Leveraging Related Entities for Knowledge Base Acceleration

Xitong Liu
University of Delaware
Newark, DE, USA
xtliu@udel.edu

Hui Fang
University of Delaware
Newark, DE, USA
hfang@udel.edu

ABSTRACT

Knowledge bases such as Wikipedia have been shown to be effective to improve the performance in many information tasks. Clearly, the effectiveness is based upon the quality of these knowledge bases. A high-quality knowledge base should have up-to-date complete information. However, constructing a high-quality knowledge base is not an easy task because it would require significant manual efforts to collect relevant documents, extract valuable information and update the knowledge bases accordingly. In this paper, we aim to automate this labor-intensive process. Specifically, we focus on how to collect relevant documents with regard to an entity from sheer volume of Web data automatically. To solve the problem, we propose to construct the profile of the entity by leveraging a set of its related entities and then discuss how to use the training data to weight the related entities. Experiments over the TREC 2012 KBA collection shows that the proposed method can outperform state-of-the-art methods.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering

General Terms

Algorithms, Experimentation

Keywords

Knowledge base acceleration; information filtering; cumulative citation recommendation

1. INTRODUCTION

Knowledge bases (e.g., Wikipedia, DBPedia) have been shown to be useful in many information tasks including query expansion [5, 13], question answering [1], entity retrieval [3] and entity linking [10]. As the sheer amount of online Web data are produced everyday and keep growing exponentially, it is becoming more crucial to maintain a high quality and up-to-date knowledge base to reflect the fast changing facts of entities. However, populating knowledge bases relies on users' manual efforts, which inevitably procrastinates the update process due to the limitation of labors. Existing study [6] reveals that notable time lag between publication date of Web documents and date of being referred can be observed for most Wikipedia citations, and the median time lag is over one year. Moreover, there exist many stale entities in large knowledge bases due to the lack of editors' domain knowledge, making the maintenance even more challenging. It is therefore crucially beneficial to make the knowledge bases population managed in an automatic way to accelerate the process of keeping them update-to-date and save budgets eventually.

In this work, we focus on *automatically collecting relevant documents* of a topic entity from large Web collections, which is a key component of knowledge base update process. We propose an approach to iteratively estimate weighting of related entities to enrich the profile of topic entity and apply the entity profile to select relevant documents. In 2012 the Text REtrieval Conference (TREC) launched Knowledge Base Acceleration (KBA) track [6] with the goal to call for research endeavors on accelerating the update of large-scale knowledge bases, and defined the Cumulative Citation Recommendation (CCR) task. The ultimate goal is to recommend a set of citation-worthy documents, which would potentially contribute to entity profiles, to knowledge base maintainers to accelerate the following update process thereafter. Since the CCR task fits our problem setup well, we conduct experimental evaluation on the KBA track data set to examine the effectiveness of our approach. Results analyses show that our method delivers superior performance over three strong baselines.

2. RELATED WORK

There have been constant research efforts on knowledge base construction recently. YAGO [12] is a well recognized project aiming at building an extensible and high quality ontology by extracting information from Wikipedia and WordNet. DBPedia [2] initiates a community effort to construct universal accessible linked data cloud through extracting structured information from Wikipedia (mainly from Infobox). Started from 2009, the Text Analysis Conference (TAC) has been hosting Knowledge Base Population track [7] with focuses on three tasks: entity linking, slot-filling and cold start KBP. The entity linking task resembles our work most

Web-KR'13, November 1, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2512405.2512407 ...$15.00.

in the sense that it focuses on linking the entity mentions in unstructured documents to entities in existing knowledge base. Wikify! [10] tackles the similar problem in two steps: detecting entity mentions and linking them back to Wikipedia entries.

The TREC 2012 KBA track defined the CCR task, i.e., selecting documents which are highly relevant to given Wikipedia entities from a chronically organized stream corpus [6]. 11 teams participated and 43 runs were submitted. Top ranked approaches include supervised learning powered classification [8] and exploiting entity profile [9]. More recent research work [4] reveals that classification based approach delivers superior performance with carefully selected features. Different from the classification based approaches, our work employs an iterative algorithm to weight related entities and build enriched entity profile which is capable of selecting relevant documents both effectively and efficiently.

