A Graph-based Method for Entity Linking

Yuhang Guo, Wanxiang Che, Ting Liu[*], Sheng Li
Research Center for Social Computing and Information Retrieval
MOE-Microsoft Key Laboratory of Natural Language Processing and Speech
School of Computer Science and Technology
Harbin Institute of Technology, China
{yhguo, car, tliu[*], sli}@ir.hit.edu.cn

Abstract

In this paper, we formalize the task of finding a knowledge base entry that a given named entity mention refers to, namely entity linking, by identifying the most "important" node among the graph nodes representing the candidate entries. With the aim of ranking these entities by their "importance", we introduce three degree-based measures of graph connectivity. Experimental results on the TAC-KBP benchmark data sets show that our graph-based method performs comparably with the state-of-the-art methods. We also show that using the name phrase feature outperforms the commonly used bag-of-word feature for entity linking.

1 Introduction

Entity linking is the task of computationally mapping a named entity mention in given context to the intended entry in a referential knowledge base. Large-scale knowledge bases have been proved to be valuable for many natural language processing applications such as question answering (MacKinnon and Vechtomova, 2008), information extraction (Paşca, 2009), information retrieval (Santamaría et al., 2010), coreference resolution (Ponzetto and Strube, 2007) and word sense disambiguation (Fogarolli, 2009). And entity linking is a natural way to access these knowledge bases.

Mention ambiguity is prevalent in language use. For example, in the following two sentences

- "Mount Bromo is one of Java's most popular tourist attractions."
- "Candidates must have technical skills in JSP, ASP, Java, HTML."

the named entity mention *Java* refers to *an island in Indonesia* and *a programming language* respectively.

In this paper, we focus on the named entity disambiguation in entity linking and approach this problem from a graphical perspective. We begin by building a graph in which nodes correspond to the context around the target mention and the candidate entries from the knowledge base, whereas directed edges represent the reference dependency between nodes. Then we calculate the "importance" score for each candidate node and assign the most "important" candidate to the target mention. Here we compare three degree-based measures of graph connectivity that assess the node importance. Through experiments performed on benchmark data sets, we show that the graph-based method achieves comparable performance to the state-of-the-art in the entity linking task. The result also indicates that to build linkage between nodes, name phrase (i.e. n-gram of name words) performs better than the traditional bag-of-word feature. Our contributions are threefold: introducing an entity linking method under the graph-based framework, a novel in-degree graph connectivity measure for entity disambiguation, and an empirical comparison of the bag-of-word and the name phrase feature.

This paper is organized as follows. Section 2 gives a brief overview of the related work. Section 3 introduces the Wikipedia encyclopedia and our graph-based method. Section 4 describes each components of our entity linking system, especially the disambiguation algorithm. Experimental settings, results and analysis are presented in

[*] Corresponding author

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 1010–1018,
Chiang Mai, Thailand, November 8 – 13, 2011. ©2011 AFNLP

Section 5. The last section offers some concluding remarks.

2 Related Work

Linking name mentions to knowledge base entries has attracted more and more attentions in these years. As an open available resource, Wikipedia is a natural choice of knowledge source for its large scale and good quality. Early work mainly focused on the usage of the structure information in Wikipedia. Bunescu and Pasca (2006) trained a taxonomy kernel on Wikipedia data to disambiguate named entities in open domain. Cucerzan (2007) integrated Wikipedia's category information in their vector space model for named entity disambiguation. Mihalcea and Csomai (2007) extracted sentences from Wikipedia, regarding the linking information as sense annotation, and used supervised machine learning models to train a classifier for disambiguation. Similarly, Milne and Witten (2008) adopted a learning approach for the disambiguation in Wikipedia. Their method is to balance the prior probability of a sense with its relatedness to the surrounding context.

