in the two clusters.
- Neural models can learn representations of clusters automatically, for example by using an RNN over the sequence of cluster mentions to encode a state corresponding to a cluster representation, or by learning distributed representations for pairs of clusters by pooling over learned representations of mention pairs.
- However, although entity-based models are more expressive, the use of cluster-level information in practice has not led to large gains in performance, so mention-ranking models are still more commonly used.