3. METHODS

3.1 Problem Setup

Given a document collection $\mathcal{D}$ and a topic entity E from a knowledge base $\mathcal{K}$, our goal is to collect a set of documents $\mathcal{D}_E$, where $\forall d \in \mathcal{D}_E$, d is relevant to E in the sense that it bears some information related to E, and the information may be incorporated into the profile of E. We denoted the relevance between d and E as $r(d, E) = \{0, 1\}$ where 0 means irrelevant and 1 means relevant, respectively. Besides, a set of labeled relevant documents $\mathcal{D}_{rel}$ is provided, where $\forall d \in \mathcal{D}_{rel}$, $d \notin \mathcal{D}$ and $r(d, E) = 1$.

A simple strategy to determine to the relevance of d and E is to detect whether d mentions E. However, this approach suffers from high false positive rate for the following reasons:

- Entity name may be ambiguous, and multiple entities may share the same surface name. For example, "Michael Jordan" may refer to the American basketball player "Michael J. Jordan" or the Berkeley professor "Michael I. Jordan".
- Mentioning does not necessarily imply relevance. A document may marginally mention an entity in the context of discussing other entities. This is especially common in webpages as a HTML page consists of several parts and the topic entity may be mentioned in some parts other than the main body (e.g., footer, sidebar, etc.).

To address the challenges, we resort to use the mentioning of both topic entity and its related entities in the same document as criterion of determining relevance. The underlying assumption is that when a topic entity is discussed, its related entities are highly likely to be mentioned in the same context. Entities are regarded as related if there are certain relations between them. For example, "NBA" and "Chicago Bulls" are related entities to "Michael J. Jordan" as "Michael J. Jordan" was a "NBA" player in "Chicago Bulls". With the help of related entities, both challenges can be resolved respectively. When a document mentions "NBA" and/or "Chicago Bulls" besides "Michael Jordan", it is highly likely that the document is talking about the NBA player rather than the Berkeley professor, thus addressing the first challenge. Moreover, mentioning related entities as well as topic entity implies that the document bears some information about the topic entity rather than marginally mentioning it, solving the second challenge.

We first collect a set of related entities $R(E)$ from $\mathcal{K}$. Since knowledge bases are commonly organized in a structured way that entities are inter-connected and the link between a pair of connected entities bears their relation. In the case of Wikipedia, hyperlinks connect entry pages, while in DBPedia predicate fields connect subject fields and object fields. By following the links connecting E in $\mathcal{K}$, we can easily collect all the related entities $R(E)$. We now discuss how to incorporate $R(E)$ into the determination of $r(d, E)$.

3.2 Related Entity based Approach

Related entities serve as complementary information to help determine the relevance between d and E, however, not all related entities are equally helpful. For example, "Michael Jordan" was born in "Brooklyn, New York" and was a "NBA" player. Clearly, "NBA" is more helpful to discrete "Michael Jordan" than "Brooklyn, New York" when they both co-occur with "Michael Jordan" in the same context. It is therefore necessary to weight related entities based on their relevance to the topic entity.

Given a document d and topic entity E, we need to estimate how likely d is relevant to E (i.e., d bears some information about E). As discussed in Section 3.1, the relevance score is influenced by the mentioning of E as well as its related entities. Formally, we define it as:

$$score(d, E) = \alpha \cdot mention(d, E) + \beta \cdot \sum_{e \in R(E)} occ(d, e) \cdot w(E, e), \tag{1}$$

where $mention(d, E)$ is a binary function which indicates whether the document d mentions E (1 means d mentions E and 0 otherwise), $occ(d, e)$ denotes the number of occurrences of e in d. Clearly, the more occurrences of e in d, the more confident we can infer that d is relevant to E. $w(E, e)$ serves as the prior weight of e to favor important entities and penalize trivial ones. α and β are coefficients to balance the impacts of two score components. The final decision of $r(e, d)$ can be determined by comparing $score(d, E)$ with a cutoff threshold. Documents with relevance score above the cutoff threshold will be labeled as relevant, and others will be labeled as irrelevant.

