ITNLP Entity Linking System at TAC 2013

Yaming Sun, Xianqi Zou, Lei Lin, Chengjie Sun
Harbin Institute of Technology
{ymsun, xqzou, linl, cjsun}@insun.hit.edu.cn

Abstract

This paper describes the ITNLP system participated in the Knowledge Base Population (KBP) track English Entity Linking task. Our Entity linking system is composed of three parts: candidate generation, candidate ranking and nil clustering. In the candidate generation process, the redirect pages and anchor texts in Wikipedia are utilized to generate candidate entities for the mentions. Ranking SVM is adopted to rank the candidates by a set of linguistic features. In the end, the hierarchical clustering algorithm is used to cluster those queries which return NIL in the ranking process.

1 Introduction

The Knowledge Base Population (KBP) track at TAC 2013 aims to develop and evaluate technologies for building and populating knowledge bases (KBs) about named entities from unstructured text. The KBP systems are required to either populate an existing reference KB or build a KB from scratch. KBP 2013 contains several tracks, including entity linking, english slot filling, cold start KBP etc. We participate in the entity linking track.

In the entity linking task, entity mentions must be aligned with entities in the reference KB or new entities discovered in the document collection. There are two different entity linking tasks, respectively are monolingual entity linking and cross-lingual entity linking. We merely take part in the monolingual entity linking task. Specifically, given a query which consists of a name string, a background document ID, and the location of the name string in the document, the system is required to provide the corresponding KB entrys ID or NILxxxx ID if there is no such KB entry about this name string.

The main challenges about entity linking are as follows: first, entities in the documents often have variable expressions, such as abbreviations, aliases, misspellings etc.; second, the ambiguity problem, namely one name string may refer to several different entities; third, the knowledge base is incomplete, so that we should cluster all the name strings which have the same meaning.

In the previous research about entity linking, most systems split the entity linking task into three steps. First, in order to reduce the computational cost, generate the most probable candidate entities for each mention in the query; then, adopt some strategies to select the best candidate entity as the entity linking result or return NIL if there is no corresponding entity; at last, cluster all the mentions which have no corresponding entity entry in the KB. We follow the prior schema and design methods for each component. Our entity linking system consists of three components: candidate generation, candidate disambiguation and mention clustering.

2 Our Approach

The framework of entity linking is as Figure 1 shows. First, for a given mention, generate the candidate list. To deal with the mentions which do not have corresponding entities in the KB, we generate a virtual candidate NIL for each mention. Then, we utilize SVM-rank [1] (Joachims, 2006) to rank all the

[1] http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

candidates, and treat the top-1 candidate as the result. Finally, cluster those mentions whose top-1 candidate ranking result is NIL.

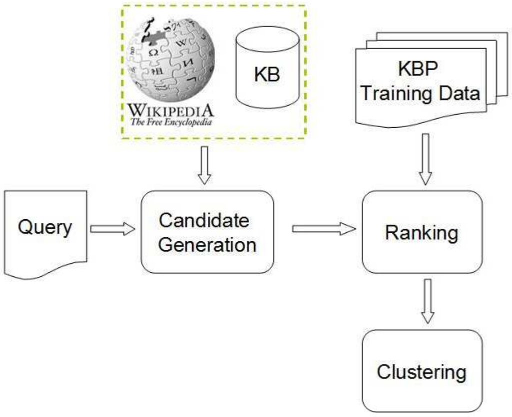

Figure 1: Flow chart of our method

2.1 Candidate Generation

A knowledge base always contains huge number of entities. It is infeasible to consider the whole knowledge base to link a mention. Therefore, the candidate generation is an essential step. As candidate generation is the first step of the entity linking task, its recall will influence the final performance. To achieve high recall in the candidate generation step, we utilize multiple resources including the Wikipedia and the KBP knowledge base.

Wikipedia contains rich information about entities. To deal with the variants of entities, we construct a dictionary using the redirect pages and the anchors of Wikipedia. To handle the ambiguity, we make use of the disambiguation pages. We download a Wikipedia dump and extract information using the toolbox JWPL[2] (Zesch et al., 2008).

Owing to the KBP knowledge base can not be covered by the Wikipedia dump, we also adopt simple matching strategies such as fuzzy matching and abbreviations matching to find candidates from the knowledge base. Finally, we combine the results and retain those covered by the KBP knowledge base.

2.2 Candidate Ranking

We treat the candidate disambiguation as a ranking problem and utilize the SVM-rank to rank the candidates. To deal with the queries which return NIL, we add a virtual candidate NIL for each mention. If the NIL is ranked to the top1, then return NIL for the query, otherwise return the top1 candidate entity as the linking result.

We adopt a set of features which are commonly utilized by previous systems such as (McNamee et al., 2009) for mentions, entities and the virtual NIL. The feature description are as follows.

- Abbreviation: Whether the letters of query are the abbreviation of Entitys name. For example, WTO matches the first letter of World Tourism Organization.
- C_Query: Whether the name of the query appears in the entitys wiki text.
- C_Entity: Whether the name of the entity appears in the querys context, namely the querys document.
- T_Entity: The entitys type in the Wikipedia knowledge base.
- N_Share: Whether the querys name and the entitys name share a common name. For example, company, bank etc.
- PartMatch: Whether the name of query matches part of the entitys name or the name of entity overlaps part of querys name.
- E_Distance: Edit distance related similarity between the querys name and the name of entity.
- C_TFIDF: The cosine similarity of the querys context and the entitys article, which is represented as TF/IDF.

2.3 NIL Clustering

For those queries which return NIL in the ranking process, we utilize the hierarchical clustering to cluster them. We assume that the queries which are semantic related would have high cosine similarity among their contexts in the document, thus we calculate the TF/IDF of the contexts to get the cosine similarity. We use the HAC algorithm which is a “bottom up” approach to cluster data. Setting the number of clusters too large will make many clusters to get only one document, while too small will

[2] http://code.google.com/p/jwpl/

induce some clusters to have too many documents. Thus, we set the number of clusters to be 140 empirically.

3 Experiment

We use the evaluation data of KBP 2011 for training the SVM-rank model. The training set contains 2250 queries, 1126 of which do not have corresponding entity entries in the knowledge base, namely should return nil. Our system uses both the wiki text and the provided offsets in queries, no web information is employed.

The candidate generation, feature selection and nil clustering processes are as the Approach section describes. Specifically, the number of clusters is set 140 empirically. We submit only one result to KBP 2013. The official evaluation result of our system is as Table 1 shows. The official score is B$\hat{3}$+ F1 over all evaluation queries.

System	All	PER	ORG	GPE
Our	**0.503**	**0.532**	**0.538**	**0.446**
Median	0.583	0.617	0.593	0.529
Highest	0.721	0.758	0.737	0.731

Table 1: Entity Linking submission scores

4 Conclusion

In this paper, we describe our participation in the entity linking task of KBP 2013. We treat the entity task as a ranking problem, and split it into three subtasks: candidate generation, candidate ranking and nil clustering. Our result is lower than the median. The primary cause may contain two aspects, first the training set is not large enough, second, the number of cluster has a heavy influence on the final result.

References

Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226. ACM.

Paul McNamee, Mark Dredze, Adam Gerber, Nikesh Garera, Tim Finin, James Mayfield, Christine Piatko, Delip Rao, David Yarowsky, and Markus Dreyer. 2009. Hltcoe approaches to knowledge base population at tac 2009. In *Text Analysis Conference (TAC)*.

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *LREC*, volume 8, pages 1646–1652.