Entity Linking from Microblogs to Knowledge Base Using ListNet Algorithm

Yan Wang*, Cheng Luo, Xin Li, Yiqun Liu, Min Zhang, and Shaoping Ma

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,c-luo12,xli12}@mails.thu.edu.cn,
{yiqunliu,z-m,msp}@mail.thu.edu.cn

Abstract. Entity Linking (EL) is a fundamental technology in Natural Language Processing and Knowledge Engineering. Previous works mainly focus on linking mentioned names recognized in news or articles to knowledge base. However, in social network, user-generated content is quite different from typical news text. Users sometimes use words more informally, even create new words. One entity may have different aliases mentioned by web users, so identifying these aliases calls for more attention than before. Several methods are proposed to mine aliases and a learning-to-rank framework is applied to combine different types of feature together. A binary classifier based on SVM is trained to judge whether the top one candidate given by ranking algorithm is accepted. The evaluation results of NLP&CC 2013[1] Entity Linking Track shows the effectiveness of this framework.

Keywords: Entity Linking, Microblog, Learning to Rank.

1 Introduction

Recent years have witnessed a big bloom in social network system (SNS), such as Facebook, Twitter, and Microblog in China. According to Sina.com, the number of active users on Sina Weibo is about 46.2 million in December, 2012[2]. Users create and share microblogs in large scale. A microblog is a short paragraph with limited number of characters, or even just a sentence. It is similar with a web page to the extent of carrying and transmitting message. The difference between a microblog and a news article is that a microblog is much more easier to produce and spread via the social network. Every user on SNS could be both information receiver and provider. Besides the spreading efficiency, some contents or opinions that are not allowed for public media could appear in users' personal pages, because individuals publish microblogs on behalf

* This work was supported by Natural Science Foundation (60903107, 61073071) and National High Technology Research and Development (863) Program (2011AA01A205) of China.

[1] NLP&CC (CCF Conference on Natural Language Processing and Chinese Computing) http://tcci.ccf.org.cn/conference/2013/

[2] http://news.xinhuanet.com/newmedia/2013-02/21/c_124369896.htm

G. Zhou et al. (Eds.): NLPCC 2013, CCIS 400, pp. 277–287, 2013.

of themselves. In this way, microblogs contribute to the variety of information on the Internet.

It is valuable to understand what microblogs are talking about. The topic trends often indicate social events, such as natural disasters, scandals and other hot spots. [1] detects earthquake through analyzing Twitter contents. [2] identifies emerging topics on Twitter by monitoring keywords whose frequencies of appearance suddenly rise up, many of which are named entities such as names of people, organizations or geographical locations. In addition, microblogs are helpful in personalized applications because the contents of microblogs shared and commented by a user suggest the user's preferences and interests. Such information is worth of mining to improve the personalized advertising and content recommendation.

Discovering what the named entities stand for is important for understanding microblogs. For example, a microblog reviews the performance of someone in a game. If the mentioned person is an NBA player, there is a great possibility that the microblog is about a certain basketball game. If we find the person in a knowledge base, we could find the team he plays in and recommend this microblog to other fans of this team. On the other hand, sharing and commenting microblogs with names of NBA players suggests the user is a fan of NBA and basketball, we can recommend related news and business items to him. From this point of view, linking entities to encyclopedia helps to understand the topic of microblogs and is useful to other applications.

Judging the real indication of an entity in microblog could be challenging. The language of a microblog is more like spoken language than written language. Word usage in microblogs is much more irregular than traditional Web articles. There are three typical ways of using words. The first one is the vast usage of abbreviations. "国移动通信集团公司", or China Mobile Communications Corporation(CMCC) could be abbreviated to "中国移动" or "中移动". Obviously the latter two abbreviations are more popular on SNS for briefness. The second one is usage of nicknames or different ways of translation. For example, "禅师", or *The Zen Master*, refers to the NBA star *Philip Douglas Jackson*. Users who are familiar with these starts tend to use nicknames. The third one is typo, such as "阿里爸爸" instead of the correct name "阿里巴巴". "巴" and "爸" share the same input sequence of *ba* in Pinyin input method, so it is possible to make a mistake. On the other hand, some users may type a typo on purpose for many reasons,for instance, the original word is forbidden to appear in a microblog.

