The NLPR_TAC Entity Linking System at TAC 2011

Tao Zhang, Kang Liu, and Jun Zhao
Institute of Automation, Chinese Academy of Sciences
HaiDian District, Beijing, China.
{tzhang, kliu, jzhao}@nlpr.ia.ac.cn

Abstract

In this paper, our system in the KBP Entity Linking task of TAC 2011 is described. Our system mainly contains three components. 1) Entity Candidate Detector, in this component, out system identifies all the possible entities for an entity mention through a variety of knowledge sources, such as the Wikipedia Anchor Dictionary, the Web, etc. Specially, for acronym mentions, we detect their entities by expanding acronym in the source documents. 2) Entity Linker, in which the similarity between an entity mention and an entity candidate in KB is computed. The entity candidate with higher scores than a threshold will be extracted as the real entity of the mention. 3) In the final component, our system clusters all the NIL entity mentions which refer to the same entity using Hierarchical Agglomerative Clustering algorithm.

1 Introduction

The NLPR_TAC team participated in the Regular Entity Linking task in the KBP track of TAC 2011. The task of entity linking is defined as follows: given a query that consists of a name string and a source document ID, the system is required to provide the ID of the knowledge base (KB) entry to which the query refers, or a "NILxxxx" ID if there is no such KB entry. At the same time, the participator's system is required to cluster together queries refers to the same non-KB (NIL) entities and provide a unique ID for each cluster, in the form of NILxxxx. The TAC data uses news and web data as context "source document" and the KB is derived from English Wikipedia Pages. For example, given the following sentences containing the query "Michael Jordan":

> ***Michael Jordan*** *is a leading researcher in machine learning and artificial intelligence.*
>
> ***Michael Jordan*** *is a former American professional basketball player.*

An Entity Linking system should link the first 'Michael Jordan' to 'Michael I. Jordan' which is a professor in Beckley and link the second 'Michael Jordan' to 'Michael Jeffery Jordan' which is a basketball player.

Our Entity Linking system for KBP 2011 task involves three stages: 1) detecting entity candidates (e.g. both "Michael I. Jordan" and "Michael Jeffrey Jordan" are entity candidates of the entity mention "Michael Jordan"); 2) name disambiguation (e.g. given the entity mention "Michael Jordan" and its context, the system should be able to determine which Michael Jordan in KB it refers to or NIL); 3) clustering, which means the queries in the same cluster refer to the same NIL.

The challenges of entity linking are the name variation problem and the name ambiguity problem. Name variation means that an entity can be mentioned in different ways such as full name,

acronyms. For example, the entity Michael Jeffrey Jordan can be mentioned using more than 10 names, such as Michael Jordan, MJ and Jordan. The name ambiguity problem means that the same entity mention may refer to different entities in different contexts.

In order to overcome the problems caused by the above two reasons, we propose three advancements in our entity linking system. Firstly, in the entity candidates detecting stage, we split queries into two categories: regular query and acronym query. We found that expanding an acronym (all capitalized short-form word) from its context can effectively reduce the ambiguities of the acronym mention. It's based on the assumption that two variants in the same document refer to the same entity. For instance, the query "ABC" refers to more than 30 entities in KB, but with its full name "American Broadcasting Company", which is unambiguous, we can directly link to the correct entity without the needs of disambiguation. Secondly, from our observation, we found that some entity mentions are misspelling, As a result, we can't obtain their entity candidate through Wikipedia, so we try to leverage the whole web information for detecting the entity candidates through web search for these entity mentions. Thirdly, in the disambiguation stage, we use three different methods to compute the similarity between the entity mention and the entity candidates. The first is based on the entity prior knowledge, which means the possibility of an entity candidate given a query according the distribution of entity candidates. The second is based on the VSM model using tf-idf similarity between context document of the query and knowledge base (KB) text. And the last one is a combined similarity based on the entity prior probability and the tf-idf similarity. Experiment results show that the best performance of our system is the first one.

The remainder of this paper is organized as follows. Section 2 gives a related work description. Section 3 introduces the entity candidates detecting stage of our system. We present our detailed entity linking method in section 4. Section 5 presents our clustering method. The experiment results are presented in section 6.