Recently, an Entity Linking task in the Knowledge Base Population (KBP) track evaluation (McNamee and Dang, 2009) provided a benchmark data set. The first KBP track was held at the Text Analysis Conference (TAC)[1], aiming to explore information about entities for Question Answering and Information Extraction. The knowledge base in the evaluation data is also based on Wikipedia. Many information retrieval based models have been proposed on this data set. For example, Dredze et al. (2010) presented a maximum margin approach to rank the candidates. They combined rich features including Wikipedia structure and entity's popularity. Zheng et al. (2010) proposed learning to rank models for the entity linking problem and obtained high accuracy.

One of the most important component of entity linking is to compute the relatedness between entities. Some of the previous works use vector space model and calculate the cosine similarity over the bag-of-word feature vectors (Mihalcea and Csomai, 2007) or the category feature vectors (Cucerzan, 2007). Others take into account citation overlap of the relevant Wikipedia entry (Milne and Witten, 2008; Kulkarni et al., 2009; Radford et al., 2010), which implies the co-occurrence of the entities. These methods work when significant overlap can be observed between the entities or their features. For example, the co-occurrent frequency of *Java (programming language)* and *HTML* is higher than *Java (island)* and *HTML* in the Wikipedia articles. Hence the *Java* probably means the programming language rather than the island when its context contains *HTML*. However, entities like *human* and *homo* are seldom cited in the same article. Although they are highly related. In fact, their relatedness can be easily captured through their mutual citations. In this paper, we compute the entity relatedness by using the direct citation in the Wikipedia.

Graph-based approaches are proved useful in the research of word sense disambiguation. Sinha and Mihalcea (2007) compared several measures of word semantic similarity and algorithms for graph centrality for word sense disambiguation. They found that the performances of their graph-based algorithms are competitive to the unsupervised state-of-the-art ones. Navigli and Lapata (2009) investigated several graph connectivity measures for word sense disambiguation. They found the best measures are degree and PageRank (Brin and Page, 1998). In this paper, we approach entity linking by leveraging graph-based methods.

3 Graph-based Entity Linking

As defined in the TAC-KBP track, the input of the entity linking task includes:

- a Knowledge Base *KB* $\subseteq \mathcal{E}$, where *KB* $= \{e_i | 1 \leq i \leq n\}$, e_i is the ith entity in *KB* and $\mathcal{E}$ is the set of all entities around the world, and

- a query that consists of a mention string $m \in \mathcal{L}$ and the background document $D \in \mathbb{D}$ it appears in, where $\mathcal{L}$ is a lexicon which is composed of words and phrases, and $\mathbb{D}$ is a collection of documents.

The output is

- the entity e_i that m refers to in the context of D, where $e_i \in$ *KB*, and

- `NIL` if such an entity is absent from the *KB*.

We formalize the task as a function:

$$\textsc{Link}(m, D) = \begin{cases} e_i & \text{if } 1 \leq i \leq n \\ \texttt{NIL} & \text{otherwise} \end{cases}$$

[1] http://www.nist.gov/tac

where $e_i = \textsc{Entity}(m, D)$ and

$$\textsc{Entity} : \mathcal{L} \times \mathbb{D} \rightarrow \mathcal{E}$$

is the function to find the corresponding entity for a query.

In our experiments we use Wikipedia as the knowledge base. In the following, we first briefly introduce the structure of Wikipedia. Next we describe our entity linking method. Note that although we use the Wikipedia in the experiments, our method is not limited to this knowledge source.

3.1 Wikipedia

Wikipedia is an online encyclopedia written by volunteers around the world. Its English version contains more than 3,400,000 articles [2]. Each article in the Wikipedia consists of a unique title and a main body which includes descriptions of the concerned entity. Articles are usually titled by the formal name of the entities, which sometimes are suffixed with a discriminative string on condition that another entity also share the same name. In the latter situation, the namesakes will be listed in a **Disambiguation Page**[3]. As an example, consider two entities of the same name *Java*, "the most populous island in Indonesia,", and "an object-oriented high-level programming language." In the page of *Java (disambiguation)*, the corresponding titles are represented as:

1. *Java (island)*,
2. *Java (programming language)*.