All these three types of word usage are obstacles to finding out what the entities stand for. But luckily, we could conclude some assumptions lying behind. The common essence of all the three cases before is that an entity could have multiple aliases. And the set of aliases remains stable in a relatively short period of time. The first two kinds of aliases are generally accepted, because they are formed by convention. Even kinds of typos could be stable because the reason could derive from the fact that different Chinese characters share the same Pinyin code. Based on these two assumptions, we could design methods to find aliases for each entity. Given a mentioned name that is informal or less frequently used, we can find its more formal or more frequently used names. That will help a lot in an entity linking task.

The rest of the paper is organized as follows. Section 2 reviews some related works. Section 3 describes the entity linking task and gives a formal definition. Section 4

introduces the algorithm framework we use with discussions about the alias list generation. Section 5 presents the experiment and analyzes some typical cases. Conclusion is drawn in Section 6.

2 Related Work

Named entity recognition is the foundation of entity linking. [3] applied Conditional Random Fields to discover the named entities. Our work is based on the assumption that entities have been pointed out from microblogs and the boundaries are correct. Other works focus on the disambiguation of named entities. [4] used bag-of-words model to compute similarities between documents and built a cross-document coreference system to disambiguate person names. [5] worked on named entity disambiguation with the help of encyclopedia. The disambiguating process is very similar with entity linking. [6] proposed a listwise learning-to-rank algorithm, ListNet, and illustrated its promising performance over other ranking algorithms in entity linking task. [7] introduces a framework of system to link the entities in articles to Wikipedia. This framework is reformed by [8] and SVM [9]is used to execute top 1 result validation.

3 Task Description and Problem Formulation

In this section, we are going to give a formal description of Entity Linking Task and review some existing methods. Learning-to-rank algorithms are applied and have shown promising performance in this task mainly in the environment of online news and articles.

3.1 NLP&CC 2013 Entity Linking Evaluation Task

The NLP&CC 2013 Entity Linking Task requires linking the entities appeared in microblogs to the knowledge base entries, which is similar with Entity Linking Track in TAC-KBP task. The Entity Linking Track in TAC-KBP task contributes to constructing and expanding the knowledge base mainly with online news corpora.

3.2 Problem Formulation

In the entity linking task, there is a set of documents $DS = \{D_1, D_2, ..., D_M\}$. Each document D_i has a list of entities $EL = \{E_1, E_2, ..., E_n\}$ to be linked. The knowledge base is a set of entries $KB = \{K_1, K_2, ..., K_m\}$. Each entry K_j has multiple fields, including $Name$, ID, $AnchorText$, Doc and other fields which depend on what the entry represents. The task is to link each entity in a document to the corresponding entry in knowledge base. If there is no entry matching the entity, NIL should be returned.

4 Algorithms

The outline of our algorithm framework is as follows.

1. For each entity, find a list of possible aliases.
2. Search for potential candidate in the knowledge base and return a list of entries.
3. Use learning-to-rank algorithm to find the entry with the highest rank score.
4. Use binary classifier to validate the top 1 entry. If the entry is accepted by classifier, return its ID. Otherwise return NIL.

In Step 1, we design several methods to find a list of aliases for each entity. It helps to find potential candidate, and provide features to train learning-to-rank algorithm and binary classifier. The details of related methods would be described further in Section 4.1. In Step 2, we search for all entries in which the entity or one of its aliases occurs. If no candidate knowledge entry is found, our system just skips the rest two steps and return NIL for this entity. In Step 3, we apply ListNet as ranking algorithm and return the knowledge entry with the highest score. We will discuss this algorithm in detail in Section 4.2. In Step 4, we build a binary classifier to judge whether to accept the entity.

