Exploiting Wikipedia Inlinks for Linking Entities in Queries

Priya Radhakrishnan, Romil Bansal
Manish Gupta, Vasudeva Varma
International Institute of Information Technology, Hyderabad

ABSTRACT

Given a knowledge base, annotating any text with entities in the knowledge base enhances automated understanding of the text. Entities provide extra contextual information for the automated system to understand and interpret the text better. In the special case when the text is in the form of short text queries, automated understanding can be critical in improving the quality of search results and recommendations. Annotation of queries helps semantic retrieval, ensuring diversity of search results including retrieval of relevant news stories. In this paper, we present SIEL@ERD, a system for automated stamping of entity information in short query text. Our system builds from the state-of-the-art TAGME [4] system and is optimized for time and performance efficiency. Our system achieved an F1 measure of 0.53 and the latency of 0.31 seconds on a dataset of 500 queries and a Freebase snapshot provided for the short track in the Entity Recognition and Disambiguation Challenge [2] at SIGIR 2014.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Storage and Retrieval—*Information Search and Retrieval*; H.4.0 [**Information Systems Applications**]: General

Keywords

Entity Linking, Entity Disambiguation, Semantic Search, Knowledge Base, Wikipedia Inlinks

1. INTRODUCTION

Automatic understanding and knowledge extraction from text documents has become an important task for various applications including web search, topic modeling and text summarization. In this paper, we have explored entity linking over search queries for the Entity Recognition and Disambiguation (ERD) Challenge [2] at SIGIR 2014. Entity linking is the task of identifying entity mentions within the document and linking them to the most relevant entity in the knowledge base. For example, in the query "total recall arnold schwarzenegger", the mention "total recall" should be linked to the 1990 movie and "arnold schwarzenegger" should be linked to the actor/governor. Linking entities in the search queries helps in better understanding the user's intent and thus better retrieval and ranking of search results.

A typical entity linking system has three phases as follows.

- Mention Detection: In this phase, all the possible mentions of any entity in the knowledge base are detected in the text provided. Along with the mentions, the candidate entities for the mentions are also extracted. For example, in the query "total recall movie", the mention "total recall" along with the possible entities "Total Recall (1990 film)" and "Total Recall (2012 film)" are extracted.
- Disambiguation: In this phase, for each mention the best entity out of the candidate entities is detected based on the context provided.
- Pruning: In this phase, mentions having incorrect or meaningless anchors are removed to reduce the noise in text understanding. The remaining mentions along with the disambiguated entities are returned.

In the proposed work, we explore Wikipedia inlinks for linking the entities in short text queries. The paper is organized as follows. In Section 2, we discuss relevant literature on linking the mentions in various forms of text with entities in knowledge bases. In Section 3, we describe our approach for linking entities in short queries. We discuss the experiments performed and results obtained over various runs of the proposed entity linking system in Section 4. Section 5 discusses the various pros and cons of the proposed approach followed by a summary and thoughts on future work in Section 6.

2. RELATED WORK

This section describes various works that have been proposed for linking entities in text documents. Numerous approaches have been proposed for linking entities over long documents [3, 6, 9, 11]. Entity linking over long documents is different from the entity linking in short documents like tweets and queries. Unlike long documents, short documents lack sufficient context to disambiguate the entities completely. Various approaches have also been proposed for linking entities in short text documents [4] and tweets [1, 5, 7, 8, 12].

The proposed work derives motivation from the work proposed by Ferragina et al. [4] for linking entities over short documents. Wikipedia inlinks are explored for detecting and linking the entities present in search queries. The mentions are detected based on the probability that the mention appears as an anchor link on Wikipedia. The disambiguation is done based on the probability of

the mention linking to a particular entity and a Wikipedia-based semantic relatedness measure as proposed by Milne et al. [10]. Pruning is done based on the coherence among the entities detected within the text document. The proposed approach which is optimized for both effectiveness and efficiency is explained in detail in the next section.

3. SIEL@ERD SYSTEM DETAILS

In this section we describe the details of the proposed entity linking system. The system consists of three main components: mention detection, entity disambiguation and anchor pruning. Before discussing these, we discuss a few Wikipedia-based measures which are an essential part of the system components, and also detail few pre-processing steps.

3.1 Wikipedia-based Measures

The following measures are calculated using Wikipedia's hyperlink structure. They are used in detecting, disambiguating and refining the annotations of a text with Wikipedia articles.