The main body of an article consists of descriptive words for the entity. In this text, many related entities are mentioned and some of the entities' titles are further wrapped with brackets to link to the corresponding articles with the aim of facilitating the access to those articles. For instance, in the following fragment of the article *Java (programming language)*,

> "Sun relicensed most of its Java technologies under the [[GNU General Public License]].[4]"

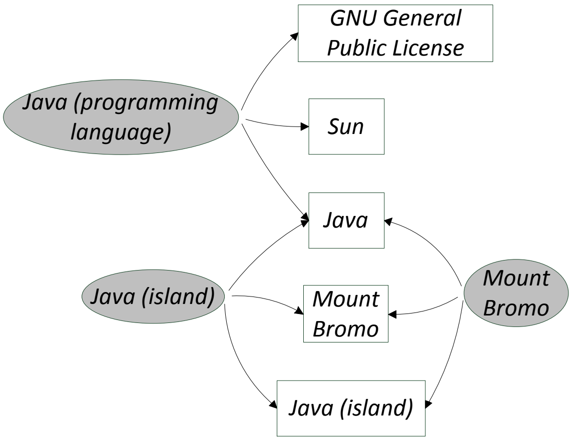


Figure 1: An example of the Wikipedia graph.

The entities: *Sun*, *Java*, and *GNU General Public License* are mentioned, where the square brackets "[[]]" will generate a cross reference link to the article of *GNU General Public License*.

We can view the Wikipedia as a graph with two types of nodes: the article nodes and the name nodes. A directed edge from an article node to a name node represents that the name appears in the article.

Figure 1 shows a partition of the Wikipedia graph. In this graph, the dark gray ellipse nodes correspond to articles which we tag with their titles and the white square nodes correspond to name strings. *Sun*, *Java*, and *GNU General Public License* are mentioned in the article of *Java (programming language)*, and hence we draw directed edges from the article to them.

3.2 The Disambiguation Method

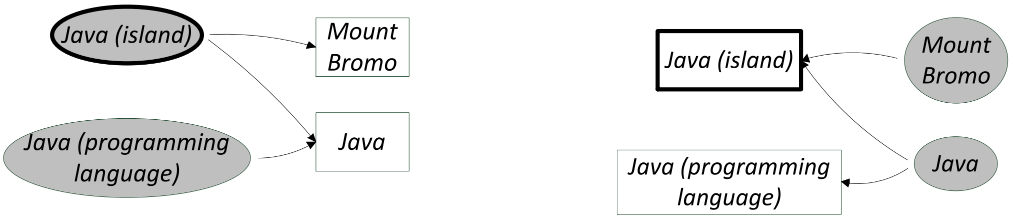
In this paper, we approach to the entity linking task in two stages. The first stage is to find the candidate entities to the target name string. And the second stage is to estimate the "importance" of each candidate according to the context of the mention and select the most "important" one. Here we focus on the second stage. The steps of our candidate extraction will be described in section 4. In this section, we will introduce a disambiguation method based on out-degree and in-degree measures of graph connectivity .

We build a graph $G = (V, E)$ corresponding to the context where the target name appears in. For the out-degree connectivity measure, the node set in the graph consists of the names that are mentioned in the context and the articles of the corre-

[2] Throughout our experiment, we will use the English version of Wikipedia snapshotted in January, 2010.

[3] See: http://en.wikipedia.org/wiki/Wikipedia:Disambiguation for detail

[4] In our experiment the main body text we use is the source of the article, which is encoded in the wiki markup language. See: http://en.wikipedia.org/wiki/Wiki_markup



(a)

(b)

Figure 2: An example of the graph-based named entity disambiguation method.

sponding candidates (i.e. context: name nodes and candidate: article nodes). There exists a directed edge from an article node to a name node when the name is mentioned in the article. The article node of the highest out-degree is considered as the most "important" one in this graph and the corresponding entity to this article node is then selected for the queried mention. For the in-degree measure, the node set consists of the names of the candidate entities and the articles of the context entities that mentioned in the context (i.e. context: article nodes and candidate: name nodes). There is an edge that linked to a candidate name node when a context article contains that name. This time name node with the highest in-degree is considered most "important" and we assign the corresponding entity of this name node to the queried mention.