4.1 Alias List Generation

In Microblog environment, one specific entity might be mentioned by users in many different ways. For example, when people talk about something related to Kobe Byrant, who is a very famous star in National Basketball Association (NBA), they may mention him as *'Black Mamba'* or *'KB24'*, which are his popular nicknames or aliases. In the knowledge base, for some entities, we have some attributes about their aliases, for example, nicknames, used names and so on. However, it is difficult to maintain a complete alias list for every entity. This is because aliases of an entity always change over time, language and geographic area. To solve this problem, we tried to utilize different features extracted from search engine click-through data, from search engine result page and from a local search engine built on knowledge base.

In the following subsections, we will introduce the methods to mining possible aliases for an entity.

Random Walk on Click-through Bipartite Graph. When people use search engines, the interaction process between users and the search engine will be record as many kinds of logs. One of them is the Click-through log which records two kinds of user behavior: **submitting query** and **clicking search result**.

The click-through log C can be represented as a set of triples $\langle q, u, f_{qu} \rangle$, where q is a query, u is a URL, and f_{qu} is the times URL u is clicked when query q is issued. Define $Q = \{q | q\ appears\ in\ C\}$,and $U = \{u | u\ appears\ in\ C\}$. Click-through data C can be presented as another equivalent form – a click-through bipartite graph $G = (Q, U, E)$. There are two types of nodes in the graph, queries and URLs. A sample portion of a bipartite graph constructed with search engine log, as shown in Figure 1.

A widely accepted assumption is that queries leading to same URLs usually reflect similar user intent. Therefore, the aggregation of a large number of user clicks is likely

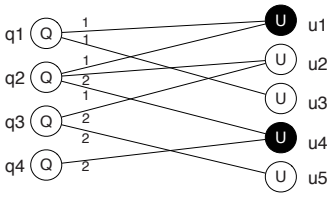

Fig. 1. An example of query-URL bipartite graph

to provide valuable 'soft' evidence of whether different queries are similar or not. The click-through graph can be constructed either at the page level or at the site level. In this study, we choose the URL itself to construct graph because the content of a Web site might be very comprehensive.

On the Query-URL bipartite graph, we can define a 2-step random walk process on the graph as follows:

$$s(q_i, q_j) = \sum_{k \in U} p_{i,k} \cdot p_{k,j} \tag{1}$$

where

$$p_{i,j} = \frac{f_{i,j}}{\sum_{(i,k) \in E} f_{i,k}} \tag{2}$$

$p_{i,j}$ can be interpreted as the transition probability from Node i to Node j and $s(q_i, q_j)$ can be interpreted as the similarity between query i and query j. Therefore, for each query which is going to be linked to knowledge base, we can generate similar queries as a alias list ranked by similarity in descending order. It is very possible that the alias contains the name which is the entity name in knowledge base.

For example, for query "禅师" and query "禅师菲尔", the alias list is shown in Table 1. Both of these two queries refer to the NBA player *Philip Douglas Jackson*, who is also called *Phil Jackson* for short. Alias lists for both queries contain the alias of "杰克逊", the Chinese name for *Jackson*. For query "皇马", we can find the full name "皇家马德里" in its alias list, which is translated from *Real Madrid*.

However, this method may fail for some long-tail queries, because it is probably that the query does not exist on the graph or is not connected with other query nodes. The more data we use to build the graph, the more likely we can find some similar queries for most of the queries.

