THUNLP at TAC KBP 2013 in Entity Linking

Yan Wang, Yankai Lin, Zhiyuan Liu, Maosong Sun
State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology
Tsinghua University, Beijing 100084, China
yan_wang10@126.com, {mrlyk423, lzy.thu}@gmail.com, sms@tsinghua.edu.cn

Abstract

This paper describes the THUNLP system submitted to Entity Linking task of KBP Track in TAC 2013. This system achieves the third place with a B-cubed+ F1 score of 0.712 in mono-lingual entity linking task. In Chinese cross-lingual entity linking, this system achieved B-cubed+ F1 score of 0.647 with the help of Google Translate.

1 Introduction

Entity linking is a task to map mentioned names in certain context to entities in a given knowledge base. It is complicated due to the diversity of context type and variety of names that refer to the same entity. The TAC entity linking task was first introduced in 2009 (McNamee and Dang, 2009). The mentioned names could represent persons (PER), geo-political entities (GPE) or organizations (ORG). The queries might come from newswire, web text or discussion fora. Queries from user-generated content are especially hard because of the misspelling, usage of nicknames and lack of rich context.

The system THUNLP is designed for this task with several strategies optimized for queries from user-generated content. In this paper, this system will be described in details, including the overall framework, specific strategies and final performance.

2 System Description

2.1 Overview

The pipeline of THUNLP system is shown in Figure 1. It is kind of a combination of framework in (Cucerzan, 2011) and (Lehmann et al., 2010).

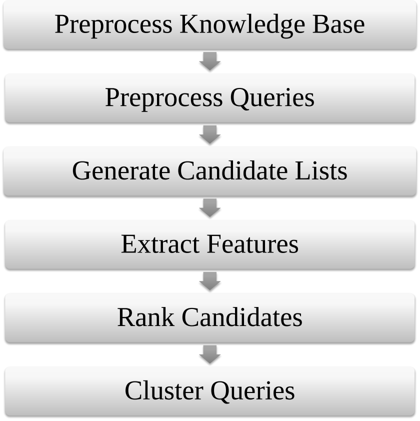

Figure 1: The pipeline of our system.

To avoid the difficult task of NIL detection, we adopt a knowledge base from a newer Wikipedia dump[1]. Denote $KB_{new} = \{E\}$ and $KB_{old} = \{e\}$ as set of entities in this new knowledge base and the target old knowledge base provided in this task respectively. Over 95% of entities in KB_{old} could be mapped to KB_{new} by simply title matching. For a set of queries $Q = \{q_i\}$, assume that we could find a list of candidate entities from KB_{new} and finally select the top one E_i. If E_i could be mapped back

[1] http://dumps.wikimedia.org/enwiki/20130708/

to KB_{old}, then the corresponding entity e_i in KB_{old} would be returned. Otherwise, a NIL cluster defined by E_i would be returned. If the selection of E_i is correct, then the result should be correct as well.

However, the underlying assumption is not very much reliable. The selection of E_i for q_i could be wrong, so the system could either link a NIL query to a wrong entity, or assign a non-NIL query with a NIL cluster. To deal with this problem, we apply NIL clustering after ranking the candidates. In the following subsections, the stages in the pipeline will be further described.

2.2 Preprocessing Wikipedia

Surface forms, a terminology mentioned in (Cucerzan, 2011), means all strings that could be used to refer to a certain entity. We discover surface forms for entities from the following sources.

- **Disambiguation pages**. All entities included in the disambiguation page could have a surface form of the title of disambiguation page. For example, both "Florida State University" and "Former Soviet Union" mentioned in disambiguation page with title "FSU", then "FSU" is a surface form of both entities.

- **Redirect pages**. It is assumed that a redirect relation links two pages that represent the same concept. For example, page "People's Republic of China" is redirected to page "China", so "People's Republic of China" is a surface form of entity "China".