Here we give a simplified example. Consider of a context fragment:

> "Mount Bromo is one of Java's most popular tourist attractions."

and candidates of a query name *Java*:

1. *Java (island)*,
2. *Java (programming language)*.

The graphs for the out-degree and the in-degree measures are illustrated in Figure 2. In Figure 2(a) we can see the article nodes are *Java (island)* and *Java (programming language)*, and the name nodes are *Mount Bromo*[5] and *Java*. The node of *Java (island)* has 2 outer links which is higher than any other nodes and therefore *Java (island)* is assigned to the query *Java*. In Figure 2(b) *Java (island)* will also be selected because its in-degree is the highest.

Formally, the above method can be represented as to find the node:

$$u^* = \arg\max_u imp(u). \tag{1}$$

For the out-degree measure,

$$imp(u) = deg_{out}(u) = |(u, v) \in E : v \in V|.$$

And for the in-degree measure,

$$imp(u) = deg_{in}(u) = |(v, u) \in E : v \in V|.$$

We can combine the out-degree and in-degree measures and get:

$$imp(u) = (1-\lambda)ndeg_{out}(u) + \lambda ndeg_{in}(u), \tag{2}$$

where

$$ndeg_{out}(u) = \frac{deg_{out}(u)}{\sum_{u \in V} deg_{out}(u)}$$

and

$$ndeg_{in}(u) = \frac{deg_{in}(u)}{\sum_{u \in V} deg_{in}(u)}$$

are normalized degree scores. In our experiment, when there are two tied candidates, we choose a random one.

4 An Entity Linking System

In this section, we will introduce an entity linking system. This system includes 2 components: candidate selection and disambiguation, in which the disambiguation part is based on the graphical method we described in section 3.2.

4.1 Candidate Selection

As described in (Dredze et al., 2010), usually an entity has three kinds of name variations, including acronyms (e.g. *American Broadcasting Company* vs. *ABC*), aliases (e.g. *Robert Gates* vs. *Bob Gates*), and alternate spellings (e.g. *Air Macau* vs. *Air Macao*) etc.

1) For an acronym, we try to find its full form in the context through the following rules:

[5] Anchor text *Bromo* appears in the Wikipedia page of *Java (island)*. In the source of the article, we can find the target *Mount Bromo*

- If the acronym is bracketed, we extract the name phrase immediately before the capitalized letter nearby (e.g. "*... The Mexican Football Federation (FMF) on Monday ...*").

- If the acronym is followed by a bracket, we extract the phrase in the bracket (e.g. "*... From the PRC (People's Republic of China) we get much benefit. ...*").

- Or else, we just find the phrase in the context with the same capitalized letter as the acronym (e.g. *"... he told the Australian Broadcasting Corporation. ..."* vs. *ABC*).

When the full form of the acronym is found, we substitute the target mention string with its full form.

2) The Wikipedia provides the most common alias names for entities through the **Redirect Pages**[6], which maps an alias to the corresponding article titled with the formal name. By this mechanism we can access the candidate with the formal name from an alias name (e.g. *Bob Gates* → *Robert Gates*), or find several candidates listed in a disambiguation page (e.g. *Gates* → *Gates (disambiguation)*).

3) However, name variations which are not included in the Wikipedia's redirect pages (e.g. *Air Macao*) could not be found by the above function. We invoke web search engines to find the most relevant term of the name string in the Wikipedia using the "within a site" search function. We construct and submit a search query like "*Air Macao site:en.wikipedia.org*" and extract the first returned entity (i.e. *Air Macau*) as a candidate.