2 Related Work

In this section, we briefly review the related work in entity linking. The crucial component of entity linking is the disambiguation stage. The essential idea is to link the query to entity in KB using discriminative feature extracted from the query's document and the entity's document. Previous work by Cucerzan [1] proposed a disambiguation approach to link an ambiguous query in a document to one entity in KB based on VSM model. The approach choose the entity with maximum agreement between the contextual information extracted from Wikipedia and the context of entity mention document, as well as the agreement among the category tags associated with the entity candidates. Bunescu and Pasca [2] regard this task as a classification problem and train a SVM classifier to disambiguate the entity candidates. They incorporated entity candidates' category information to help improve the accuracy of the classifier. Dredze et al. [3] employed a SVM ranking method to disambiguate entity candidates. They used a comprehensive feature set to accomplish the entity linking task. Because the supervised learning method usually requires lots of training data, Zhang et al. [4] proposed a novel method to automatically generate a large scale corpus annotation for ambiguous mentions leveraging on their unambiguous synonyms in the document collection. Radford et al. [5] used a Graph-Based ranking method in entity linking task, in their approach, context entities are taken into account in order to reach a global optimized solution together with the query entity. Nemeskey et al. [6] considered using information retrieval method to disambiguate queries, in their approach, the entire background source document is considered as a single query, the task is to retrieve the most relevant Wikipedia article. Han and Sun [7] proposed an entity mention model, which can leverage heterogeneous entity knowledge for the entity linking task.

3 Entity Candidate Detector

Entity candidate detector finds possible KB entity candidates for the given query. We split query into two classes: regular query and acronym query. For different query type, our system uses different detection strategies. We find entity candidates of regular query through two ways: 1) using

Wikipedia dictionary; 2) leveraging web information. And for acronym query, we expand it from its context to disambiguate it. In the following, we respectively describe them in detail.

3.1 Entity Candidates Detection Using Wikipedia Dictionary

Wikipedia contains many name variants of entities like confusable names, spelling variations, nick name etc. we use the anchor dictionary of Wikipedia to detect the entity candidate. For example, in Figure 1, the three anchor texts of ABC are respectively its full name "American Broadcasting Company", acronyms "ABC" and "American Broadcasting Companies, Inc". Using the anchor information in Wikipedia, we can build a table between query and entity candidate. Also, we can obtain the count information that the entity using the query as anchor in Wikipedia. This information is also called the entity prior knowledge. It represents the likely of an entity candidate given a query. This information is used in our first run. Part of the table is shown in table 1.

The ***American Broadcasting Company*** is an American commercial broadcasting television network.

ABC is owned by the Walt Disney Company.

The Formal name of the operation is ***American Broadcasting Companies, Inc***.

Figure1. Three anchors of entity ABC

query	Entity candidate	The number that the query link to the entity candidate
ABC	American Broadcasting Company	10023
	Australian Broadcasting Corporation	1452
	ABC (band)	196
	ABC Television	99
	…	…
AI	Artificial intelligence	657
	Game Artificial intelligence	57
	Ai (singer)	8
	Strong AI	3
	…	…
…	…	…

Table1. Part of the query entity table

Using the table, we can easily retrieve the entity candidates of a given query through string matching in the table.

3.2 Entity Candidates Detection Using the Web

Through the table, there are still some queries which cannot find entity candidates in the table. We believe there are two reasons. The first one: the real target entity of a query is not in KB, in this situation, the output should be NIL for this query. The second one, some queries are misspelled infrequently used, so that our system cannot identity the mention in the query as a variation. Specially, for the second reason, we try to leverage the whole web information for detecting the entity candidates through web search for query whose entity candidates set is empty. Given a query whose entity candidates set is empty, we submit to the yahoo search engine with the form "query Wikipedia", and then retrieve the first web page if the page is a Wikipedia page. And then if the first page is a Wikipedia page, we extract its title as the query's candidate entity. Otherwise, we return NIL for this query. This contributes to a performance improvement on the KBP 2009 test data. For example, the search results of the query Angel Merkel is shown in Figure 2, we can detect the entity Angela Merkel for this query.

