Master Thesis
Entity Disambiguation using Freebase and Wikipedia

Ragavan Natarajan

Institut für Informatik,
Technische Fakultät,
Albert-Ludwigs-Universität, Freiburg
Agenda

- Definition
- Knowledge Base creation
- Identifying the relevant phrases
- Eliminating unnecessary entities
- The Collective Entity Linking algorithm
- Leaving out the irrelevant phrases
- Demo and Results
Definition
It is the process of identifying **important** phrases in a text and mapping them to **relevant entities** based on the **context of occurrences** of the phrases.
Entity Recognition and Disambiguation

Overview

Definition
It is the process of identifying important phrases in a text and mapping them to relevant entities based on the context of occurrences of the phrases.

Example
At BKC rally, BJP’s prime ministerial candidate Narendra Modi takes a dig at Rahul Gandhi for his remarks on corruption, slams Congress government in state for trying to shield its leaders by sweeping Adarsh scam case under the carpet.
Entity Recognition and Disambiguation

Abbreviations and partial name mentions

- BJP
- Congress
- Bharatiya Janata Party
- Bangladesh Jatiya Party
- Indian National Congress
- United States Congress
- National Democratic Alliance
Knowledge base creation

Sources

**Freebase**
Freebase contains tens of millions of **topics**, thousands of **types**, and tens of thousands of **properties**.*

**Example**

**Topic**: Nestlé  
**Types**: Business Operation, Candy Bar Manufacturer  
**Properties**: KitKat, Milkybar

* https://developers.google.com/freebase/index
Knowledge base creation

Sources

Freebase data dump

- Available in N-Triples RDF † format under CC-BY license.
- Uncompressed size of approximately 300GB.
- Contains subject-predicate-object expressions.

† http://www.w3.org/TR/rdf-testcases/#ntriples
Knowledge base creation

Sources

Wikipedia
Collaboratively edited free Internet encyclopedia.

Facts about English Wikipedia

- Contains 7.5 million\(^\dagger\) articles
- 11 million user identified name mentions
- 35 million distinct words excluding stopwords
- 5 million linked entities

\(^\dagger\) As on January 2014
Identifying the relevant phrases

**Link Probability**

\[ n \text{-grams of up to 10 words are generated from the input text and matched against a database of phrases.} \]

**Link Probability of a phrase** \( l(p) \)

\[ l(p) = \frac{|\text{link}(p)|}{DF(p)} \]

where, \( \text{link}(p) \) is the set of all documents where the phrase \( p \) appears as a link.
Identifying the relevant phrases

Link Probability

$n$-grams of up to 10 words are generated from the input text and matched against a database of phrases.

**Link Probability of a phrase** $l(p)$

$$ l(p) = \frac{|\text{link}(p)|}{\text{DF}(p)} $$

where, $\text{link}(p)$ is the set of all documents where the phrase $p$ appears as a link.

**Normalized Link Probability** $N_k(p)$

$$ N_k(p) = \frac{l(p)}{\sum_{p \in \mathbb{P}} l(p)} $$

where $\mathbb{P}$ is the set of all phrases in the input document.
Identifying the relevant phrases

$\text{TF} \times \text{IDF}$

$\text{IDF}(p) = \log \left( \frac{N}{\text{DF}(p)} \right)$

where, $N$ is the total number of documents in the knowledge base.
Identifying the relevant phrases

**TF × IDF**

\[
\text{IDF}(p) = \log \left( \frac{N}{\text{DF}(p)} \right)
\]

where, \( N \) is the total number of documents in the knowledge base.

**Normalized TF × IDF based importance \( I(p) \)**

\[
I(p) = \frac{\text{TF} \times \text{IDF}(p)}{\sum_{p \in D} \text{TF} \times \text{IDF}(p)}
\]

where \( D \) is the input document.
Identifying the relevant phrases
Putting it all together

Phrase Retention Score $\mathcal{R}(p)$

$$
\mathcal{R}(p) = \frac{\mathcal{I}(p) \times \mathcal{N}_k(p)}{\sum_{p \in \mathcal{P}} \mathcal{I}(p) \times \mathcal{N}_k(p)}
$$

- Experiments indicate $0.05 \leq \mathcal{R}(p) \leq 0.2$ works well, typically 0.1
Identifying the relevant phrases

Putting it all together

Phrase Retention Score $\mathcal{R}(p)$

$$\mathcal{R}(p) = \frac{\mathcal{I}(p) \times N_k(p)}{\sum_{p \in \mathcal{P}} \mathcal{I}(p) \times N_k(p)}$$

- Experiments indicate $0.05 \leq \mathcal{R}(p) \leq 0.2$ works well, typically $0.1$

Further eliminating the phrases

- Among the phrases retained by means of retention score, only a top $x\%$ of them are further retained.
- This is left as a choice to the user, as often the $\mathcal{R}(p)$ mechanism works well.
Eliminating unnecessary entities
Phrase-entity compatibility

Helps determine the potential disambiguation candidates before the actual disambiguation happens.

Compatibility Score, $CP(p, e)$

$$CP(p, e) = \frac{\vec{p} \cdot \vec{e}}{|\vec{p}| |\vec{e}|}$$

where,

- $\vec{p}$ = vector of TF×IDF scores of local context of phrase $p$.
- $\vec{e}$ = vector of TF×IDF scores of words of entity $e$.