N-gram Model on Search Engine Result Page. In modern search engines, when a user submits a query, the search engine will return a list of ranked results. All the results are closely related to the query, which provide a good resource to extract aliases of the query. We submit each of the entities to a popular Chinese search engine and extract the title and summary of each search result. To find the possible aliases of the entities, we propose the n-gram model. We change k from 2 to 10 and extract all the k-grams (continuous k characters) of the titles and summaries of the results. We retain the most

Table 1. Aliases from Random Walk on Click-through Bipartite Graph

Query	Aliases	Probability
禅师	禅师	0.0890
	杰克逊	0.0494
	禅师杰克	0.0398
	禅师杰克逊	0.0398
	hanshi	0.0151
	chanshi	0.0151
禅师菲尔	**杰克逊**	0.2222
	克逊	0.2222
	菲尔杰克逊	0.1277
	菲尔杰克	0.1277
	教过NBA全	0.0500
	过NBA全明星	0.0500
皇马	皇马	0.8214
	马德里	0.2997
	家马德里	0.2990
	皇家马德里	0.2984
	皇马球迷俱乐	0.1351
	皇马球迷俱乐部	0.1351

Table 2. Aliases from N-gram Model on Search Engine Result Page

Length	Results of titles	Results of summaries
2	TV	TV
3	BTV	BTV
4	北京电视	北京卫视
5	北京电视台	北京电视台
6	视台BTV在	201305
7	视台BTV在线	BTV北京卫视
8	电视台BTV在线	北京电视台养生堂
9	京电视台BTV在线	V北京电视台养生堂
10	北京电视台BTV在线	BTV北京电视台养生

frequent k-gram as the alias of length-k of the entity. We apply the n-gram model on the titles and summaries separately and get 2 lists of possible aliases. For example, when we apply the model on the entity *'BTV'*, which stands for the Beijing TV station, the results of titles and summaries with length from 2 to 10 are shown in Table 2.

From the results, the aliases "北京卫视" and "北京电视台" are two Chinese names of *'BTV'* and they are exactly what we want.

Local Search on Knowledge Base. It is very possible that an entity which is going to be linked to knowledge base does not appear in the target entity's name, but appears in its Infobox or body text. To solve this problem, we tried to build a local search engine which index the knowledge base including entity name, Infobox and full body text. In

the search engine, each entity in the knowledge base is organized as a document while each entity going to be linked is regarded as a query. When a query is issued, top 30 entities returned by search engine are selected as a entities candidate list. It is ranked by relevance given by search engine in descending order.

Moreover, when indexing the knowledge base, different domains of entities have different weights. An entity's name has the greatest weight, and its Infobox data follows, and its body text has the smallest weight. The relevance is calculated based on Vector Space Model (VSM). A simple example for entity "北航" is presented as following. In the Infobox of knowledge entry "北京航空航天大学", the university's abbreviations are provided as "北航" and *BUAA*. When searching the entity in our local search engine, this entry is returned at the top of all results.

```
<entity>
<entity_id>KBBD043253</entity_id>
<category>科学\数理化\航空航天\北京航空航天大学.html</category>
<name>北京航空航天大学</name>
<fact>
<中文名>北京航空航天大学</中文名>
<外文名>Beihang University</外文名>
<简称>北航（BUAA）</简称>
<校训>德才兼备，知行合一</校训>
<创办时间>1952年10月25日</创办时间>
<类别>公立大学</类别>
...
</fact>
</entity>
```

4.2 ListNet

ListNet is a supervised learning-to-rank algorithm proposed by [6]. It uses a probabilistic method and a listwise loss function to rank a list of objects. In Step 3, we apply ListNet to rank the candidate knowledge entries for each entity. Given entity e_i and a set of candidates $\{K_1^{(i)}, K_2^{(i)}, ..., K_{n_i}^{(i)}\}$, if each $K_j^{(i)}$ is assigned with a score s_j, then we can get a permutation π according to the score, and the probability would be

$$P_s(\pi) = \prod_{t=1}^{n_i} \frac{\exp(s_t)}{\sum_{l=t}^{n_i} \exp(s_l)} \tag{3}$$

If the permutation of the top k candidates is $G_k = (K_{j_1}^{(i)}, K_{j_2}^{(i)}, ..., K_{j_k}^{(i)})$, the probability would be

$$P_s(G_k) = \prod_{t=1}^{k} \frac{\exp(s_{j_t})}{\sum_{l=t}^{n_i} \exp(s_{j_l})}$$