4.2 Disambiguation

To build a graph for the disambiguation, we need to extract names from the context of the query (either as the name node or the article node). We use a segmentation technique which is inspired from a Chinese word segmentation algorithm, the forward maximum matching algorithm (Guo, 1997) on the context to find all the names which are included in the Wikipedia title list (i.e. all the name phrases in our Wikipedia graph are the Wikipedia article titles). This algorithm prefers to find the longest names that match with the string. Here we refer the context name as neighboring name and the corresponding entity as neighboring entity of the target name string.

For the out-degree measure (as described in Section 3.2), we search for each neighboring name in the article of each candidate. If there is a match, we draw a directed edge from the candidate node to the neighboring name node. This procedure can be represented as Algorithm 1, where C_a is the article node set of the candidate entities, N_n is the node set of the neighboring names, and $Article(a)$ is the main body text of an article node a.

Algorithm 1 Out-degree measure based graph construction

```
Require: $C_a$ and $N_n$
Ensure: Graph $G = (V, E)$
$V := C_a \cup N_n$
$E := \emptyset$
for all $c \in C_a$ do
    for all $n \in N_n$ do
        if $n \in Article(c)$ then
            $E := E \cup (c, n)$
        end if
    end for
end for
return $(V, E)$
```

Similarly, for the in-degree measure we build the graph in Algorithm 2, where C_n is the name node set of the candidate entities and N_a is the article node set of the neighboring entities.

Algorithm 2 In-degree measure based graph construction

```
Require: $C_n$ and $N_a$
Ensure: Graph $G = (V, E)$
$V := C_n \cup N_a$
$E := \emptyset$
for all $c \in C_n$ do
    for all $n \in N_a$ do
        if $c \in Article(n)$ then
            $E := E \cup (n, c)$
        end if
    end for
end for
return $(V, E)$
```

For the combined measure we build both of the above graphs. And then we normalize the measures and combine them with a λ parameter (see

[6] See http://en.wikipedia.org/wiki/Wikipedia:Redirect for detailed instructions

Equation 2) for each candidate node.

When the graph is constructed, we then select the candidate node with the maximum out-degree or in-degree or the combined degree based measure. In our method, if the maximum out-degree or in-degree of the candidate nodes is zero, which means for all the candidate nodes there is no edges out or in, then the system will return NIL to assert the corresponding entity is not included in the knowledge base.

5 Experiment

5.1 Data set

We evaluated our disambiguation method on two benchmark data sets. Specifically, we use the entity linking data from TAC-KBP track in 2009 (McNamee and Dang, 2009) and the same track in 2010 (Ji et al., 2010).

The TAC-KBP 2009 data set includes 3,904 queries for 560 distinct entities and a track knowledge base (TKB) which contains 818,741 entities. The knowledge base were derived from a snapshot of English Wikipedia in October, 2008. Each query is comprised of a target name mention and a context document where the name occurs. These documents are mainly newswire documents. Over a half (2229) of the queries could not be linked to any entity in the TKB and should be tagged with NIL.

The TAC-KBP track in 2010 inherit the knowledge base used in the TAC-KBP 2009 and its test data set contains 2,250 queries. Similar to the track in 2009, Over a half (1230) of the entities are absent from the knowledge base. In this data set, a third (750) of the context documents are from weblog texts and the rest are from newswire documents.

In our system, we use Wikipedia as the knowledge base (KB). The result of our system can be easily mapped to the TKB entries because the KB is a superset of the TKB. In the entity linking, if the selected entity in the Wikipedia KB is not included in the TKB, our system will return NIL. We used the snapshot of English Wikipedia in January, 2010 and employed a Java based application programming interface (Zesch et al., 2008) to access this archive. The Wikipedia dump is open available in the web site: http://dumps.wikimedia.org/enwiki/.

TAC-KBP track	2009	2010
candidates/query	6.36	4.55
coverage	0.8083	0.7862

Table 1: Data sets and the result of the candidate selection.

# sentence	1	3	5	7	9	all
# neighbor	6	10	14	16	18	36

Table 2: The average number of the neighboring names for each query with different context window sizes in the TAC-KBP 2009 data set.