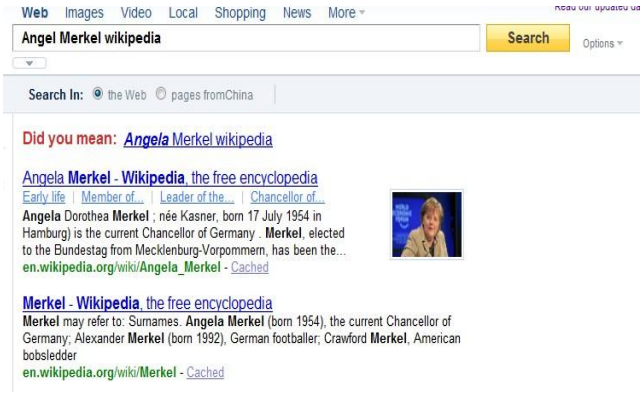

Figure2. Web Search Results of Angel Merkel + Wikipedia

3.3 Acronym Expansion

In this section, we describe our algorithm for finding an acronym's expansion in a source document for an acronym query.

From our observation, acronym is usually high ambiguous, but its full name, is usually unambiguous. Hence, expanding an acronym from its context can effectively reduce the ambiguities of an acronym query, under the assumption that two variants in the same document refer to the same entity. Han and Zhao [8] only allow expansions adjacent to the acronym in parenthesis Zhang et al. [9] propose a supervised learning algorithm to expand more complicated acronyms encountered. But this method requires a lot of training data, and because of the statistic classifier, its efficiency is not very high. We propose a simple method based on some heuristic rules and the table between the entity mention and entity candidates in our system.

Our acronym expansion method is based on the assumption that for any acronym, there must be its other name variant in other place in the document. Given a document D and an acronym A, which are usually capitalized words, we want to find its full name in this document. We employ two heuristic rules in our system.

First, we check if the document contains pattern "(A)", if the document contains the pattern "(A)", we extract the n contiguous sequence of tokens that start with the acronym's first letter and do not contain punctuations or more than 2 stop words before the pattern "(A)" as the target entity, the n represents the number of the letter in A. For example in Figure 3, we extract the target entity for the query ABC is All Basotho Convention.

Thabane, once seen as Prime Minister Pakalitha Mosisili's heir apparent, quite the cabinet to form the ***All Basotho Convention*** (ABC) last October with a populist pledge to fight hunger, poverty, disease, crime and corruption.

Figure3. An example for acronym expansion using heuristic rule one

Second, for the acronym query that does not contain the pattern "(A)" in its source documents, we process it as follows. We search the query entity table obtained in section 3.1, and identify all the entity candidates of the acronym query and obtain their prior probability. We process these entity candidates according to their prior probability. The biggest prior probability entity candidate is processed firstly. For each entity candidate, we obtain all its anchors from the query entity table, we judge whether the source document contain the anchor, If the source document contain the anchor and the anchor is not acronym, the system select the entity candidate as the target entity, otherwise, the system process other anchors of the entity candidate. For example in Figure 4, give the acronym query UT and its source document, our system first identify the entity candidates Utah, University of Texas at Austin, University of Tennessee et al. of the acronym query UT. And we process them according to their prior probability. For the entity candidate University of Texas at Austin, we obtain all its anchors University of Texas at Austin, University of Texas, and we found that the source document contain the anchor university of Texas, so we return the entity University of Texas at Austin as the query's target entity.

Cuomo's office on Wednesday sent a subpoena to Columbia University and set letters to the University of Southern California and the ***University of Texas*** seeking information about financial aid officers ownership of stock in a loan company that appears on each school's list of preferred lenders.

Figure4. An example for acronym expansion using heuristic rule two

4 Entity Linker

Given the entity candidates of the query, we need to link the query with the entity it refers to. The role of the entity linker is to measure the similarity between the query and each entity candidate. Based on the computed similarity, we select the most similar entity candidate as the query's target entity. We use three different methods to compute the similarity between the query and the entity candidate. These three methods represent out three runs for the task.

4.1 The Similarity Based on Prior Knowledge

The first method computes the similarity based on the entity candidate's prior knowledge. The intuition behind this method is that the more number of links that the query is used as a link to entity candidate in Wikipedia, the more similar between the query and the entity candidate. The

prior probability of an entity candidate is computed as follows:

$$P_{prior} = \frac{N}{M}$$

The N represents the number of links that the query is used as a link to the entity candidate in Wikipedia; the M represents the total number of links that the query is used as a link in Wikipedia. We simply select the entity candidate with the largest prior probability as the target entity.