- **Anchors of outlinks**. There are many outlinks in the articles and the anchor text could be surface form of the target entity. For example, the link of "Carnegie Foundation for the Advancement of Teaching" has anchor text of "Carnegie Foundation".

We build index of surface forms in lower case, ignoring all non-English characters. This index is used in candidate generation.

Besides surface forms, we also train a support-vector regression model with LibLinear to predict the type of entities. The type of an entity is represented by a vector, $(p(PER), p(GPE), p(ORG))$, in which $p(PER)$ is the probability that the entity represents a person, and $p(GPE)$ and $p(ORG)$ are interpreted similarly. This vector is normalized to ensure $p(PER) + p(GPE) + p(ORG) = 1$. The training data is selected from the old wikipedia, and the feature vector is *one-hot* representation of attribute keys of infobox.

2.3 Preprocessing Queries

We apply local search and name clustering to find possible aliases for a mentioned name. Aliases are helpful in both candidate generation and feature extraction.

Local search is based on rules and heuristic search methods rather than machine learning algorithms. It mainly deals with acronyms and incomplete names. There are over 15% queries in 2013 evaluation data set with acronym mentions. Given a document that contains an acronym, the full name that is referred by this acronym might appear as well. Especially in newswire, the full name often appears around the first occurrence of corresponding acronym. Take query EL13_ENG_1370 as an example, where "ISF", the query mentioned name, is the acronym of "International Ski Federation". Apart from acronyms, local search also benefits queries of incomplete names, especially names of people. As a person is often referred to by only first name or last name, a full name is of great help to search in the knowledge base. There are still difficult cases for this method, such as query EL13_ENG_0171 with a mentioned name "Becks" . Though "Victoria Beckham" also appears in this post, it is hard to confirm that they refer to the same person.

After local search, we cluster names to find possible aliases for queries. This strategy helps to correct spelling errors, such as query in which "Cairo" is spelled as "Ciaro" by mistake. In addition, finding similar mentioned names is also part of preprocessing of clustering process after ranking. The reason will be introduced in Section 2.6.

2.4 Candidate Generation

Candidate generation plays an important role in the pipeline. On the one hand, the recall of candidate generation directly limits system performance. On the other hand, the constitution of candidate lists decides the training data and test data of learning-to-rank model, thus indirectly influence the ranking performance. This process is about the balance between recall and average size of candidate lists. Intuitively, to achieve a high recall rate, candidate lists tend to grow larger and stronger noise is introduced. We test our strategy on previous evaluation queries and the results are shown in Table 1. When computing recall, we define a query is recalled when it is a non-NIL query and the expected entity is included in the corresponding candidate list.

Data Set	Recall	Average Size
2009	0.934	21.1
2010	0.970	44.0
2011	0.958	35.3
2012	0.955	61.5

Table 1: The recall and average candidate list size on previous evaluation queries.

Our strategy finds 59.7 candidates for each query in 2013 Evaluation Queries on average.

2.5 Ranking

ListNet is a successful listwise learning-to-rank algorithm proposed in (Cao et al., 2007). It is proved to be effective in entity linking task in (Zheng et al., 2010) as well. The top-1 ranking version of ListNet is easy to implement and could converge quickly. The training data is composed of non-NIL queries of which the correct entity is recalled in previous evaluation data sets. At this stage, we could test the performance of our system. The model trained on queries in 2009, 2010 and 2011 Evaluation Queries achieved a B-cubed+ F1 score of 0.668 on 2012 Evaluation Queries, while the accuracy on training data is 0.747. Then the final model was trained on all of four previous evaluation data sets with an accuracy of 0.763.

We evaluate features to represent each candidate entity of each query. In this process, we apply SENNA[2], a software that could output a host of Natural Language Processing predictions, including part-of-speech (POS) tagging, chunking, named entity recognition (NER) and so on. Its output of NER task helps to recognize other named entities around the query, and identify the type of the query, PER, GPE or ORG.

To put it simple, in the following descriptions, the term *document* refers to the document which a query belongs to, and the term *article* refers to the descriptive article of a wikipedia entity.