5.2 Candidate Coverage

As a result of the candidate selection (see Section 4.1), we obtained 6.36 candidates for each query on average from TAC-KBP track 2009 and 4.55 from TAC-KBP track 2010. In order to isolate the impact of the disambiguation method, we evaluated the coverage of the candidate set, which is the percentage of the intended queries that fall into the candidate set. Formally,

$$coverage = \frac{\sum_{q \in Q} |\{e_q \in C_q \cap \mathit{TKB}\}|}{\sum_{q \in Q} |\{e_q \in \mathit{TKB}\}|},$$

where Q is the set of the queries, e_q is the corresponding entity for the query q, C_q is the candidate entity set of q, and *TKB* is the track knowledge base, which is a set of entities here. In Table 1, we show the result of the candidate selection for the two data sets.

5.3 Entity Linking

We segment the context document into word or name phrase fragments and filter out stop words (e.g. *about*, *have*, *the*, etc.). In order to evaluate our graph-based method in different scales, we select nodes of neighboring entities from these fragments in several context window sizes around the target mention name: the sentence where the target name appears in, plus the immediately adjacent sentence before and after the sentence containing the target name, and plus the adjacent two sentences before and after, etc. From Table 2 we can see that in the data set of TAC-KBP 2009, the average number of the neighbor nodes per query we extracted increases as the context range increases.

Figure 3 shows the micro-averaged accuracies of our graph-based method on the TAC-KBP 2009

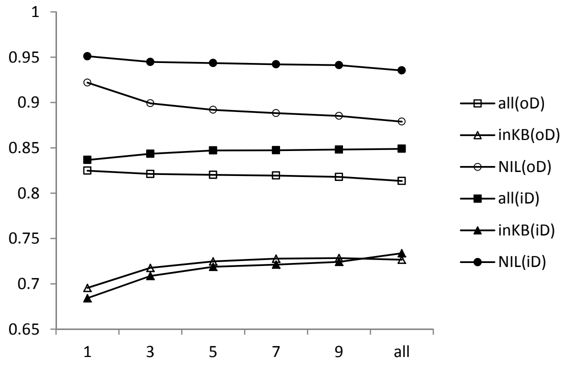


Figure 3: Accuracies for the out-degree based algorithm (oD) and the in-degree based algorithm (iD) on TAC-KBP 2009 data with different number of sentences around the target name mention as context.

data set. Our evaluation metric includes the accuracy of all queries (all), the accuracy of the queries that are in KB (inKB), and the accuracy of identifying the out-of-KB entities (NIL). The horizontal axis is the number of the sentences around the target mention name (i.e. context window size), where "all" means that all the sentences in the document are included. From this figure, we can see that the inKB accuracies of the out-degree measure and the in-degree measure increase and portray a similar trend as more neighbor nodes imported. On the contrary, the NIL accuracies decrease and the overall accuracies have no obvious changes. The accuracies of the two measures for the inKB queries are very close, but for the NIL queries the in-degree measure outperforms the out-degree significantly (z test with p=0.01). This results in that for all queries the accuracies of the in-degree measure (i.e. all(iD)) are higher than the out-degree measure (i.e. all(oD)) in all the context ranges. We find that among the candidate nodes for each query, more than 2 nodes have non-zero out-degree on average, whereas less than 0.5 node has non-zero in-degree, which means that the in-degree measure returns more NIL entities, resulting in higher precision on NIL queries in this data set.

We combine the out-degree measure and the in-degree measure through Equation 2. The system performance with the λ parameter is illustrated in Figure 4. Here we set the context window size as 5. Note that when $\lambda = 0$ or $\lambda = 1$, the method reduces to the pure out-degree measure or in-degree based measure. In Figure 4 we can see that for the non-trivial combination (i.e. $\lambda \neq 0$ and 1) the accuracies have no obvious changes with the λ parameter. The NIL accuracies of the combined method are nearly the same as the out-degree ones and significantly lower than the in-degree measure. For the inKB queries, the combined method performs better than the other two methods. For all queries, the accuracy of the combined method is lower than the in-degree but higher than the out-degree ones. The reason for the higher accuracy of the in-degree measure than the combined measure is that in the combined measure the candidates of zero score are fewer than that in the in-degree measure even if λ is near to 1. So that the accuracy for NIL queries of the combined measure is lower than the in-degree measure.