Then we assign each $K_j^{(i)}$ with a score $y_j^{(i)}$, and define a ranking function f_ω that can calculate a score $z_j^{(i)}$ for each candidate

$$z_j^{(i)} = f_\omega(x_j^{(i)}) \tag{4}$$

where $x_j^{(i)}$ is the feature vector of entry $K_j^{(i)}$ given e_i. The probability for permutation G_k to be the top k candidates would be $P_{y^{(i)}}(G_k)$ and $P_{z^{(i)}}(G_k)$ with the two scoring strategy. ListNet algorithm uses cross entropy as loss function

$$L(y^{(i)}, z^{(i)}) = -\sum_{\forall g \in G_k} P_{y^{(i)}}(g) \log(P_{z^{(i)}}(g)) \tag{5}$$

The gradient of cross entropy could be calculated by

$$\Delta\omega = \frac{\partial L(y^{(i)}, z^{(i)})}{\partial \omega} = -\sum_{\forall g \in G_k} \frac{\partial P_{z^{(i)}}(g)}{\partial \omega} \frac{P_{y^{(i)}}(g)}{P_{z^{(i)}}(g)} \tag{6}$$

In stochastic gradient descent, ω is updated with $-\eta\Delta\omega$ in each iteration. Here η is the learning rate.

5 Experiments

5.1 Dataset

We use data set from NLP&CC 2013 Entity Linking Evaluation Track. The data set is divided into two parts, the training set and the testing set. The training set is annotated. The annotating set is a subset of testing set, annotated to compute the precision, recall, and F1 measure. The scale of each set is shown in Table 3.

Table 3. Comparison results of accuracies

Data Set	# of microblogs	# of entities	Linkable	Unlinkable
Training Set	176	248	141	107
Testing Set	787	1249	–	–
Annotating Set	560	826	421	405

The knowledge base we use is a subset of Baidu Baike, a Chinese online encyclopedia just like Wikipedia. In this dump provided by NLP&CC, nearly 45, 000 knowledge entries are included. Each entry has a name, id, anchor text, infobox and an article.

5.2 Experiment Setup

We find the alias list for each entity in advance, using method described in Section 4.1. We build the local search engine with Lucene.

In entity linking task, we only care about whether the entry with highest score is correct. So we use top 1 permutation in ListNet. During training process, we manually assign score 1 to the correct knowledge entry, and 0 to others.

As for top 1 validation, we use SVM as the binary classifier. To balance the number of negative and positive samples, we select the correct entry for each entity as positive sample and other 2 entries with the highest score as negative samples. We append the rank score returned by ListNet to the feature vector input into ListNet, and generate the input feature vector for SVM.

5.3 Feature Selection

Considering the relationships among context, knowledge base, entity and aliases, we use three sets of features described below.

1. The compatibility of an entity and a knowledge entry.
   - Whether entity is substring of entry's name or anchor text, 1 for true and 0 for false.
   - The edit distance between the entity and the entry's name.
   - The minimum edit distance between the entity and each field of the entry's infobox.
   - Number of times the entity occurs in entry's article.
   - Whether this entry is in the top 30 results when searching the entity in local search engine.

2. The compatibility of aliases and a knowledge entry.
   - Whether one of aliases is substring of entry's name or anchor text, 1 for true and 0 for false.
   - The length of the longest common sequence of each field of entry's infobox and each alias.
   - The minimum edit distance between the the entry's name and each one of alias.
   - The sum of times each alias occurs in entry's article.

3. The compatibility of other entities co-occurs in the same microblog and the entries in knowledge base.
   - The sum of times each other entity in the same microblog occurs in the entry's article.

5.4 Experiment Result

The result on annotating set is shown in Table 4.The result of 'THUAI-RUN1' is given by this framework. We achieve an overall precision of 87.89% with a rank of 6th position. Considering the in-KB Results, we achieve a precision of 87.59% with a rank of 3rd position. This fact again illustrates that as a learning-to-rank algorithm, ListNet is suitable for entity linking. One limit of the performance is the inadequate size of training set. What's more, we tie the first place on recall of 93.83% among NIL Results. That means our binary classifier used for top 1 validation tends to refuse the result. If we had found better features for input of SVM, the performance might have been better.