- **Title Match**. An integer value of lexical distance between the mentioned name of query and the title of entity.
- **Title in Document**. Assign 1 if the title or one of surface forms of the entity appears in the document. Assign 0 otherwise.
- **Surface Form Frequency**. An integer number of times that the surface form by which the candidate entity is found actually refer to this entity.
- **Type Similarity**. A real value within [0, 1]. Assign each query with a type predicted by SENNA, and the corresponding dimension of the entity's type vector is the type similarity.
- **BOW Similarity**. A real value within [0, 1] that computed as the cosine similarity between bag-of-words vectors of the document and the article.

Apart from the features listed above, there are two more features that play important role, link compatibility and category compatibility. These two features are evaluated in the similar way. For query q with a list of candidates $\{c_i\}$, SENNA could label other named entities in the same document, denoted as $\{n^{(j)}\}$. In most situations, a list of candidates $\{c_k^{(j)}\}$ could be found by $n^{(j)}$ in the index of surface forms. Assume that there is a measure of similarity between two entities, and the compatibility of a set of entities could be the average similarity of all entity pairs. For each candidate in $\{c_i\}$, we could pick one entity from every $\{c_k^{(j)}\}$ and form a new set of entities with maximum compatibility. Then this compatibility value is a feature of candidate c_i. In practice, we implemented a greedy algorithm to compute the approximate compatibility of a set of entity because the complexity grows so fast when $\{n^{(j)}\}$ is large.

Link compatibility and category compatibility are calculated in the same framework but with different ways to measure the similarity of a pair of entities. Most articles of entities consist of outlinks and

[2] http://ml.nec-labs.com/senna/

category tags. When evaluating link compatibility, each entity is represented by a bag-of-words vector of outlinks in the vector space of all entities, and the similarity between entities are computed as the cosine similarity between two vectors. For category compatibility, similarly, entities are represented by bag-of-words vector of category tags. These two features depict the compatibility of the query with the whole article. The weights of these two features learned by ListNet are positive and large.

2.6 Clustering

Given the ranking results, we further adjust them by clustering similar queries. An SVM classifier is trained to judge whether a pair of queries should be linked to the same entity. Here we only consider queries with similar mentioned names to reduce the number of pairs to classify. The training data is constructed from previous evaluation data sets, including both non-NIL queries and NIL queries.

We evaluate three features between two queries of each pair. We applied MALLET, a package described in (McCallum, 2002). We use its topic modeling module to predict the topic distribution of and documents. The model is trained on LDC2009E57 corpus, TAC 2009 KBP Evaluation Source Data. We get the whole corpus stemmed with Porter Stemmer in advance. The three features are

- **Title Match**. An integer value of lexical distance between the mentioned name of query and the title of entity.
- **BOW Similarity**. A real value within [0, 1] that computed as the cosine similarity between bag-of-words vectors of the two documents.
- **Topic Similarity**. A real value within [0, 1] that computed as the cosine similarity between topic vectors of the two documents.

On the 2013 Evaluation Queries, we build a graph from queries and relations among them. Each query is a node, and we add an edge between two nodes if the classifier predicts that this pair of queries should be linked to the same entity. Then an isolated clique in this graph indicates that corresponding queries should be linked to the same cluster, a unique NIL cluster or a certain entity. After queries voting equally to decide which cluster to assign, we adjust the system output according to the result. The strict requirement of isolated clique ensures high credibility of the adjustment. This strategy is inspired by micro collaborative ranking proposed in (Chen and Ji, 2011).

3 Evaluation

3.1 Data

The evaluation data in 2013 Evaluation Queries consists of 2190 queries in total, with 1183 non-NIL queries and 1007 NIL queries. Considering the document type, 1134 queries are from newswires, 713 queries are from discussion fora and 343 queries are from web texts. In other words, about a half of the queries come from user-generated content.

