Entity Linking

David Soares Batista

Disciplina de Recuperação de Informação, Instituto Superior Técnico

November 11, 2011

Motivation

Entity Linking is the process of associating an **entity mentioned in a text** to an entry, representing that entity, in a **knowledge base**

Motivation

Entity Linking is the process of associating an **entity mentioned in a text** to an entry, representing that entity, in a **knowledge base**

"There are hardly any countries here which said they were ready to go along with the EFSF (euro zone rescue fund)," German Chancellor Angela Merkel told a news conference.

Potential sovereign investors such as China and Brazil wanted to see more detail before they made any firm commitment to put money into the bailout fund.

Global stocks and the euro fell as doubts resurfaced about Europe's financial rescue package.

Motivation

Entity Linking is the process of associating an **entity mentioned in a text** to an entry, representing that entity, in a **knowledge base**

"There are hardly any countries here which said they were ready to go along with the EFSF (euro zone rescue fund)," German Chancellor Angela Merkel told a news conference.

Angela Dorothea Merkel (; née **Kasner**, born 17 July 1954 in Hamburg) is the current Chancellor of Germany (since 22 November 2005).

96% probability of being a link

Brazil wanted to see

nt to put money into the

ced about Europe's financial rescue package.

Motivation

Entity Linking is the process of associating an **entity mentioned in a text** to an entry, representing that entity, in a **knowledge base**

"There are hardly any countries here which said they were ready to go along with the EFSF (euro zone rescue fund)," German Chancellor Angela Merkel told a news conference.

Potential sovereign investors such as China and Brazil wanted to see m t to put money into the ba

G ed about Europe's fir

Brazil (; ,), officially the **Federative Republic of Brazil** (,), is the largest country in South America.

94% probability of being a link

Ambiguity

the same string can refer more than one entity

Ambiguity

the same string can refer more than one entity

"James Cook" in Wikipedia:

- 4 organizations (Universities, Institutes, etc.)
- 11 persons (British explorer, former NFL player, UK boxer, etc)

Ambiguity

the same string can refer more than one entity

"James Cook" in Wikipedia:

- 4 organizations (Universities, Institutes, etc.)
- 11 persons (British explorer, former NFL player, UK boxer, etc)

the same entity can be referred to by more than one string

Ambiguity

the same string can refer more than one entity

"James Cook" in Wikipedia:

- 4 organizations (Universities, Institutes, etc.)
- 11 persons (British explorer, former NFL player, UK boxer, etc)

the same entity can be referred to by more than one string

- *"Praça do Comércio"* and *"Terreiro do Paço"*
- *"Roger Bacon"* also known as *"Doctor Mirabilis"*

Entity Linking task in the Text Analysis Conference

Evaluate systems on the Entity Linking task: gold-standard

Entity Linking task in the Text Analysis Conference

Evaluate systems on the Entity Linking task: gold-standard

- Participants are provided with: a set of **queries**, an **external knowledge** base and a **collection of documents**
- **Goal**: For every query, return the identifier of the entity in the knowledge base, or NIL if the entity is not part of it

Example

Query

- query_string: 'Andres Velasco'
- doc_id: 'APW_ENG_20081104.0767.LDC2009T13'

Support Document:

- 'Finance Minister *Andres Velasco* joined Tuesday's announcement, saying Chile's economy is solid enough to survive "this huge international storm."'

Example

Knowledge Base entry:

```
<entity wiki_title="Christopher_Gorham" type="PER" id="E0811064" name="Christopher Gorham">
  <facts class="Infobox actor">
    <fact name="name">Christopher Gorham</fact>
    <fact name="birthdate">August 14, 1974 (1974-08-14) (age 34)</fact>
    <fact name="birthplace">
      <link entity_id="E0604659">Fresno, California</link>
    </fact>
    <fact name="spouse">Anel Lopez Gorham (2000-present) 2 children</fact>
  </facts>
  <wiki_text>
    <![CDATA[
      Christopher Gorham> Christopher Gorham (born August 14, 1974) is an American actor.
      Gorham was born in Fresno, California. He attended Roosevelt School of the Arts and
      graduated from UCLA with a BA in Film & Theater Arts. Gorham has appeared in a number
      of science fiction TV series, ranging from a starring role in Odyssey 5 to a lead role
      in Jake 2.0. He also had roles on Party of Five and Felicity. Gorham has also been in a
      few films, including the 2001 film The Other Side of Heaven co-starring Anne Hathaway.
      He also has done voice-overs for some computer and video games.
    ]]>
  </wiki_text>
</entity>
```