- **Wikipedia Link-based Measure** (δ): Wikipedia's extensive network of cross-references provide a huge amount of explicitly defined semantics. *Anchors* are terms or phrases in Wikipedia articles, to which links are attached. Each *link* is a manually-defined connection between anchor and its disambiguated concept. Wikipedia provides millions of these connections. Anchors are used to identify candidate concepts for mentions. Wikipedia's documentation dictates that any term or phrase that relates to a significant topic should be linked to the article that discusses it. Utilizing this article-link-anchor collection, Milne et al. [10] proposed Wikipedia Link-based Measure for obtaining similarity between two pages p_a and p_b as shown in Eq. 1.

$$\delta = \frac{log(max(|in(p_a)|, |in(p_b)|)) - log(|in(p_a) \cap in(p_b)|)}{log(W) - log(min(|in(p_a)|, |in(p_b)|))} \quad (1)$$

  where $in(p_a)$ and $in(p_b)$ are the set of Wikipedia pages pointing to the pages p_a and p_b respectively. W is the total number of pages in Wikipedia.

- **Senses of Anchor** ($Pg(a)$): Wikipedia provides a vast number of anchor texts which capture the various senses they represent. For example, 'plane' links to different articles depending on the context in which it is found, and 'plane', 'airplane' and 'aeroplane' are all used to link to the same article. Because of this polysemy, the same anchor, a may occur in Wikipedia many times pointing to many different pages. We denote this set by $Pg(a)$.

- **Frequency of Occurence** ($freq(a)$): $freq(a)$ denotes the number of times a phrase a occurs in Wikipedia (as an anchor or not).

- **Frequency of Occurence as Link** ($link(a)$): $link(a)$ indicates the number of times the phrase a occurs as an anchor in Wikipedia. Clearly, $link(a)$ is always less than or equal to $freq(a)$.

- **Prior Probability** ($Pr(p|a)$): The prior probability score is the ratio of number of times the anchor links to a particular page p to the total number of times the mention is used as an anchor in Wikipedia. It is the probability that occurrence of a is an anchor pointing to the Wikipedia page p.

- **Link Probability,** $lp(a)$: $lp(a)$ denotes the probability that an occurrence of a phrase a is used as an anchor in Wikipedia. It is the ratio of $link(a)$ to $freq(a)$.

3.2 Data Preprocessing

The English Wikipedia dump was processed to create three indexes to facilitate the calculation of the measures described in the previous subsection.

- **Anchor Dictionary**: Anchors occuring in Wikipedia are indexed to efficiently compute the link frequency $link(a)$, total frequency ($freq(a)$), pages linking to a ($Pg(a)$) along with their prior probability ($Pr(p|a)$). The pages are sorted in the descending order of their prior probabilitites for faster computations.

- **WikiTitlePageId Index**: This index is used for maintaining the mapping between Wikipedia titles and the corresponding PageIds.

- **In-Link Graph Index**: Each Wikipedia page title is indexed to indicate the list of Wikipedia titles linking to this page.

3.3 Mention Detection

A word or group of words that could potentially identify an entity in the knowledge base is called a *mention*. In this module we identify mentions using the link probability, $lp(a)$. For detecting the mentions, we take continuous word sequences of up to 6 words and find if the string appears as an anchor in Wikipedia. If the probability of being an anchor is greater than a predefined threshold, then the anchor is taken as a detected mention and all the pages referred by it are taken as possible candidates for the detected mention.

However this involves a large number of look-ups on the Wikipedia anchor index. In order to reduce the number of look-ups, we used two mention chunking methods as follows.

- **Stopword Chunking**: If the mention identified in the given query text contains only stopwords, we ignore that mention. We use the standard JMLR stopword list [1].

- **Twitter POS Chunking**: The query text is Part-Of-Speech (POS) tagged with a tweet POS tagger [12]. Mentions that do not contain at least one word with POS tag as NN (indicating noun) are ignored.

3.4 Disambiguation

We adapt the disambiguation function from the TAGME [4] system, after making a few changes for efficient computations.

We define the relatedness between two pages p_a and p_b as shown in Eq. 2.

$$rel(p_a, p_b) = 1 - \delta(p_a, p_b) \quad (2)$$

When p_a and p_b are identical pages, $\delta(p_a, p_b)$=0 and hence relatedness score $rel(p_a, p_b)$=1. Otherwise $rel(p_a, p_b)$ is a score between 0 and 1.

