Time-Aware Entity Search in DBpedia

Lei Zhang[1], Wentao Chen[1], Thanh Tran[2], and Achim Rettinger[1]

[1] Institute AIFB, Karlsruhe Institute of Technology (KIT), Germany
[2] San Jose State University, USA
{l.zhang,rettinger}@kit.edu,wentao.chen@student.kit.edu
ducthanh.tran@sjsu.edu

Abstract. Searching for entities is a common user activity on the Web. There is an increasing effort in developing entity search techniques in the research community. Existing approaches are usually based on static measures that do not reflect the time-awareness, which is a factor that should be taken into account in entity search. In this paper, we propose a novel approach to time-aware entity search in DBpedia, which takes into account both popularity and temporality of entities. The experimental results show that our approach can significantly improve the performance of entity search with temporal focus compared with the baselines.

1 Introduction

The ever-increasing quantities of entities in large knowledge bases on the Web, such as Wikipedia, DBpedia and YAGO, pose new challenges but at the same time open up new opportunities of information access on the Web. In this regard, many research activities involving entities have emerged in recent years. Entity search has become a major area of interest because users often search for specific entities instead of documents, which helps users to directly find the intended information. On the other hand, time-awareness is a crucial factor in entity search due to the high dynamics of the underlying data.

In this paper, we address the problem of searching entities for a given user query in a time-aware setting. We formulate the *time-aware entity search* task as follows: given an entity collection $E = \{e_1, e_2, \cdots, e_N\}$[3], the input is a user query $q = \langle s, t \rangle$, which consists of an entity name s and a time range of contiguous days $t = \{d_1, d_2, \cdots, d_M\}$, where d_i represents a specific day, and the output is the intended entity matching s of particular interest within t. Our approach allows users to restrict their search interests to a time range. However, in a real-life search scenario, users usually do not specify the time range explicitly. In this case, our system can easily use the current day on which users issue the query and a certain period of time before it (e.g., one week) as the time range. Assuming that users search for the entity name *Irving* on *2014-02-21* and the intended entity is *Kyrie Irving*, who won the NBA All-Star Game MVP Award

[3] We use DBpedia as the entity collection, which extracts various kinds of structured information from Wikipedia and each DBpedia entity is tied to a Wikipedia article.

on *2014-02-17*, the time range can be specified by our system as, for example, one week *from 2014-02-15 to 2014-02-21*.

The challenge of our entity search task is name ambiguity, i.e., a entity name could refer to different entities. For entity linking [1,2], where the goal is to link words or phrases in text documents with entities in knowledge bases, the entity disambiguation can be performed based on the context in documents. Without such contextual information in our entity search scenario, we propose a novel approach by taking into account both *popularity* and *temporality* of entities.

2 Approach

In this section, we present our approach to *time-aware entity search* in DBpedia, where each DBpedia entity corresponds to a Wikipedia article. In order to rank entities for a query $q = \langle s, t \rangle$, we calculate the score $\text{Score}(e, s, t)$ for each entity e based on different components, which will be discussed in the following.

Candidate Entity Generation. Given a query entity name s, we first generate a set of candidate entities matching s, denoted as E_s. For this, we need to extract the *surface forms* of each entity, i.e., words or phrases that can be used to refer to the corresponding entity. Wikipedia provides several structures that associate entities with their *surface forms*. Similar to [2], we make use of the following structures in Wikipedia: (1) *titles of articles*: the title of each Wikipedia article is generally the most common name for the entity; (2) *redirect pages*: a redirect page exists for each alternative name that can be used to refer to an entity; (3) *disambiguation pages*: when multiple entities could have the same name, a disambiguation page in Wikipedia containing the references to those entities is usually created; (4) *anchor texts of hyperlinks*: articles in Wikipedia often contain hyperlinks with anchor texts pointing to the mentioned entities, which provide a very useful source of surface forms of linked entities.

Given an entity name s, the probability $P(e|s) = \frac{C(e,s)}{\sum_{e_i \in E_s} C(e_i,s)}$, where $C(e, s)$ denotes the number of links pointing to e with anchor text s, has been widely used to model the likelihood of observing $e \in E_s$ [1,2]. However, we do not integrate this distribution into our approach since it would unduly favor entities with more links and might result in poor performance as shown in the experiments.

In order to rank the candidate entities in E_s w.r.t. the time range t, we calculate $\text{Score}(e, s, t)$ for each entity $e \in E_s$ as

$$\text{Score}(e, s, t) = \text{Score}_{popu}(e, s, t) \cdot \text{Score}_{temp}(e, s, t) \tag{1}$$

where $\text{Score}_{popu}(e, s, t)$ represents the popularity of e and $\text{Score}_{temp}(e, s, t)$ captures its temporality. Both scoring functions employ the page view statistics, which capture the number of times Wikipedia pages have been requested, and thus can be treated as a query log of entities.

