The NLPRIR Entity Linking System at TAC 2012

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, we describe our KBP Entity Lining system at TAC 2012. Our system consists of three modules. 1) Query expansion and candidate entity selection module. In this module, we identify all the possible entities for an entity mention through a variety knowledge sources. 2) Entity disambiguation module. In this module, we use a maximum margin approach to rank the candidate entity. 3) NIL clustering module. In this module, we cluster all the NIL entities which have been detected in the second module using Hierarchical Agglomerative Clustering alrorithm.

1 Introduction

The nlprir team participated in the regular entity linking task in the KBP track of TAC 2012. The aim of this track is to automatically discover information about named entities and to automatically maintain the existing knowledge base. 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. In order to maintain the knowledge base, the system is required to group the NIL mentions which refer to the same entities together. The TAC data use news and web data as source document and the KB consists of over 800,000 entities and is derived from the Wikipedia dump from October 2008. For example, given the following two queries which contain the name string "Michael Jordan":

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

The system is required to disambiguate which entity in the knowledge base that "Michael Jordan" refers to based on the context of document in which it appears.

In our system, entity linking is done through 3 steps; 1) Expanding entity mention from the source document and select candidate entities. (e.g. both "Michael I. Jordan" and "Michael Jeffrey Jordan" are candidate entities of the entity mention "Michael Jordan") 2) Selecting a entity using a supervised learning to rank method based on a variety of evidence or return NIL. 3) Clustering all NIL queries.

The reminder of this paper is organized as follows. Section 2 introduces the candidate selection stage of our system. The entity disambiguation module is presented in section 3. Section 4 presents our clustering method. The experiment results are presented in section 5.

2 Candidate Entity Selection

2.1 Query Expansion

In the candidate entity selection module, we find possible candidate entity for the given query. From our observation, acronym query is usually high ambiguous, but its full name, is usually unambiguous. Thus, expanding the query from its document can effectively reduce the ambiguities of the query. For example, *ABC* in Wikipedia refers to more than 20 entries, but with its full name *American Broadcasting Company*, which is

unambiguous, we can safely link it with the correct entity without disambiguation. Thus, in the first stage of our candidate entity selection module, we first expand acronym query from the document to reduce the ambiguities.

Our acronym expansion method is the same as in our 2010 system. Firstly, for the capitalized query A, 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 represent the number of the letter in A. The initiation of this method is based on the observation that the full name usually appear before the acronym in the first occurrence of this acronym to help reader to understand the real meaning of the acronym query. For example, in figure 1, using the above acronym expansion method, we expand the full name of the acronym query "ABC" as "All Basotho Convention".

Query name: ABC

Query Document: 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.

Figure 1. An example for acronym expansion using heuristic rule

The second query expansion method we adopt is based on the observation that some queries usually contain part of its full name, thus if a query is wholly contained in a string of a named entity in the associated document, we use the named entity as the full name of the query. For example, in figure 2, we find the full name of entity mention "Bryant" as "Kobe Bryant".

Query name: Bryant

Query Document: Did it make the Lakers the immediate favorites in the Western Conference?Not necessarily, but that's not the point. Right now, the Lakers' mid- to long-term prospects look better than anyone else's in the NBA. And that's the real benefit, especially if they just increased the chances that **Kobe Bryant** will be part of their future.

Figure 2. An example of query expansion rule using named entity recognition method

After the query expansion, we use the full name instead of the query in the following processing.

2.2 Candidate entity generation

Given the entity mention, we find the possible candidate entity using Wikipedia knowledge and string matching method.

Wikipedia knowledge: We use the four knowledge sources in Wikipedia: "entity pages", "disambiguation pages", "redirect pages", and "anchor text" to find candidate entity.

Each entity page in Wikipedia describes a single entity. The title of the page represents the most common name for the entity. We select the entity as the candidate entity if the entity mention is an exact match with the title.

A redirect page is an aid for navigation. When a page in Wikipedia is redirected, it means an alternative name for an existing entity in Wikipedia. Thus, we select the entity as the candidate entity if the entity mention is exactly match of the alternative name of the entity. For example, "United States of America" is the full name of the "United States", it is therefore an alternative for the entity "United States". Therefore, for entity mention "United States of America", the entity "United States" should be a candidate entity.

Disambiguation page is created for ambiguous names that denote two or more entities. It contains a list of references to pages for these ambiguous entities that share the same name. All the entities listed in the disambiguation page are considered as candidate entities if the entity mention matches the title of the disambiguation page.