The overall vote given to a candidate page p_a of mention a, by a mention b is defined as shown in Eq. 3.

$$vote_b(p_a) = \sum_{p_b \in Pg(b)} rel(p_b, p_a) \times Pr(p_b|b) \quad (3)$$

[1] http://jmlr.org/papers/volume5/lewis04a/a11-smart-stop-list/english.stop

where $Pg(b)$ are all possible candidate pages for the mention b and $Pr(p_b|b)$ is the prior probability of the mention b linking to a page p_b. In the TAGME system, the overall vote is normalized by the total number of senses of b, i.e., $|Pg(b)|$. We experiment with both the normalized and the non-normalized version.

The total relatedness score given to a candidate page p_a for a given mention a is calculated as the sum of votes from all other mentions in the input text T, denoted as A_T.

$$rel_a(p_a) = \sum_{b \in A_T \setminus \{a\}} vote_b(p_a) \quad (4)$$

The overall score given to a candidate entity consists of the relatedness score $rel_a(p_a)$ and the prior probability $Pr(p_a|a)$ of the candidate page p_a. These two factors are combined to get the overall in the following two ways.

1. **Linear Combination**

$$score_a(p_a) = \alpha \times rel_a(p_a) + (1 - \alpha) \times Pr(p_a|a) \quad (5)$$

The value of α was determined experimentally as α = 0.83. The anchor a is then annotated with the Wikipedia page with the highest $score_a(p_a)$.

2. **Threshold Combination** Among all the pages that link to anchor a (i.e., $Pg(a)$) choose the page that has the highest relatedness $rel_a(p_a)$ denoted as $rel_{best}(p_a)$. The set of the other pages in $Pg(a)$, that yield $rel_a(p_a)$ varying less than 25% with respect to $rel_{best}(p_a)$ is determined. From this set, the page p with the highest value of prior probability ($Pr(p|a)$) is chosen as the page for annotating the anchor a.

3.5 Anchor Pruning

The disambiguation phase produces a set of candidate annotations, one per anchor detected in the input text T. This set has to be pruned in order to possibly discard the meaningless annotations. This removal is done using a scoring function similar to the one in TAGME [4].

It uses two features: the link probability $lp(a)$ of the anchor a and the coherence between the candidate annotation of anchor $a \mapsto p_a$ and the candidate annotations of the other anchors in T. Coherence is defined as the average relatedness between the candidate sense p_a and the candidate senses p_b assigned to all other anchors b in T.

$$coherence(a \mapsto p_a) = \frac{\sum_{p_b \in S \setminus \{p_a\}} rel(p_b, p_a)}{|S| - 1} \quad (6)$$

where S is the set of distinct senses assigned to the anchors of the text T.

Pruning score ρ combines the link probability and coherence by a linear combination as shown in Eq. 7.

$$\rho(a \mapsto p_a) = coherence(a \mapsto p_a) + \gamma lp(a) \quad (7)$$

γ was determined empirically. We observed that the best results are obtained when γ is set to 0.1. The mentions with ρ value less than threshold are pruned. Threshold value of ρ was empirically found to be 0.05.

4. EXPERIMENTS

In this section, we will explain in brief the ERD Challenge dataset and analysis of the results of our experiments.

4.1 Dataset

For the ERD Challenge [2], the participant teams were supplied a 9/29/2013 snapshot of Freebase as the knowledge base. This contains only those entities that have English Wikipedia pages associated with them. As the evaluation of the ERD challenge was restricted to these entities, we indexed this dataset and restricted our final results to these entities.