Popularity Ranking. An entity well-known to most people usually gets more page views than others that are relatively obscure. For example, the NBA player Michael Jeffrey Jordan get more page views than the Berkeley professor

Michael I. Jordan. Based on that, we use the page view statistics within a long period of time T (e.g., one or several years) to calculate $\text{Score}_{popu}(e, s, t)$ as

$$\text{Score}_{popu}(e, s, t) = \sum_{d_i \in T} C(e, d_i) \tag{2}$$

where $C(e, d)$ denotes the number of page views of entity e on date d. We choose T as the union of the time range t and the recent year before t in our experiments, which makes our popularity ranking function also time-dependent. As shown in the experiments, our popularity ranking achieves better results than the approach based on the well-known PageRank algorithm.

Temporality Ranking. The score $\text{Score}_{temp}(e, s, t)$ is of particular interest in this work. The intuition is an entity will likely get more page views when an event about it takes place. For example, an Olympic athlete will get more page views when he has won a medal during the Olympics. Therefore, we use page views of each entity as a proxy for interest and equate page view spike with it.

In this regard, we track per-day page views and maintain a sliding window of n days (with $n = 10$ in our experiments) over the previous page view counts to compute the spikes for each entity e. We first compute the mean $\mu(e, d)$ and standard deviation $\sigma(e, d)$ of page views for entity e w.r.t. date d

$$\mu(e, d) = \frac{1}{n} \sum_{d_i = d-n}^{d-1} C(e, d_i) \tag{3}$$

$$\sigma(e, d) = \sqrt{\frac{1}{n} \sum_{d_i = d-n}^{d-1} (C(e, d_i) - \mu(e, d))^2} \tag{4}$$

Similar to [3], we then calculate the page view spike $S(e, d)$ as

$$S(e, d) = \begin{cases} \frac{C(e,d) - \mu(e,d)}{\sigma(e,d)} & \text{if } \frac{C(e,d) - \mu(e,d)}{\sigma(e,d)} > k, \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where only the page view count $C(e, d)$ that is large enough, i.e., $\frac{C(e,d) - \mu(e,d)}{\sigma(e,d)} > k$ (k is set as 0.5 in our experiments), is taken into account to make a contribution to the page view spike. Based on that, we calculate $\text{Score}_{temp}(e, s, t)$ as

$$\text{Score}_{temp}(e, s, t) = \sum_{d_i \in t} S(e, d) \tag{6}$$

3 Evaluation

Existing datasets for evaluating entity search usually only aim to quantify the degree to which the entities are relevant to the keyword query without involving time aspects, which makes these datasets unsuitable for our task. Therefore, we

Methods	recall@1	recall@5	recall@10	recall@20	MRR
LinkProb	0.00	0.22	0.38	0.59	0.11
PageRank	0.31	0.75	0.87	0.96	0.49
Popu	0.56	0.91	0.97	0.97	0.69
Temp	0.72	0.94	0.97	0.97	0.81
Popu+Temp	**0.78**	**0.97**	**1.00**	**1.00**	**0.85**

Table 1: The Experimental Results.

asked volunteers to provide queries consisting of the entity name and the time range along with the underlying information needs, i.e., the intended entities. By filtering out the unambiguous queries, for which there is only one candidate entity, it results in a final dataset containing 30 queries. As quality criteria, we consider recall at cutoff rank k (recall@k) and Mean Reciprocal Rank (MRR).

We conducted the experiments with several approaches: (1) the baseline approach based on the link probability $P(e|s)$, as discussed in Sec. 2, denoted as *LinkProb*; (2) the baseline approach based on PageRank algorithm performed on Wikipedia link structures, denoted as *PageRank*; (3) our approach using only popularity ranking (Eq. 2), denoted as *Popu*; (4) our approach using only temporality ranking (Eq. 6), denoted as *Temp*; (5) our approach using both *Popu* and *Temp* ranking (Eq. 1), denoted as *Popu+Temp*.

The experimental results are shown in Table 1. It is observed that our approach with both popularity and temporality ranking yields the best results. Compared with the baselines, our approach achieves a significant performance improvement. While both ranking functions used in our approach contribute to the final performance improvement, temporality ranking contributes the most and achieves the best results among the individual ranking functions.

4 Conclusions

In this paper, we address the problem of time-aware entity search, where we believe that time-awareness is a very important issue. For this purpose, we defined a scoring function that aims to rank entities based on both popularity (for a long period of time) and temporality (regarding the user's time range of interest). We have experimentally shown that our approach achieves a significant improvement over the baselines in terms of recall@k and MRR.

Acknowledgment. This work is supported by the European Community's Seventh Framework Programme FP7-ICT-2013-10 (XLiMe, Grant 611346).

References

1. Milne, D.N., Witten, I.H.: Learning to link with wikipedia. In: CIKM. (2008) 509–518
2. Shen, W., Wang, J., Luo, P., Wang, M.: LINDEN: linking named entities with knowledge base via semantic knowledge. In: WWW. (2012) 449–458
3. Osborne, M., Petrović, S., Mccreadie, R., Macdonald, C., Ounis, I.: Bieber no more: First story detection using twitter and wikipedia. In: #TAIA. (2012)