We now discuss how to estimate the prior weight of related entity $w(E, e)$. By using the labeled relevant document set $\mathcal{D}_{rel}$ as guide, we propose an algorithm which estimates the weight of related entities by maximizing the performance gain in an iterative way. The details are described in Algorithm 1, which consists of the following major steps.

1. The weight for each related entity is initialized to 1 at line 3.
2. $R_{sel}(E) \subseteq R(E)$ is the set of selected entities with weight re-estimated and $Gain(\mathcal{D}_{rel}, R_{sel}(E))$ calculates the performance gain when $R_{sel}(E)$ is applied on $\mathcal{D}_{rel}$ using the approach in Equation 1 based on some evaluation measure. In each iteration, an entity e^* which maximizes the performance gain will be selected as candidate at line 10.
3. Convergence check is conducted by checking whether adding e^* to $R_{sel}(E)$ will lead to further performance improvement at line 13. If so, the weight $w(E, e)$ will

Algorithm 1 Related Entity Weighting

Input: topic entity E, related entity set $R(E)$, labeled relevant document set $\mathcal{D}_{rel}$

```
1: /*Initialization*/
2: for e ∈ R(E) do
3:    w(E, e) = 1
4: end for
5: R_sel(E) ← {}
6: R_left(E) ← R(E)
7: G = 0
8: while R_left(E) ≠ ∅ do
9:    /* Select the entity which maximizes performance
      gain when added to R_sel(E) */
10:   e* = arg max_{e∈R_left(E)} Gain(D_rel, R_sel(E) ∪ {e})
11:   G' = Gain(D_rel, R_sel(E) ∪ {e*})
12:   ΔG = G' − G
13:   if ΔG > 0 then
14:      /* Re-estimate the weight */
15:      w(E, e) = weight(e*, R_sel(E))
16:      /* Incrementally augment R_sel(E) */
17:      R_sel(E) = R_sel(E) ∪ {e*}
18:      R_left(E) = R_left(E) \ {e*}
19:      G = G'
20:   else
21:      /* No further performance improvement */
22:      for e ∈ R_left(E) do
23:         w(E, e) = 0
24:      end for
25:      R_left(E) ← ∅ /* Force quit the loop */
26:   end if
27: end while
```

be re-estimated at line 15 and e^* is added to $R_{sel}(E)$ then at line 17. If not, the algorithm converges at the local optimum as no further increment can be achieved.

4. If the algorithm converges, the weight of all the entities which have not been selected before (i.e., $R_{left}(E)$) will be assigned to 0, which means they will not take any effect in Equation 1 eventually. The iteration is forced to end then.

There are several ways to estimate $weight(e^*, R_{sel}(E))$ at line 15. One simple strategy is to assign non-zero uniform weight (e.g., 1) for each of them. Alternatively, we assign a linear decaying weight based on the order e is selected at line 10:

$$weight(e^*, R_{sel}(E)) = |R(E)| - |R_{sel}(E)| \quad (2)$$

Clearly, entities selected earlier will be favored more than those selected later based on the assumption that in each iteration the entity selected at line 10 contributes more to performance gain than others thus deserves higher weight.

4. EXPERIMENTS

4.1 Experimental Setup

Data Collection. We use the document collection (called KBA Stream Corpus 2012) released with TREC 2012 KBA track as testbed to evaluate our methods. All the documents were harvested from three categories of sources:

- **Linking:** URL list from bitly.com;
- **Social:** blogs and forums with rich category metadata;
- **News:** a set of global public news wires.

The documents are organized in chronological order and each document has associated with timestamp, indicating when it was published (or fetched), as well as a unique ID. Documents in the same hour are stored in the same batch with total of 4,973 hours, ranging from October 2011 to April 2012. We use the cleansed-only version which only contain visible text part of English HTML documents, and it consists of 367,660,530 documents with size of 275GB compressed.