Table 4. NLP&CC Evaluation Result

ID	Overall Results		in-KB Results			NIL Results		
	Correct	Precision	Precision	Recall	F1	Precision	Recall	F1
1	628	0.7603	0.7576	0.6532	0.7015	0.7624	0.8716	0.8134
2	638	0.7724	0.7819	0.6556	0.7132	0.7653	0.8938	0.8246
3	664	0.8039	0.7360	0.7482	0.7420	0.8769	0.8617	0.8692
4	731	0.8850	0.8662	0.8456	0.8558	0.9036	0.9260	0.9146
5	589	0.7131	0.7339	0.6583	0.6940	0.7694	0.8516	0.8084
6	622	0.7530	0.7452	0.7864	0.7652	0.8536	0.8047	0.8284
7	751	0.9092	0.8983	0.8812	0.8897	0.9201	0.9383	0.9291
8	743	0.8995	0.8859	0.8670	0.8764	0.9130	0.9333	0.9231
9	698	0.8450	0.8342	0.7767	0.8044	0.8548	0.9160	0.8844
10	697	0.8438	0.8142	0.7910	0.8024	0.8729	0.8988	0.8856
11	730	0.8838	0.8602	0.8480	0.8541	0.9075	0.9210	0.9142
12	730	0.8838	0.8602	0.8480	0.8541	0.9075	0.9210	0.9142
13	702	0.8499	0.8523	0.7815	0.8154	0.8477	0.9210	0.8828
14	701	0.8487	0.8497	0.7791	0.8129	0.8477	0.9210	0.8828
THUAI-RUN1	**726**	**0.8789**	**0.8759**	**0.8219**	**0.8480**	**0.8817**	**0.9383**	**0.9091**
THUAI-RUN2	**536**	**0.6489**	**0.6726**	**0.5416**	**0.6000**	**0.6324**	**0.7605**	**0.6906**
16	536	0.6489	0.6726	0.5416	0.6000	0.6324	0.7605	0.6906
17	673	0.8148	0.8407	0.8070	0.8235	0.8775	0.9141	0.8954
18	675	0.8172	0.8504	0.8120	0.8308	0.8731	0.9141	0.8931

6 Conclusion and Future Work

While data mining on microblogs has attracted more and more attention, entity linking on microblogs is an important foundation of many other applications. We propose a novel framework combining different possible aliases mining together to help linking entities from microblogs to knowledge base. This method utilizes external information such as query logs created by Web users. The effectiveness of this method is illustrated by the result of NLP&CC Evaluation Entity Linking Track. In our future work, we could use more features such as similarity on knowledge graphs to improve the performance.

References

1. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of the 19th International Conference on World Wide Web. ACM (2010)
2. Mathioudakis, M., Koudas, N.: Twittermonitor: trend detection over the twitter stream. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. ACM (2010)
3. McCallum, A., Li, W.: Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, vol. 4, Association for Computational Linguistics (2003)

4. Bagga, A., Baldwin, B.: Entity-based cross-document coreferencing using the vector space model. In: Proceedings of the 17th International Conference on Computational Linguistics, vol. 1. Association for Computational Linguistics (1998)
5. Bunescu, R.C., Pasca, M.: Using Encyclopedic Knowledge for Named entity Disambiguation. In: EACL, vol. 6 (2006)
6. Cao, Z., et al.: Learning to rank: from pairwise approach to listwise approach. In: Proceedings of the 24th International Conference on Machine Learning. ACM (2007)
7. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management. ACM (2007)
8. Zheng, Z., et al.: 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. Association for Computational Linguistics (2010)
9. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory. ACM (1992)
10. Salton, G., Wong, A., Yang, C.-S.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
11. Craswell, N., Szummer, M.: Random walks on the click graph. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2007)