Example

Knowledge Base entry:

```
▼<entity wiki_title="Christopher_Gorham" type="PER" id="E0811064" name="Christopher Gorham">
 ▼<facts class="Infobox actor">
    <fact name="name">Christopher Gorham</fact>
    <fact name="birthdate">August 14, 1974 (1974-08-14) (age 34)</fact>
   ▼<fact name="birthplace">
      <link entity_id="E0604659">Fresno, California</link>
    </fact>
    <fact name="spouse">Anel Lopez Gorham (2000-present) 2 children</fact>
  </facts>
 ▼<wiki_text>
  ▼<![CDATA[
     Christopher Gorham> Christopher Gorham (born August 14, 1974) is an American actor.
     Gorham was born in Fresno, California. He attended Roosevelt School of the Arts and
     graduated from UCLA with a BA in Film & Theater Arts. Gorham has appeared in a number
     of science fiction TV series, ranging from a starring role in Odyssey 5 to a lead role
     in Jake 2.0. He also had roles on Party of Five and Felicity. Gorham has also been in a
     few films, including the 2001 film The Other Side of Heaven co-starring Anne Hathaway.
     He also has done voice-overs for some computer and video games.
    ]]>
  </wiki_text>
</entity>
```

Evaluation:

$$Accuracy = \frac{Number\ of\ Correct\ Queries}{Total\ Number\ of\ Queries}$$

Typical Approach

- **Query Processing**: generate all possible senses based on the entity string and the support document

Typical Approach

- **Query Processing**: generate all possible senses based on the entity string and the support document
- **Candidates Generation**: query the knowledge base retrieving a possible number of candidates

Typical Approach

- **Query Processing**: generate all possible senses based on the entity string and the support document
- **Candidates Generation**: query the knowledge base retrieving a possible number of candidates
- **Candidates Ranking**: comparing the candidates context with the query support document in order to decide which one is correct

Typical Approach

- **Query Processing**: generate all possible senses based on the entity string and the support document
- **Candidates Generation**: query the knowledge base retrieving a possible number of candidates
- **Candidates Ranking**: comparing the candidates context with the query support document in order to decide which one is correct
- **NIL clustering:** queries that had a NIL answer and refer to the same entity are clustered

Language Computer Corporation system - KBP 2010
1st place for Entity Linking Task

Query processing

Context independent mappings (Wikipedia)

- **Redirects and normalized article title**
- **Hyperlink anchor texts to their target pages**
- **Disambiguation pages**

Query processing

Context independent mappings (Wikipedia)

- **Redirects and normalized article title**
- **Hyperlink anchor texts to their target pages**
- **Disambiguation pages**

Context dependent mappings (source document)

- **Longer Mentions** ("Black Panthers", "New Black Panthers")
- **Soft Mentions** query("Moss") document("Carrie Ann Moss") sense("Carrie-Ann Moss")
- **Expanded Acronym**
- **Search Engine** top three results as sense candidates

Candidate Ranking

Context is modelled with "Learning to Link with Wikipedia" (Milne, D. and Witten, I.H. (2008))

- **Commonness** the number of times a term is used as the destination in an anchor link
- **Relatedness(**a, b**)** counts the overlap between outgoing and incoming links for concepts a and b
- **Relatedness(**a**)** weighted average of its relatedness to each context article

Candidate Ranking

"Depth-First-Search is an algorithm for traversing or searching a **tree** or graph [...] formally, DFS is an **uninformed search** that progresses [...] all freshly expanded nodes are added to a **LIFO stack**."

Sense	Commonness	Relatedness
Tree (woody plant)	92.82%	15.97%
Tree (graph theory)	2.94%	59.91%
Tree (data structure)	2.57%	63.26%

Candidate Ranking

- Select 16 tokens surrounding the entity mention as context terms
- **Linkability**: percentage of times the string is linked in Wikipedia
- **Context term score**: average of its linkability and relatedness

Surface Features

Focuses on query string independent of context

- **Link Probability**: percentage of anchor links which target the candidate sense

Surface Features

Focuses on query string independent of context

- **Link Probability**: percentage of anchor links which target the candidate sense

Similarity between the query string and the candidate string:

- **Dice Test**: true if Dice coefficient score passes a threshold
- **Acronym Test**: query string an acronym of the candidate string ?
- **Substring Test**: candidate or query a substring of the other ?