Topic Entities. Along with the data collection, a set of 29 topic entities (27 persons and 2 organizations) are provided, denoted as *target entities*. All the entities are chosen from Wikipedia dump of January 2012, each of which is represented by a "urlname" as unique ID. Topic entities are chosen based on the criterion that they have rich link relations with other entities in Wikipedia, making it feasible to leverage the related entities.

Training and testing data. The documents are split into two parts by time: training data (from 10/07/2011 to 12/31/2011) and testing data (from 1/1/2012 to 05/02/2012). Judgments are made manually by accessors in both parts at four consecutive levels: garbage (e.g., spam), neutral (not relevant, no information could be inferred), relevant (relates indirectly) and central (entity is central topic of document).

Evaluation. The guideline of CCR task requires each returned document should be assigned with a score at range $(0, 1000]$, serving as the confidence to recommend the document as citation of the given target entity's entry article in Wikipedia. The evaluation is conducted by varying a *cutoff* value from 0 to 1000. Precision, recall and F-1 score are calculated at different cutoff levels and the best F-1 score is reported as the main measure. Moreover, the Scale Utility (SU) [11] is also employed as an auxiliary measure which reflects the capability of a information filtering system to accept relevant and reject non-relevant documents. Note that the results of two measures are reported based on two categories of relevance: (1) only central relevant documents are treated as positives (denoted as **C**); (2) both relevant and central documents are treated as positives (denoted as **R+C**). The final performance is reported by averaging F1 and SU scores over all 29 topic entities at *single* optimal cutoff.

Preprocessing. To save the computational cost, we first select a subset of documents with mentions of target entities as working set. The process can be done by exact matching against the target entity name. However, due to the fact that one entity may have multiple surface name variations, it is hard to reach high recall using exact matching only. Balog et al. [4] realized the problem and employed name variants extracted from DBPedia to matching, which was proved to reach high recall (97.4%) with low false positive rate. We therefore use the document list (37,905 documents in training data and 70,411 documents in testing data) released with their work to construct the working set.

4.2 Result Analyses

We choose the top two runs of CCR task as the baselines: hltcoe group by Kjersten et al. [8] and udel_fang group by Liu et al. [9], and denote them as **HLTCOE** and **UDel**, respectively. Moreover, we also choose the best runs reported

Table 1: Performance Comparison

Method	C		R+C	
	F1	SU	F1	SU
HLTCOE [8]	0.359	**0.402**	0.492	0.555
UDel [9]	0.355	0.331	0.597	0.591
MC [4]	0.360	0.263	0.691	0.673
Weight-Uniform	0.356	0.340	**0.709**	**0.704**
Weight-Linear	**0.377**	0.329	0.708	0.700

in the following up work by Balog et al. [4] as a stronger baseline as they have demonstrated their approach reached superior performance over the two previous baselines, and denote it as **MC** (Multi-step Classification). Based on Algorithm 1, we implement two alternatives of entity weighting function (i.e., $weight(e^*, R_{sel}(E))$): (1) non-zero uniform weighting, denoted as **Weight-Uniform**; (2) linear decaying weighting in Equation 2, denoted as **Weight-Linear**. Depending on the relevance category, We use the labeled central documents in training data as $\mathcal{D}_{rel}$ under category **C** and the union of relevant and central documents under category **R+C**, respectively. α is set to 100 and β is set to 50, and the confidence score is estimated by Equation 1 and clamped up to 1000. The performance for all the methods are summarized in Table 1.

We observe that **Weight-Linear** delivers superior performance over all the three baselines under both category **C** and **R+C**, and **Weight-Uniform** outperforms all baselines under category **R+C**, respectively, proving the effectiveness of our approach. Note that **UDel** employs the entity profile based approach as shown in Equation 1, but no weighting of related entities is incorporated. The improvement of our approach over **UDel** validates our hypothesis that not all related entities are same important. Noticeably, all the methods but **Weight-Linear** reach close performance under category **C**, showing that identifying central documents from relevant ones is a challenging task, conforming to the observations by Balog et al. [4]. The obvious advantage of **Weight-Linear** over **Weight-Uniform** under category **C** demonstrates that the linear decaying weighting is more effective than uniform weighting.