3.2 Results

system	All	in KB	NIL
MS_MLI2	0.746	0.722	0.772
SYDNEY_CMCRC1	0.727	0.714	0.738
THUNLP4	0.712	0.721	0.700
UI_CCG5	0.694	0.686	0.700
HITS1	0.684	0.678	0.681
highest	0.746	0.722	0.777

Table 2: Overall B-cubed+ F1 scores of top 5 teams.

The overall results of top 5 teams are shown in Table 2. Our system produces five runs with different combination of some parameters in ranking and clustering. Here we only list THUNLP4 because it is our best run with an overall B-cubed+ F1 score of 0.712. Our system falls behind on NIL queries, but performs well on non-NIL queries, or in-KB queries. The performance of these five teams on queries from different type of documents are shown in Table 3. Apparently, queries from web text and discussion fora are more difficult than those from newswire. The effective candidate generation strategy with alias mining might contributes to the advantage of our system on queries from user-generated content.

4 Cross-lingual Entity Linking

In the Chinese cross-lingual entity linking task, we follow the similar framework and strategies applied

system	NW	WB	DF
MS_MLI2	0.829	0.672	0.648
SYDNEY_CMCRC1	0.796	0.639	0.657
THUNLP4	0.759	0.662	0.662
UI_CCG5	0.770	0.639	0.600
HITS1	0.749	0.616	0.612
highest	0.829	0.678	0.662

Table 3: B-cubed+ F1 scores of top 5 teams on queries from different sources.

in mono-lingual entity linking task. Chinese articles are translated to English with the help of Google Translate to extract features. We mine surface forms in Chinese version of Wikipedia and merge them into English surface form index by interlanguage links.

Data Set	Recall	Average Size
2011 Training	0.903	9.62
2011 Evaluation	0.904	14.9
2012 Evaluation	0.902	17.3

Table 4: The recall and average candidate list size on Chinese cross-lingual data sets.

The recall rates and average candidate list sizes are shown in Table 4. On 2013 Evaluation Queries, the average candidate list size is 20.0. Compared to candidate generation process in mono-lingual task, the main difficulty here is cross-lingual issues rather than noise (i.e. spelling errors).

system	All	in KB	NIL
THUNLP1	0.637	0.658	0.602
THUNLP2	0.639	0.658	0.606
THUNLP3	0.622	0.645	0.587
THUNLP4	0.647	0.645	0.650

Table 5: B-cubed+ F1 scores in Chinese cross-lingual entity linking task.

The evaluation data set consists of 1283 non-NIL queries and 955 NIL queries. The performance is shown in Table 5. The reason that B-cubed+ F1 scores are not as high as mono-lingual task are listed as follows.

1. The recall of candidate generation is too low. A recall of 0.9 means that more than 100 non-NIL queries are wrong from the beginning.
2. The result of machine translation system is not optimized for this kind of task. The errors in translating Chinese entities into English harms the quality of features (i.e. link compatibility) directly.

5 Conclusion

The entity linking system described in this paper has some advantages when dealing with queries from user-generated content. In the circumstance of web blogs or discussion forums, the aliases of entities should be detected to rise recall rate. In addition, because the amount of helpful textual context is limited, more information might be utilized, such as the interest of bloggers and type of discussion forum.

References

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM.

Zheng Chen and Heng Ji. 2011. Collaborative ranking: A case study on entity linking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 771–781. Association for Computational Linguistics.

Silviu Cucerzan. 2011. Tac entity linking by performing full-document entity extraction and disambiguation. In *Proceedings of the Text Analysis Conference*.

John Lehmann, Sean Monahan, Luke Nezda, Arnold Jung, and Ying Shi. 2010. Lcc approaches to knowledge base population at tac 2010. In *Proc. TAC 2010 Workshop*.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit.

Paul McNamee and Hoa Trang Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Text Analysis Conference (TAC)*, volume 17, pages 111–113.

Zhicheng Zheng, Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010. Learning to link entities with knowledge base. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 483–491. Association for Computational Linguistics.