4.2 Results

We submitted seven runs to the ERD Challenge. Each run consists of running our system with 500 search query strings. For each query, the system outputs the entity mentions and the freebaseID (present in ERD dataset) they link to. We explain the seven runs in the chronological order adding/removing the features in that order.
Run 1: As the first run, this was run on a 100 query subset. This was the base system. For mention detection, Twitter POS chunking was used for mention chunking. For disambiguation, vote3 was normalized and linear combination method was used. The anchor dictionary was indexed with mutiple rows per anchor. This probably caused high latency leading to timeouts. The system had an F1 of 0.53.
Run 2: The Run 1 system used link probability instead of the prior probability when computing the overall score for disambiguation. In Run 2, the link probability was replaced by prior probability as shown in Eq. 5. The system had an F1 of 0.50 (on 100 queries).
Run 3: From Run 3 onwards, the runs were evaluated on the full 500 query set. In mention detection, stopword chunking was used for mention chunking. In the disambiguation component, $vote$ (Eq. 3) was not normalized and threshold combination method was used. The anchor dictionary was re-indexed to have single row per anchor. The system had an F1 of 0.483.
Run 4: Run 3 system was enhanced with optimization when making database connections, to re-use the open database connections. In the disambiguation component, we replaced the combination method by the linear combination method. The system had an F1 of 0.472.
Run 5: In the mention detection component, Twitter POS Chunking was used. The system regained the F1 as 0.483.
Run 6: Run 5 system was enhanced with corrections when computing the link probability. Eq. 7 was corrected to use link probability instead of prior probability. The system had an F1 of 0.44.
Run 7: Tested on the final test dataset. In the mention detection component, mention chunking was moved back to stopword mention chunking. The system had an F1 of 0.53.

5. DISCUSSIONS

We believe that the following could be the reasons for the low F1 score for the proposed entity linking system.

- Twitter-POS gave worse performance than merely using the stopword list. A tweet is a more self-contained sentence when compared to a search query. When typing a search query user relies on the search engine to understand the string. On the other hand, when typing a tweet, the user relies on human intelligence to understand. Accordingly he builds in more context in tweets, leading to the more self-contained sentences. Hence it is intuitive that Twitter-POS was expecting a more self-contained sentence, which the search queries were not. Hence the poor performance of Twitter-POS on search strings.

- The entity linking system annotates only those entities that are present in the ERD Challenge dataset. So if the system

detects any entity other than this predefined subset of entities, it is not reported.

6. CONCLUSIONS

In this paper, we presented an entity linking system that accepts a query string and is able to link the relevant entity mentions to corresponding Wikipedia pages. Various syntactic as well as semantic features are utilized for segmenting and linking the entity mentions within the query strings. The system can be leveraged by a large number of applications in the Semantic Web.

7. ACKNOWLEDGEMENTS

We would like to thank Debarshi Dutta, Dilip Vamsi and Vikas Bhandari for providing Wikipedia inlink crawl.

8. REFERENCES

[1] A. E. Cano, G. Rizzo, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Microposts2014 NEEL Challenge: Measuring the Performance of Entity Linking Systems in Social Streams. In *Proc. of the Microposts2014 NEEL Challenge*, 2014.

[2] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, and K. Wang. ERD 2014: Entity Recognition and Disambiguation Challenge. *SIGIR Forum*, 2014 (forthcoming).

[3] S. Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 708–716. Association for Computational Linguistics, Jun 2007.

[4] P. Ferragina and U. Scaiella. TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proc. of the* 19^{th} *ACM Intl. Conf. on Information and Knowledge Management (CIKM)*, pages 1625–1628, 2010.

[5] S. Guo, M.-W. Chang, and E. Kıcıman. To Link or Not to Link? A Study on End-to-End Tweet Entity Linking. In *Proc. of the Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 1020–1030, 2013.

[6] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. In *Proc. of the* 15^{th} *ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 457–466. ACM, 2009.

[7] X. Liu, Y. Li, H. Wu, M. Zhou, F. Wei, and Y. Lu. Entity Linking for Tweets. In *Proc. of the* 51^{th} *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1304–1311, 2013.

[8] E. Meij, W. Weerkamp, and M. de Rijke. Adding Semantics to Microblog Posts. In *Proc. of the* 5^{th} *ACM Intl. Conf. on Web Search and Data Mining (WSDM)*, pages 563–572. ACM, 2012.

[9] R. Mihalcea and A. Csomai. Wikify!: Linking Documents to Encyclopedic Knowledge. In *Proc. of the* 16^{th} *ACM Conf. on Information and Knowledge Management (CIKM)*, pages 233–242. ACM, 2007.

[10] D. Milne and I. H. Witten. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *Proc. of the AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy*, 2008.

[11] D. Milne and I. H. Witten. Learning to Link with Wikipedia. In *Proc. of the* 17^{th} *ACM Conf. on Information and Knowledge Management (CIKM)*, pages 509–518. ACM, 2008.

[12] A. Ritter, S. Clark, Mausam, and O. Etzioni. Named Entity Recognition in Tweets: An Experimental Study. In *Proc. of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011.