The last knowledge source we use in Wikipedia is "anchor text". The article in Wikipedia contains hyperlinks that are associated with anchor texts and their entities. We thus select the entity as the candidate entity if the entity mention is a match with the anchor text.

String Match: Through the Wikipedia knowledge, there are still some entity mentions which cannot find the candidate entities. We found that the main reason behind this is that the entity mention for this query is misspelled. To find more candidate entities and resolve this problem, we select the entity which has a high bigram Dice coefficient with the entity mention. For example, the dice coefficient between "Organisation of the Islamic Conference" and "Organization of the Islamic Conference" is 0.957. We can select "Organisation of the Islamic Conference" as the candidate entity for the entity mention "Organization of the Islamic Conference". Also, for the entity mention "Angel Merkel", we can find "Angela Merkel" as its candidate entity. The threshold for the dice coefficient is set to be 0.6 in our candidate entity selection module. We did not optimize the threshold, although the threshold could be tuned to minimize the candidate entity set and maximize recall.

To evaluate our candidate entity selection module, we evaluate the coverage of the candidate entity set in TAC-KBP track 2009 data and TAC-KBP track 2010 data. In table 1, we show the recall of the candidate entity in the two data sets.

Table 1: recall of candidate entity selection in the two data sets

Data Set	TAC 2009	TAC 2010
recall	0.9398	0.9298

3 Candidate Entity Disambiguation

In candidate entity disambiguation module, our system selects a single entity using a supervised learning to rank method. In the learning to rank method, each entity mention and the associated candidate entity is formed by a battery of features representing contextual, semantic, and surface evidence. During the ranking, each candidate entity is given a score based on the feature. Our system thus selects the entity with the highest score as the answer.

We use ranking SVM as our learning method. The intuition behind this is that the correct entity should receive a higher score than all other possible entities. This learning constraint is equivalent to the ranking SVM algorithm of Joachims (2002). We use a linear kernel, set the slack parameter C as 200, and take the loss function as the total number of swapped pairs summed over all training examples. The kernel function we used is a linear kernel. Compared with other kernel function, linear kernel has the advantage in efficiency.

3.1 Features for Entity Disambiguation

In the learning to rank stage, we use a total of 6 features to represent the candidate entity. We will introduce them in detail in the following section.

NIL feature: We add a special entity nil into the candidate entity. Our institution is the same as in Dredze (2010). We learn when to predict NIL using the SVM ranker by adding nil entity into the candidate entity set. This is equivalent to learning a single threshold across quires. The ranker can set the threshold optimally without hand tuning. In the training stage, the NIL feature is set to be 1.0. In the ranking stage, we found that 1.0 is not a good choice. This has caused the relative low performance of our ranker. The ranker selected too many NIL as the answer. Perhaps the reason behind this is that the training date has too many NIL entity. Thus, after the training process, the weight of the NIL feature is too big compared with other features. We found that setting the NIL feature to be 0.7 is a good choice in the development set. .

Dice Coefficient between query name and candidate entity: If the query name and the candidate entity have a high dice coefficient, this is a strong indication of a match. Thus, we compute the dice coefficient score between the entity mention and the title of the candidate entity as a feature. For nil entity, this feature is set to be 0. For the candidate contain parenthetical expression, we compute their dice coefficient after removing the parenthetical. For example, the dice coefficient between "Michael Jordan (football player)" and "Michael Jordan" is 1. For acronym query name, we also compute the dice coefficient between the acronym and the first letters in the candidate entity. And select the bigger dice coefficient as the feature. For example, the dice coefficient between "ABC" and "American Broadcasting Company" is 1.

Entity mention feature: This type of feature is based on the presence of names in the text. It includes two features: Whether the title of the candidate entity appears in the document of the query; whether the entity mention of the query

appears in the KB text. For the NIL entity, these two features are set to be 0.

Link probability of candidate entity: The link probability is based on the percent of entity mention string link to the candidate entity in Wikipedia. It indicates the likely of the candidate entity without any knowledge. The more number the entity mention links to the entity, the more likely the entity is a correct match. For example, given "Michael Jordan", without any background knowledge, the Michael Jordan (Basketball player) has a high probability to be the answer compared with other Michael Jordan. The link probability can be viewed as a prior knowledge. For some candidate entities, the query name may have no link to these entities. In this situation, the link probability is 0. Again, for the nil, this type of feature is also set to be 0.

