UBC Entity Recognition and Disambiguation at ERD 2014

Ander Barrena
University of the Basque Country
Donostia, Basque Country
abarrena014@ikasle.ehu.es

Eneko Agirre
University of the Basque Country
Donostia, Basque Country
e.agirre@ehu.es

Aitor Soroa
University of the Basque Country
Donostia, Basque Country
a.soroa@ehu.es

ABSTRACT

This paper describes the system developed at the University of the Basque Country (UBC) for the Entity Recognition and Disambiguation Challenge (ERD 2014). We developed a single system for both *long* and *short* tracks. We implemented a very basic mention detection component and complement it with a strong disambiguation step, based on Personalized PageRank algorithm. The result and confidence of the disambiguation step is used to decide whether a mention has to be linked, that is, we only link mentions if the disambiguation algorithm is confident enough. This simple method obtained good results in the ERD challenge, reaching the top 10 for both tracks.

Categories and Subject Descriptors

I.7 [**DOCUMENT AND TEXT PROCESSING**]: General

Keywords

ERD 2014, NER, NED, UBC

1. INTRODUCTION

This paper describes the system developed at the University of the Basque Country (UBC) for the Entity Recognition and Disambiguation Challenge (ERD 2014, [4]). Unlike previous competitions (i.e., the TAC-KBP challenges), the ERD 2014 challenge requires participants to correctly identify the document mentions that have to be disambiguated (the target mentions), as they are not provided by the organizers. This way, the ERD challenge includes the mention detection task as an additional step before the entity linking task, and therefore evaluates the systems on a realistic setting.

The entity linking task consists of matching named entity mentions to a reference Knowledge Base (KB). In ERD 2014 the reference KB is a subset of Freebase[1] which comprises entity references exclusively. Common concept entries, like "chair" or "episode" are not included in the reference KB. Besides, all entries are mapped with the corresponding Wikipedia page.

The ERD challenge consists of two separate tracks. The *short* track is focused on short web queries and documents consist of a small number of words. In this track the words are all lowercased and there are no punctuation marks. The *long* track targets web documents and therefore documents in this track are usually much bigger. Besides, the documents contain punctuation marks, acronyms, etc. and words are properly capitalized. As usual, participants are required to link the document named entities to the proper entries of the reference KB. For instance, in the sentence "sesame street episode 3806"[2] the systems are meant to link the named entity "sesame street" to the Freebase node with id `/m/0cwrr`. Note that "episode" or "3806" are not considered named entities, and therefore should not be linked.

Within the ERD challenge each participant has to develop a publicly accessible web service which implements their particular system. Besides, systems have to return their answers within a tight time-frame (20 seconds and 60 seconds per query for the *short* and *long* tracks, respectively).

Our approach is based on a single system composed of 3 steps. First, in the name detection and candidate generation step, in section 3, we search for named entity mentions occurring in documents. In this step, we also generate the possible candidate entities for each mention. Then, in the candidate ranking step in section 4, we disambiguate each mention among candidates, linking to its correct entity. Finally, in the mention filtering step in section 5, we discard linked mentions if the disambiguation algorithm is not confident enough.

The paper is structured as follows. We first present the resources we have used. We then present each of the 3 steps mentioned before. Finally, results and presented and conclusion are drawn.

2. RESOURCES

We use a 2013 Wikipedia snapshot in order to detect and rank possible entity mentions occurring in documents. From

[1]ERD uses a dump from 9/29/2013.

[2]This is an actual example of a document from the *short* track.

ERD'14, July 11, 2014, Gold Coast, Queensland, Australia.
Copyright 2014 ACM 978-1-4503-3023-7/14/07 ...$15.00.
http://dx.doi.org/10.1145/2633211.2634357

the snapshot we extract two information resources: a dictionary and graph with entity relations.

We represent the whole Wikipedia as an undirected graph. Nodes are Wikipedia articles, and edges represent hyperlinks between articles. ERD organizers release the reference KB with Freebase entities, keeping only those entities that have Wikipedia entry associated with them. Using this resource we map graph nodes to corresponding Freebase identifiers. Those nodes having no corresponding Freebase identifier are left untouched. Note that no edge is removed from the graph.

The dictionary is an association between strings and graph nodes. We construct the dictionary using article titles, redirections, disambiguation pages, and anchor text. Mentions are lowercased and all text between parenthesis is removed. If the mention links to a disambiguation page, it is associated with all possible articles the disambiguation page points to. Each association between a string and article is scored with the prior probability, estimated as the number of times that the mention occurs in the anchor text of an article divided by the total number of occurrences of the mention. Note that our dictionary can disambiguate any mention, just returning the article with highest score (most frequent sense baseline, or MFS).

3. MENTION DETECTION AND CANDIDATE GENERATION

In this step we identify all strings appearing as an entry in our dictionary. We scan the document tokens from left to right and consider the longest possible span which has a dictionary entry as a candidate mention. For example, in the example above our system will select both “sesame street” and “episode” as candidate mentions. It would not consider “sesame” (there is already a longer match) nor “3806” (it has no entry in the dictionary).

Mention detection is further complicated in the *long* track, as the system also needs to compute the actual offsets (in bytes) of each detected mention. This step turned to be more complex than expected, mainly due to the encoding and end-of-line issues.

The method described above identify mentions which refer to named entities (“sesame_street”) but also mentions referring to general concepts (“episode”). We further apply a heuristic to identify which mentions refer to named entities, by discarding those mentions that do not contain any uppercase character. This heuristic is only applied on the *long* track, as *short* documents comprise lowercased words only.