Only the top 10 entities in terms of their compatibility scores are retained.
Relationship among entities

Entity-entity compatibility

Helps determine how semantically related are two entities to each other.

Semantic Relatedness, $\text{SR}(a, b)$

$$\text{SR}(a, b) = 1 - \frac{\log(\max(|A|, |B|)) - \log(|A \cap B|)}{\log(|U|) - \log(\min(|A|, |B|))}$$

where,

- $a, b =$ Entities of interest.
- $A, B =$ Set of articles in which $a$ and $b$ appear, respectively.
- $U =$ Set of all documents in the knowledge base.
- $\text{SR}(a, b)$ is set to zero when $A \cap B = \emptyset$
The Collective Entity Linking algorithm
due to Han et al. §

Overview

▶ Attempts to exploit the global interdependence between disambiguation decisions.
▶ Uses a so-called Referent Graph to model the global interdependence.
▶ Jointly disambiguates the phrases using a collective inference algorithm.

§http://doi.acm.org/10.1145/2009916.2010019
The Collective Entity Linking algorithm

Referent Graph

Referent Graph properties

- Is a directed graph $G(V, E)$.
- $V = \mathcal{P} \cup \mathcal{E}$, where $\mathcal{P}$ = set of phrases, $\mathcal{E}$ = set of entities.
- $(p, e) \in E$, if $p \in \mathcal{P}$ has a link to $e \in \mathcal{E}$.
- If $SR(e_i, e_j) \neq 0$ for $\{e_i, e_j\} \subseteq \mathcal{E}$, then $\{(e_i, e_j), (e_j, e_i)\} \subseteq E$ with weights $SR(e_i, e_j)$.
- $\forall e \in \mathcal{E}, p \in \mathcal{P}, (e, p) \notin E$.
The importance measure gets reinforced by means of evidence propagation.

Propagation through phrase-entity edges

\[ P(p \to e) = \frac{CP(p,e)}{\sum_{e \in N_p} CP(p,e)} \]

where,

- \( P \) is the evidence propagation ratio.
- \( N_p \) is the set of neighboring entities of phrase \( p \).
The Collective Entity Linking algorithm
Evidence Propagation

Propagation through entity-entity edges

\[ P(e_i \rightarrow e_j) = \frac{\sum_{e \in N_{e_i}} SR(e_i, e)}{SR(e_i, e_j)} \]

where,

- \( P \) is the evidence propagation ratio.
- \( N_{e_i} \) is the set of neighboring entities of phrase \( p \).
The Collective Entity Linking algorithm

Disambiguation

Let $\mathbb{P}$ be the set of phrases and let $\mathbb{E}$ be the set of entities. Let $\mathbb{E}_p$ be the set of target entities of a phrase $p \in \mathbb{P}$. Then the most relevant target entity $\mathcal{T}(p)$, of the phrase $p$, is identified as follows.

Disambiguated target of a phrase

$$\mathcal{T}(p) = \arg \max_{e \in \mathbb{E}_p} CP(p, e) \times r_d(e)$$

where,

- $r_d(e)$ is the evidence score for the entity $e$ to be a referent entity of the document $d$
The Collective Entity Linking algorithm
Computing $r_d(e)$

Algorithm

- Let $s$ be the initial evidence vector of size $|V| \times 1$ where $s_i = I(i)$ if $i \in \mathbb{P}$
- Let $M_{|V| \times |V|}$ be the evidence propagation matrix.
- Then, the evidence vector $r$ is computed as follows.

$$r = \lambda(I - cM)^{-1} \times s$$

where, $\lambda = 0.1$ and $c = 1 - \lambda$. 
The Collective Entity Linking algorithm

Complexity

Analyzing the complexity of the algorithm

Since most of the computations involving the knowledge base could be done and stored beforehand, the complexity depends mainly on the input text.

- Computation of $r$ involves a matrix inverse operation which takes $O((|P| + |E|)^3)$
- Computation of evidence propagation matrix $M$ takes $O(|P||E||D| + |E|^2)$
Leaving out irrelevant phrases

Posterior phrase importance measure

Posterior phrase importance

\[ I_{\text{post}} = I(p) \times r_d(T(p)) \]
Measuring the result
Precision and Recall

Let $\mathbb{P}_r$ be the set of phrases linked correctly, $\mathbb{P}_i$ be the set of phrases that are linked incorrectly or insignificant but included in the result, and $\mathbb{P}_u$ be the set of significant phrases that were unidentified.

Precision

$\triangleright$ Precision, $\mathcal{P} = \frac{|\mathbb{P}_r|}{|\mathbb{P}_r| + |\mathbb{P}_i|}$

Recall

$\triangleright$ Recall, $\mathcal{R} = \frac{|\mathbb{P}_r|}{|\mathbb{P}_r| + |\mathbb{P}_u|}$
Freebase taxonomy

Problems

- Has a lot of facts, but not exhaustive.
- For topics like Films, Music etc., it has a lot of associated facts.
- For topics like Brands and a lot of other topics the facts are not yet identified, even though they often exist independently.
- The algorithms that use Freebase for entity disambiguation work with a very small subset (< 1% of the volume) of Freebase carefully identified by the researchers.
- Results don’t look impressive if the entire Freebase taxonomy is used, due to the missing facts.
Questions?