The Mention-Pair Architecture

Natalie Parde
UIC CS 421

The Mention-Pair Architecture

Simple premise:

- Given:
  - Pair of mentions (candidate anaphor and candidate antecedent)
- Decide:
  - Whether or not they corefer

How does this work?

- Compute coreference probabilities for every plausible pair of mentions
- Goal: High probability for actual coreferring pairs, and low probability for other pairs

The Mention-Pair Architecture

The **University of Illinois at Chicago** is an excellent place to study **natural language processing**. **UIC** has many **faculty** currently working in **NLP**, including but not limited to **Natalie Parde**, **Barbara Di Eugenio**, **Cornelia Caragea**, **Bing Liu**, and **Philip Yu**. **The school** is located in bustling downtown **Chicago**, and as **a bonus** **it** will be opening a snazzy new (non-brutalist) **CS building** in 2022.

The Mention-Pair Architecture

The **University of Illinois at Chicago** is an excellent place to study **natural language processing**. **UIC** has many **faculty** currently working in **NLP**, including but not limited to **Natalie Parde**, **Barbara Di Eugenio**, **Cornelia Caragea**, **Bing Liu**, and **Philip Yu**. **The school** is located in bustling downtown **Chicago**, and as **a bonus** **it** will be opening a snazzy new (non-brutalist) **CS building** in 2022.

The Mention-Pair Architecture

The **University of Illinois at Chicago** is an excellent place to study **natural language processing**. **UIC** has many **faculty** currently working in **NLP**, including but not limited to **Natalie Parde**, **Barbara Di Eugenio**, **Cornelia Caragea**, **Bing Liu**, and **Philip Yu**. **The school** is located in bustling downtown **Chicago**, and as a **bonus** **it** will be opening a snazzy new (non-brutalist) **CS building** in 2022.

The Mention-Pair Architecture

The **University of Illinois at Chicago** is an excellent place to study **natural language processing**. **UIC** has many **faculty** currently working in **NLP**, including but not limited to **Natalie Parde**, **Barbara Di Eugenio**, **Cornelia Caragea**, **Bing Liu**, and **Philip Yu**. **The school** is located in bustling downtown **Chicago**, and as a **bonus** **it** will be opening a snazzy new (non-brutalist) **CS building** in 2022.

How do we learn these probabilities?

- Select training samples
  - One positive instance (m_i, m_j) where m_j is the closest antecedent to m_i
  - A negative instance (m_i, m_k) for each m_k between m_j and m_i
- Extract features
  - Hand-built features, and/or
  - Implicitly learned representations
- Train classification model

How do we make predictions?

- Apply the trained classifier to each test instance in a clustering step
  - **Closest-first clustering**
    - For mention *i*, classifier is run backwards through prior *i*-1 mentions
    - First antecedent with probability > 0.5 is selected and linked to *i*
  - **Best-first clustering**
    - Classifier is run on all possible *i*-1 antecedents
    - Mention with highest probability is selected as the antecedent for *i*

Mention-Pair Architecture

- Advantage:
  - **Simplest** coreference resolution architecture
- Disadvantage:
  - **Doesn't directly compare candidate antecedents** with one another
  - **Considers only mentions**, not overall entities

How can we address these limitations?

- One option: The **Mention-Rank Architecture**
  - Directly compares antecedents with one another
  - Selects the highest-scoring antecedent for each anaphor
- How does this work?
  - For a mention *i*, we have:
    - Random variable y_i ranging over the values $Y(i) = \{1, \ldots, i-1, \varepsilon\}$
      - ε = dummy mention meaning *i* does not have an antecedent
  - At test time, for *i* the model computes a softmax over all possible antecedents
  - When training:
    - Use heuristics to determine the best antecedent for an anaphor (e.g., closest = best)
    - Or, learn more optimal ways to model latent antecedents using machine learning