Contextual Features

Source document

- **Context Similarity**: stores a candidate sense's average link similarity to each of its context terms
- **Context Terms**: stores the number of context terms
- **Context Terms Weights**: the sum of all context terms scores
- **Alias Hit**: presence of an alternative alias of the sense
- **Fact Hit**: tests the presence of a related entity or fact known through a relation contained in DBpedia

Semantic and Generation Features

Semantic Features

- **Query Type**: semantic type of the query
- **Candidate Type**: semantic type of the candidate
- **Semantic Type**: true if the query and candidate have the same semantic type

Semantic and Generation Features

Semantic Features

- **Query Type**: semantic type of the query
- **Candidate Type**: semantic type of the candidate
- **Semantic Type**: true if the query and candidate have the same semantic type

Generation Features

- **Source**: true for each source that generated the sense
- **Source Count**: number of generators that recommended the sense

Other Features

- **Link Combo**: weighted average between Context Similarity and Link Probability
- **Log Link Candidate**: log of total link count to the candidate page
- **Sense Count**: number of candidate sense generated
- **Is Blog**: true if support document is a blog

Ranking Methods and NIL Sense Detection

Ranking Methods

- **Heuristic** combines contextual, surface, and semantic features into a numeric score
- **Machine Learning** a classifier is used to re-rank the top n candidates initially ranked with the heuristic

NIL Sense Detection

- **Binary Logistic Classifier**
- Training on all senses which the system ranked to position 1-3 on all training data from 2009 and 2010

Results

	ALL	NIL	PER	ORG	GPE
Heuristic	85.8	91.2	96.0	82.4	78.9
Machine Learning	86.8	92.0	95.6	85.2	79.6

Results

	ALL	NIL	PER	ORG	GPE
Heuristic	85.8	91.2	96.0	82.4	78.9
Machine Learning	86.8	92.0	95.6	85.2	79.6

- Locations often had insufficient or misleading document context
- Too many candidates, difficult disambiguation (disambiguation pages mappings)
- Successfully used *single-sense-per-discourse*

THU QUANTA - KBP 2009
2nd place for Entity Linking Task

Pre-Processing and Query Expansion

Two indexes for the Knowledge Base

- on the title of the article about the entity,
- on the whole text of the article

Use spelling correction suggestions from search engines

Pre-Processing and Query Expansion

Two indexes for the Knowledge Base

- on the title of the article about the entity,
- on the whole text of the article

Use spelling correction suggestions from search engines

Query expansion:

- acronym expansions of query in support document
- redirection links from wikipedia
- anchor text links

Candidates Generation

Two types of queries:

- **query the index on the title**: alternative names gathered in the query expansion merged by an OR retrieving the top 20 results

Candidates Generation

Two types of queries:

- **query the index on the title**: alternative names gathered in the query expansion merged by an OR retrieving the top 20 results
- **query the index on the text**: words in the query string are merged by an AND Boolean operator

Candidates Generation

Information about a query, Q:

- **Q.textRetrievalSet**: entities retrieved with the second query
- **Q.nameEntitySet**: list of named-entities in the support document
- **Q.sourceText**: the support document of the query

Candidates Generation

Information about a query, Q:

- **Q.textRetrievalSet**: entities retrieved with the second query
- **Q.nameEntitySet**: list of named-entities in the support document
- **Q.sourceText**: the support document of the query

Information about a candidate entity, C:

- **C.title**: corresponding Wikipedia title
- **C.titleExpand**: union of redirect set and the anchor text set
- **C.nameEntitySet**: named-entities in the support document

Candidates Ranking Features I

Surface features: similarity between strings (candidate/query)

- **StrSimSurface**:
  $\forall\ s \in C.titleExpand : max(Similarity(s, queryString))$

Candidates Ranking Features I

Surface features: similarity between strings (candidate/query)

- **StrSimSurface**:
  $\forall\ s \in C.titleExpand : max(Similarity(s, queryString))$
- **ExactEqualSurface**: query string $\in$ C.titleExpand ?