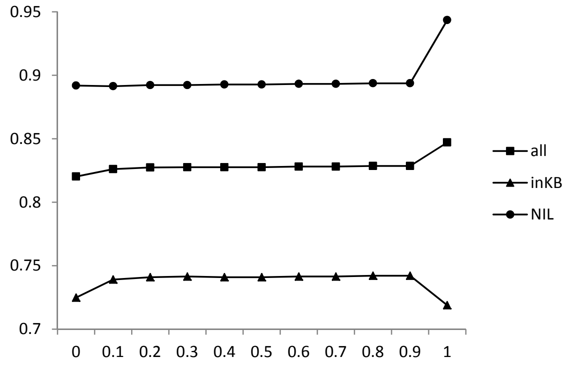


Figure 4: Accuracies for the combined measure with the λ parameter.

In Table 3 we show the results of our graph based method on TAC-KBP 2009 and TAC-KBP 2010 data set. A number of the state-of-the-art system results are compared. According to our experiment on TAC-KBP 2009 data set, here we set the context window size as 1 for the out-degree measure (oD), as "all" for the in-degree measure (iD) and as 5 for the linear combined measure (Comb.), and set $\lambda = 0.5$.

We list the results of the top 3 systems in the TAC-KBP track 2009 and 2010. Most of them used sophisticated feature or labeled data for training. On the contrary, our graph-based methods need no feature other than the named phrases. Besides, our system has few parameters to tune. Among these system results, our graph-based method with in-degree measure outperform-

	TAC-KBP 2009			TAC-KBP 2010		
Acc.	all	inKB	NIL	all	inKB	NIL
Rank 1	0.8217	0.7654	0.8641	0.8680	0.8059	0.9195
Rank 2	0.8033	0.7725	0.8264	0.8373	0.7520	0.9081
Rank 3	0.7984	0.7063	0.8677	0.8191	0.7373	0.8870
sLesk	0.8066	0.7075	0.8811	0.7938	0.7059	0.8667
oD	0.8248	0.6955	0.9219	0.8169	0.7059	0.9089
iD	0.8489	0.7337	0.9354	0.8240	0.7127	0.9163
Comb.	0.8276	0.7409	0.8928	0.8160	0.7402	0.8789

Table 3: System accuracies on TAC-KBP 2009 and 2010 data sets. Rank 1-3 are top 3 systems in the TAC-KBP track 2009 and 2010, sLesk is the simplified Lesk algorithm based system, oD and iD are the out-degree based and the in-degree based systems and Comb. is the system that combined the out-degree and in-degree measure.

s the best system in TAC-KBP 2009 significantly (z test, p=0.01) and can outperform the third rank system in TAC-KBP 2010.

Simplified Lesk algorithm (sLesk) (Lesk, 1986; Banerjee and Pedersen, 2002; Agirre and Edmonds, 2006) is a well-known disambiguation algorithm which is similar to our graph-based method with in-degree measure. This algorithm is usually used as the baseline for word sense disambiguation. The main idea of this algorithm is to find the sense, the glossary of which has the most overlap with the context of the target multi-meaning word. The difference between these two algorithms is that our method uses name phrases as the feature other than the bag-of-word feature used in sLesk. Here we set the context window size of sLesk the same as the out-degree measure. The result shows that on both data sets our method with out-degree measure outperforms the simplified Lesk algorithm by a significant margin (z test, p=0.05).