4.2 The Similarity Based on VSM Model

The second method computes the similarity based on VSM model. The intuition behind the basic vector space model (VSM) is that the more similar (based on the word co-occurrence information) between the entity candidate text and the query's source document text, the more likely the query refers to the entity candidate. Using the VSM model, both the entity candidate and the query are represented as a vector of word features, and each word is weighted using the standard TF-IDF measure. Thus, given the vector representation of the entity candidate and the query, we use the cosine similarity between vectors as the similarity between the entity candidate and the query.

$$Sim_{vsm} = \frac{\overrightarrow{v(e)} * \overrightarrow{v(q)}}{\left|\overrightarrow{v(e)}\right| * \left|\overrightarrow{v(q)}\right|}$$

Once the similarity is computed, we select the entity candidate with the largest similarity. In order to distinguish NIL from non-NIL, we use a threshold T to determine if the query refers to NIL. If the largest similarity is lower than T, the system returns NIL as the query's target entity.

4.3 The Hybrid Similarity Based on Prior Knowledge and VSM Model

The third method computes the similarity based on a hybrid model which combine prior knowledge and VSM model together. As mentioned above, the prior knowledge method capture the entity candidate's prior probability in Wikipedia, and the VSM model method reflect the word co-occurrence information. So we can derive a hybrid similarity which combines both the above similarities for achieving better entity linking performance. The hybrid similarity is computed as follows:

$$Sim_{hybrid} = p_{prior} * sim_{vsm}$$

5 NIL Queries Cluster

Given the NIL queries determined by the above two stages, we need to cluster together queries referring to the same entities and provide a unique ID for each cluster. First, we cluster NIL queries based on their query representations. The same query representations are believed to belong to the same cluster. And the queries whose representations contain the other are believed to belong to the same cluster. And then, we use hierarchical agglomerative clustering (HAC) algorithm to cluster NIL queries in each cluster determined by the first stage. This algorithm works as follows: Initially, each query is an individual cluster; then we iteratively merge the two clusters with the largest similarity value to form a new cluster until this similarity value is smaller than a threshold. We employ the average-link method to compute the similarity between two clusters. The similarity between queries is determined by the VSM model.

6 Results

KBP 2011 entity linking task contains 2250 queries. They use the B-Cubed+ and B-Cubed+ recall to evaluate the results. We submitted 3 runs. The description of our 3 runs is shown in Table 2. The HAC clustering threshold of these 3 run3 is 0.1. The results of our 3 runs are show in Table 3

Run	Description
NLPR_TAC1	Rank the entity candidates based on their prior probabilities
NLPR_TAC2	Rank the entity candidates based on their VSM similarities, threshold is 0.1
NLPR_TAC3	Rank the entity candidates based on their hybrid similarities

Table2 Description of our 3 runs

Run	Micro-average	B^3 Precision	B^3 Recall	B^3 F1
NLPR_TAC1	0.680	0.659	0.619	0.638
NLPR_TAC2	0.564	0.543	0.525	0.534
NLPR_TAC3	0.623	0.595	0.609	0.602

Table3 Results of our 3 runs

From results in Table3, we found that our first run obtain the best result. Compared with the other two runs, the first run did not rely on any information from the query's source document, this indicates that the prior probability is a very important feature in entity linking task. The results also show that our clustering method needs to be improved. The similarity based on the VSM model seems not be an effective similarity measure in clustering stage.

Acknowledgments

The work is supported by the National Natural Science Foundation of China under Grants no. 61070106 and 60875041.

References

[1]Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In: Proceeding s of EMNLP-CoNLL..

[2]Bunescu and Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of EACL.

[3]Dredze et al. 2010. Entity Disambiguation for Knowledge Base Population. In: Proceeding of International Conference on Computational Linguistics.

[4]Zhang et al. 2010. Entity Linking Leveraging Automatically Generated Annotation. In: Proceeding of International Conference on Computational Linguistics.

[5]Radford et al. 2010. CMCRC at TAC 10: Document-level Entity Linking with Graph-based Re-ranking. In: Proceeding TAC 2010 Workshop.

[6]Nemeskey et al. 2010. BUDAPESTACAD at TAC 2010. In: Proceeding TAC 2010 Workshop.

[7] Han and Sun. 2010. A Generative Entity-Mention Model for Linking Entities with Knowledge Base. In: Proceeding of ACL.

[8]Han and Zhao. 2009. NLPR_KBP in TAC 2009 KBP Track: A Two-Stage Method to Entity Linking. In: Proceeding of Text Analysis Conference.

[9]Zhang. et al. 2011. Entity Linking with Effective Acronym Expansion, Instance Selection and Topic Modeling. In: International Joint Conference on Artificial Intelligence.

.