Candidates Ranking Features I

Surface features: similarity between strings (candidate/query)

- **StrSimSurface**:
  $\forall\ s \in C.titleExpand : max(Similarity(s, queryString))$
- **ExactEqualSurface**: query string $\in$ C.titleExpand ?
- **ContainsQuery**: query string a substring of a string in C.titleExpand ?

Candidates Ranking Features I

Surface features: similarity between strings (candidate/query)

- **StrSimSurface**:
  $\forall\ s \in C.titleExpand : max(Similarity(s, queryString))$
- **ExactEqualSurface**: query string $\in$ C.titleExpand ?
- **ContainsQuery**: query string a substring of a string in C.titleExpand ?
- **SubStringQuery**: any s $\in$ C.titleExpand which is susbstring of the query string ?

Candidates Ranking Features I

Surface features: similarity between strings (candidate/query)

- **StrSimSurface**:
  $\forall\ s \in C.titleExpand : max(Similarity(s, queryString))$
- **ExactEqualSurface**: query string $\in$ C.titleExpand ?
- **ContainsQuery**: query string a substring of a string in C.titleExpand ?
- **SubStringQuery**: any s $\in$ C.titleExpand which is susbstring of the query string ?
- **EqualWordNumSurface**: maximum number of same words between any s $\in$ C.titleExpand and query string

Candidates Ranking Features I

Surface features: similarity between strings (candidate/query)

- **StrSimSurface**:
  $\forall\ s \in C.titleExpand : max(Similarity(s, queryString))$
- **ExactEqualSurface**: query string $\in$ C.titleExpand ?
- **ContainsQuery**: query string a substring of a string in C.titleExpand ?
- **SubStringQuery**: any s $\in$ C.titleExpand which is susbstring of the query string ?
- **EqualWordNumSurface**: maximum number of same words between any s $\in$ C.titleExpand and query string
- **MissWordNumSurface**: minimum number of different words any s $\in$ C.titleExpand and query string

Candidates Ranking Features II

Context features: measure the contexts similarity

- **TFSimContext**: TF-IDF similarity (C.article, Q.sourceText)

Candidates Ranking Features II

Context features: measure the contexts similarity

- **TFSimContext**: TF-IDF similarity (C.article, Q.sourceText)
- **TFSimRankContext**: *Rank of* $C.TFSimContext^{-1}$ in CSet

Candidates Ranking Features II

Context features: measure the contexts similarity

- **TFSimContext**: TF-IDF similarity (C.article, Q.sourceText)
- **TFSimRankContext**: *Rank of C.TFSimContext*$^{-1}$ in CSet
- **AllWordsInSource**: all words in C.title exist in Q.sourceText?

Candidates Ranking Features II

Context features: measure the contexts similarity

- **TFSimContext**: TF-IDF similarity (C.article, Q.sourceText)
- **TFSimRankContext**: *Rank of* $C.TFSimContext^{-1}$ in CSet
- **AllWordsInSource**: all words in C.title exist in Q.sourceText?
- **QueryInArticle**: is C $\in$ q.textRetrievalSet

Candidates Ranking Features II

Context features: measure the contexts similarity

- **TFSimContext**: TF-IDF similarity (C.article, Q.sourceText)
- **TFSimRankContext**: *Rank of* $C.TFSimContext^{-1}$ in CSet
- **AllWordsInSource**: all words in C.title exist in Q.sourceText?
- **QueryInArticle**: is C $\in$ q.textRetrievalSet
- **NENumMatch**: number of same entity between C.nameEntitySet and Q.nameEntitySet

Candidates Ranking Features II

Context features: measure the contexts similarity

- **TFSimContext**: TF-IDF similarity (C.article, Q.sourceText)
- **TFSimRankContext**: *Rank of* $C.TFSimContext^{-1}$ in CSet
- **AllWordsInSource**: all words in C.title exist in Q.sourceText?
- **QueryInArticle**: is C $\in$ q.textRetrievalSet
- **NENumMatch**: number of same entity between C.nameEntitySet and Q.nameEntitySet
- **NENumMiss**: number of missing name entities in Q.nameEntitySet compared to C.nameEntitySet