Semantic relatedness of Candidate entity: This type of feature stores a candidate entity's average semantic relatedness to each of the concept in the document of the query. First, we recognize the Wikipedia concept using the Wikipedia-Miner toolkit. The Wikipedia-Miner toolkit takes the general unstructured text as input and use machine learning approach to detect the Wikipedia concepts in the input documents. For the given query text, we firstly remove the entity mention from the query text, and then utilize the Wikipedia-Miner toolkit to obtain the Wikipedia concepts. This toolkit first scans the text, select candidate concept based on the string match. In the selecting of the Wikipedia concept, each Wikipedia concept is given a score based on the link probability and relatedness. Link probability is the percentage of this term is used as an anchor in Wikipedia. Relatedness is the average link similarity to all other candidate concept. After obtaining the concepts in the document, we compute the semantic relatedness score between the candidate entity and the concept in the query text.

The relatedness score between two concepts is defined as follows:

$$sr(c_i, c_j) = 1 - \frac{log(max(|C_i|, |C_j|)) - log(|C_i \cap C_j|)}{log(|W|) - log(min(|C_i|, |C_j|))}$$

where c_i and c_j are two Wikipedia articles, C_i and C_j are the sets of all articles that link to c_i and c_j respectively, and W is set of all Wikipedia articles.

The average link similarity of an candidate entity is then computed as:

$$f(e) = \frac{\sum_{i \epsilon c} sr(c_i, e)}{n}$$

where e is a candidate entity, c_i is a Wikipedia concept in the query text recognized by the Wikipedia miner toolkit, n is the number of the Wikipedia concepts in the document of the query. For the nil, this feature is set to be 0.

Similarity based on VSM model: This feature captures the similarity between the document of the query and the document of the candidate entity based on the VSM model. The intuition behind this is that the more similar between the document of the candidate entity and the document of the query, the more likely the entity mention refers to the candidate entity. Using the VSM model, both the candidate entity and the query are represented as vector of word features. Each word is weighted using the standard TF-IDF measure. Thus, given the vector representation of the candidate entity and the query, we use the cosine similarity between vectors as the feature.

4 NIL Clustering

Given the NIL queries determined by the above two modules, the system is required 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 entity mentions. The dice coefficient is used to determine which two queries maybe refers to the same entity. The two quires which the dice coefficient between them has higher than 0.6 are believed to belong to the same cluster. And the queries whose entity mentions 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 similarity between the Wikipedia concepts.

$$sr(q_i, q_j) = \frac{\sum_{c_i} \sum_{c_j} sr(c_i, c_j)}{n * m}$$

Where c_i and c_j are Wikipedia concepts in the document of query i and query j respectively. n and m represent the number of Wikipedia concepts in the query i and query j respectively.

5 Results

The KBP 2009 dataset is used as our training data. The KBP 2009 dataset contains 3904 queries which are selected from English newswire articles. The KBP 2010 dataset is used as our development data. The dataset contains 2250 queries and query document come from news wire and Web pages. The NIL feature is set to be 0.7 in the test data. KBP 2012 entity linking task contains 2226 queries. They use the B^3+ F1 to evaluate the results. We submitted 1 run. The result of our run is show in Table 2.

Table 2: result of our entity linking system

Evaluation metric	results
B^3+ F1 (All -- 2226 queries)	0.562
B^3+ F1 (in KB -- 1177 queries)	0.477
B^3+ F1 (not in KB -- 1049 queries)	0.657
B^3+ F1 (NW docs -- 1471 queries)	0.597
B^3+ F1 (WB docs -- 755 queries)	0.495
B^3+ F1 (PER -- 918 queries)	0.691
B^3+ F1 (ORG -- 706 queries)	0.517
B^3+ F1 (GPE -- 602 queries)	0.414

We found our entity linking system can achieve competitive results. The lowest performing category of queries is GPE. The reason behind is that we found the GPE query often has misleading document context. Also, the query expansion method does not work well for the GPE query.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61070106), the National Basic Research Program of China (No. 2012CB316300), the Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDA06030300) and the Tsinghua National Laboratory for Information Science and Technology (TNList) Cross-discipline Foundation.

References

D. Milne, Ian H. Witten. Learning to Link with Wikipedia. In Proc. of CIKM, 2008.

D. Milne and Ian H. Witten. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In Proc. of AAAI, 2008.

M. Dredze, P. McNamee, D. Rao, A. Gerber and T. Finin. Entity Disambiguation for Knowledge Base Population. In proc. of Coling 2010.

T. Joachims. Optimizing search engines using clickthrough data. In Knowledge Discovery and Data Mining. In proc. of KDD 2002.