From the last three rows in Table 3 we can see that in our graph based methods, the in-degree measure performs best among the three measures for all queries. The combined measure has a higher accuracy in inKB queries. The high NIL accuracy of the in-degree measure makes it to be suitable for the task of identifying novel concepts such as knowledge base population.

6 Conclusion

In this paper, we presented a preliminary study of graph based method for entity linking. We evaluated three degree-based measures to find the most suitable entity node for the target name mention. Our experimental results on two benchmark data sets show that our simple but effective method performs comparably to the sophisticated state-of-the-art methods and the in-degree measure outperforms the other two measures.

Based on the comparison between the simplified Lesk algorithm and our out-degree based method, we also conclude that the name phrase feature is better than the common used bag-of-words.

Acknowledgments

This work was supported by National Natural Science Foundation of China (NSFC) via grant 60803093, 60975055, 61073126, 61133012 Natural Scientific Research Innovation Foundation in Harbin Institute of Technology (HIT.NSRIF.2009069), and Fundamental Research Funds for the Central Universities (HIT.KLOF.2010064).

References

Eneko Agirre and Philip Edmonds, editors. 2006. *Word Sense Disambiguation: Algorithms and Applications*, volume 33 of *Text, Speech and Language Technology*. Springer, July.

Satanjeev Banerjee and Ted Pedersen. 2002. An adapted lesk algorithm for word sense disambiguation using wordnet. In *CICLing '02: Proceedings of the Third International Conference on Computational Linguistics and Intelligent Text Processing*, pages 136–145, London, UK. Springer-Verlag.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands. Elsevier Science Publishers B. V.

Razvan C. Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL*. The Association for Computer Linguistics.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 708–716, Prague, Czech Republic, June. Association for Computational Linguistics.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 277–285, Beijing, China, August. Coling 2010 Organizing Committee.

Angela Fogarolli. 2009. Word sense disambiguation based on wikipedia link structure. *International Conference on Semantic Computing*, 0:77–82.

Jin Guo. 1997. Critical tokenization and its properties. *Comput. Linguist.*, 23:569–596, December.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the tac 2010 knowledge base population track. In *Proceedings of the Third Text Analysis Conference (TAC2010)*.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 457–466, New York, NY, USA. ACM.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, SIGDOC '86, pages 24–26, New York, NY, USA. ACM.

Ian MacKinnon and Olga Vechtomova. 2008. Improving complex interactive question answering with wikipedia anchor text. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*, ECIR'08, pages 438–445, Berlin, Heidelberg. Springer-Verlag.

P. McNamee and H.T. Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Proceedings of the Second Text Analysis Conference (TAC2009)*.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242, New York, NY, USA. ACM.

David Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518, New York, NY, USA. ACM.

Roberto Navigli and Mirella Lapata. 2009. An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(1):1–1.

Marius Paşca. 2009. Outclassing Wikipedia in open-domain information extraction: Weakly-supervised acquisition of attributes over conceptual hierarchies. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 639–647, Athens, Greece, March. Association for Computational Linguistics.

Simone Paolo Ponzetto and Michael Strube. 2007. Knowledge derived from wikipedia for computing semantic relatedness. *J. Artif. Int. Res.*, 30:181–212, October.

Will Radford, Ben Hachey, Joel Northman, Matthew Honnibal, and James R. Curran. 2010. Document-level entity linking: Cmcrc at tac 2010. In *Proceedings of the Text Analysis Conference*, Gaithersburg, MD, USA.

Celina Santamaría, Julio Gonzalo, and Javier Artiles. 2010. Wikipedia as sense inventory to improve diversity in web search results. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1357–1366, Uppsala, Sweden, July. Association for Computational Linguistics.

Ravi Sinha and Rada Mihalcea. 2007. Unsupervised graph-based word sense disambiguation using measures of word semantic similarity. In *International Conference on Semantic Computing*, volume 0, pages 363–369, Los Alamitos, CA, USA. IEEE Computer Society. unread.

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *LREC*. European Language Resources Association.

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, Los Angeles, California, June. Association for Computational Linguistics.