Candidates Ranking Features II

Context features: measure the contexts similarity

- **TFSimContext**: TF-IDF similarity (C.article, Q.sourceText)
- **TFSimRankContext**: *Rank of C.TFSimContext*$^{-1}$ in CSet
- **AllWordsInSource**: all words in C.title exist in Q.sourceText?
- **QueryInArticle**: is C $\in$ q.textRetrievalSet
- **NENumMatch**: number of same entity between C.nameEntitySet and Q.nameEntitySet
- **NENumMiss**: number of missing name entities in Q.nameEntitySet compared to C.nameEntitySet

Other:

- **TypeMatch**: true if semantic type of query and candidate match

Candidates Ranking: Two Approaches

Learning-to-Rank

- First Step: listwise algorithm for candidates ranking
- Second Step: 2-class SVM Classification decides if top one is correct or not

Hidden Naïve Bayes Method

- Naïve Bayes is easy to estimate but assumption of independency is too strong
- Captures dependency among attributes

Candidates Ranking: Listwise approach

Learning:

- lists of objects: $d^{(i)} = (d_1, ..., d_n)$ associated to query $q^{(i)}$
- list of scores: $y^{(i)} = (y_1, ..., y_n)$ as ground truth
- ranking function: $f(d^{(i)}) = z^{(i)} = (z_1, ..., z_n)$
- minimization of difference with respect to the training data

$$\sum_{i=0}^{m} L(y^{(i)}, z^{(i)})$$

Ranking:

- new query: $q^{(i')}$
- associated documents: $d^{(i')}$
- use the ranking function to assign scores
- rank the documents according to the score

Results

- ListNet for learning to Rank
- 3 trained ranking models (PER,ORG,GPE)
- Total of 416 queries in the trainning set: 285 have a target entity
- Used the 285 to train the ListNet algorithm
- Used the 416 for the trainning the SVM

Results

- ListNet for learning to Rank
- 3 trained ranking models (PER,ORG,GPE)
- Total of 416 queries in the trainning set: 285 have a target entity
- Used the 285 to train the ListNet algorithm
- Used the 416 for the trainning the SVM

Run	All	Non-NIL	NIL
ListNet+SVM	0.8033	0.7725	0.8264
Bayes Method	0.7871	0.6478	0.8918

- Identify subset of KB which contains reasonable candidates for a query
- Select the most likely candidates

Identify subset of KB with candidates

Query Expansion/Processing

- Dictionary with alternative sense/names: wikipedia structure, public ontologies "same-as" relationships

Identify subset of KB with candidates

Query Expansion/Processing

- Dictionary with alternative sense/names: wikipedia structure, public ontologies "same-as" relationships
- Acronyms expansion: web crawls, wikipedia, support document

Identify subset of KB with candidates

Query Expansion/Processing

- Dictionary with alternative sense/names: wikipedia structure, public ontologies "same-as" relationships
- Acronyms expansion: web crawls, wikipedia, support document
- Surface features: string similarity (Levensthein, Jaccardi, Dice, etc)

Identify subset of KB with candidates

Query Expansion/Processing

- Dictionary with alternative sense/names: wikipedia structure, public ontologies "same-as" relationships
- Acronyms expansion: web crawls, wikipedia, support document
- Surface features: string similarity (Levensthein, Jaccardi, Dice, etc)
- How to index the Knowledge Base? Explore link structure?

Select the most likely candidates

Modelling contexts

- Context features: context similarity

Select the most likely candidates

Modelling contexts

- Context features: context similarity
- LDA: probabilistic distributions over topics

Select the most likely candidates

Modelling contexts

- Context features: context similarity
- LDA: probabilistic distributions over topics
- Desambiguate other named-entities: support document, description

Select the most likely candidates

Modelling contexts

- Context features: context similarity
- LDA: probabilistic distributions over topics
- Desambiguate other named-entities: support document, description
- Learning to Rank: Listwise

- "Overview of the TAC 2010 Knowledge Base Population Track" (H. Ji et. all, 2010)
- "LCC Approaches to Knowledge Base Population at TAC 2010" (J. Lehmann et. all, 2010)
- "Learning to Link with Wikipedia" (Milne D. and Witten I.H. 2008)
- "THU QUANTA at TAC 2009 KBP and RTE Track" (Li et. all, 2009)
- "Learning to Rank: From Pairwise Approach to Listwise Approach Cao" (et. all 2007)