By conducting per-topic analyses on the weighted related entities, we have some interesting observations. For topic entity "Basic_Element_(music_group)", "rapping" is the related entity with highest weight, as it is the genre of "Basic_Element_(music_group)", which successfully discriminates it from "Basic_Element_(company)" which shares the same surface name. "Elite_squad" is the related entity with highest weight of "Rodrigo_Pimentel", as the latter co-wrote the film "Elite_squad". The highly weighted related entities show that our algorithm is capable of identifying important related entities from trivial ones.

4.3 Discussions

There are two categories of methods discussed in our experiment setup. One category is supervised classification (i.e., **HLTCOE**, **MC**) in the sense that they employ supervised learning techniques with a set of features over the training data to learn a model and conduct classification on the testing data. The other category is entity profile based filtering (i.e., **UDel**, **Weight-Uniform** and **Weight-Linear**), and the profile of topic entity is first built and then applied to estimate the confidence score of documents. The advantage of the entity profile based filtering methods mainly lies in the term of efficiency, since no model training and feature selection is required, and the profile building can be done offline, while in the online process, they benefit from speed as the computational cost is relatively low comparing to supervised learning method, making them more suitable for processing large volume of data.

5. CONCLUSIONS AND FUTURE WORK

We studied the problem of automatically collecting relevant documents of entities to help accelerate the update process of knowledge base profiles. We proposed a simple yet effective approach to weight related entities to build enriched related entity profiles of given topic entity in an iterative way. We conducted experimental evaluation on the testbed of TREC 2012 KBA track and demonstrated that our approach is capable of delivering superior performance over state-of-the-art methods, while maintaining relatively low computational cost.

There are many directions for our future work. More specifically, we plan to extend our approach by incorporating temporal correlation features of related entity to better capture the dynamics of related entities.

6. REFERENCES

[1] D. Ahn, V. Jijkoun, G. Mishne, K. Muller, M. de Rijke, and S. Schlobach. Using Wikipedia at the TREC QA Track. In *Proceedings of TREC*, 2004.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *The Semantic Web*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer-Verlag, 2007.

[3] K. Balog, P. Serdyukov, and A. P. de Vries. Overview of the TREC 2011 Entity Track. In *Proceedings of TREC*, 2011.

[4] K. Balog, N. Takhirov, H. Ramampiaro, and K. Nørvåg. Multi-step Classification Approaches to Cumulative Citation Recommendation. In *Open research Areas in Information Retrieval (OAIR 2013)*, pages 121–128, 2013.

[5] J. L. Elsas, J. Arguello, J. Callan, and J. G. Carbonell. Retrieval and Feedback Models for Blog Feed Search. In *Proceedings of SIGIR*, 2008.

[6] J. R. Frank, M. Kleiman-Weiner, D. A. Roberts, F. Niu, C. Zhang, C. Ré, and I. Soboroff. Building an Entity-Centric Stream Filtering Test Collection for TREC 2012. In *Proceedings of TREC*, 2012.

[7] H. Ji and R. Grishman. Knowledge Base Population: Successful Approaches and Challenges. In *Proceedings of ACL HLT*, pages 1148–1158, 2011.

[8] B. Kjersten and P. McName. The HLTCOE Approach to the TREC 2012 KBA Track. In *Proceedings of TREC*, 2012.

[9] X. Liu and H. Fang. Entity Profile based Approach in Automatic Knowledge Finding. In *Proceedings of TREC*, 2012.

[10] R. Mihalcea and A. Csomai. Wikify! Linking Documents to Encyclopedic Knowledge. In *Proceedings of CIKM*, pages 233–242, 2007.

[11] S. Robertson and I. Soboroff. The TREC 2002 Filtering Track Report. In *Proceedings of TREC*, 2002.

[12] F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In *Proceedings of World Wide Web Conference*, pages 697–706, 2007.

[13] Y. Xu, G. J. F. Jones, and B. Wang. Query Dependent Pseudo-Relevance Feedback based on Wikipedia. In *Proceedings of SIGIR*, 2009.