Candidate generation is performed by just considering all graph nodes linked to a particular mention in the dictionary.

4. CANDIDATE RANKING

In order to disambiguate the recognized mentions given the candidates, we applied Personalized PageRank (PPR) on the Wikipedia graph using freely available software [2, 1][3].

[3]`http://ixa2.si.ehu.es/ukb`

The PageRank algorithm [3] ranks the vertices in a graph according to their relative structural importance. Alternatively, PageRank can also be viewed as the result of a random walk process, where the final rank of a node represents the probability of a random walk over the graph ending on that node, at a sufficiently large time. Personalized PageRank [7] is a variant of the PageRank algorithm which biases the computation to prefer certain nodes of the graph.

In our setting, the input comprises the target entity mention and its context (the mentions detected in the whole document). We compute the PPR algorithm over the graph initializing it to the set of all articles referred by the mentions words, excluding the articles from the target mention itself. In the initialization step we use the prior probabilities for each mention-article association. The PPR algorithm produces a probability distribution over all Wikipedia articles. We multiply the prior probability of the target mention with the PageRank probabilities before computing the final ranks. Finally, we select the article with highest rank in among the possible articles for the target entity mention. In the rare cases where no known mention is found in the context, we return the node with the highest prior.

Due to the naive name detection of our system, an due to timeout issues, we are sometimes unable to disambiguate all the document mentions. This is particularly important on the *long* track, where documents may contain a large number of mentions. We calculated that our PPR algorithm is able to disambiguate up to 20 mentions within the provided timeframe. We thus select the mentions to be disambiguated using the PPR algorithm, and apply the MFS baseline on the rest.

We sort the candidate mentions in ascending order according to the prior value of its most frequent sense. We disambiguate the first 20 mentions using the PPR algorithm, and assign the MFS baseline to the rest. The motivation behind this step is that mentions whose sense distribution is very skewed towards the most frequent sense are in principle easier to disambiguate. Thus, we reserve the more complex PPR algorithm to the difficult cases only. Note that in the *short* track there are less than 20 mentions per document, so we were able to disambiguate all of them using PPR.

As a result from the disambiguation step, we obtain a graph node for each candidate mention. This graph node may refer to a Freebase entity or to a general Wikipedia article. In the latter case we produce no output for the mention, that is, we discard a mention whenever the disambiguator links it to an non-entity node. Note that this way we use the disambiguation output as a method of filtering out mentions which do not refer to named entities.

5. FURTHER FILTERING MENTIONS

The last paragraph describes our general method to discard mentions which are not named entities. During development we saw that this method alone was not good enough to decide when to discard a mention. When analyzing the results, we found out many cases where the mention is an obvious concept but the disambiguator links it to a KB entity. We thus decided to further filter out mentions according to the confidence value given by the disambiguator: when this confidence value is lower than a given threshold, we discard the mention. As said before in section 4, our systems can return an entity disambiguated using the PPR method or the MFS baseline, and we apply different thresholds depending on the method used for the disambiguation.

We tried different thresholds during development, seeking to maximize the F1 measure. Those are the thresholds performing best in the development dataset on each track:

- Best performance in *short* track: discarding all mention with confidence lower than 0.8 both for PPR and MFS.
- Best performance in *long* track: discarding all mention with confidence lower than 0.8 for PPR and 0.6 for MFS.

In general setting a threshold yields to better precision as a cost of a small loss of recall, obtaining better performance for f1 score. As a consequence, we can conclude that our system without thresholds generates too many mentions, and this filtering step helps significantly improves the results.

6. EXPERIMENTAL RESULTS

Table 1 shows the results obtained during development. The *short* track results are close to the best system reported, obtaining a F1 score of 61.9. In the *long* track we obtained a result of 68.6 F1, ranking 10th. The latency of our systems is slightly higher than the average, due to the time elapsed by the PPR algorithm.

Run	F score	Latency	Rank
UBC Short	61.9	4.50	7/19
Best Short	66.4	1.03	1/19
UBC Long	68.6	38.93	10/17
Best Long	81.1	1.55	1/17

Table 1: Results for *short* and *long* track during development. Rank gives the position obtained among all participants, reaching the top 10 in both tracks.

The final results for ERD 2014 are shown on table 2. Our system ranked 6th in the *short* track, scoring 63.7 F1, 5 points below the winning team. Leadership is very tight since the top teams scored really close to each other. In the *long* track we ranked 10th, the same position we had in the development phase, 13 points below the best system. All in all, our system ranked in top 10 for both tracks.

Run	F score	Latency	Rank
UBC Short	63.7	5.01	6/19
Best Short	68.6	2.21	1/19
UBC Long	63.2	38.29	10/17
Best Long	76.0	1.49	1/17

Table 2: Results for short and long track during test.

7. CONCLUSIONS AND FUTURE WORK

We present the UBC system for the Entity Recognition and Disambiguation challenge, which meets all the requirements established by the organizers. We used an unified system for both tracks, based on a simple but effective method based on 3 steps. Overall we are pleased with the results. Moreover, this challenge pushed us to build a web service that will be very useful for our research group. It was great and fun to develop system while we could see the progress of the other groups. And much more in the near end of the contest, when the competitiveness was increasing. We think that this kind of challenges push the researchers to participate with more enthusiasm.

We have seen that our system needs to improve the mention detection, so in the future we will put all our efforts in this step. We did not discard the possibility of introducing a statistical based algorithm like [6] or [5] in order to speed up the disambiguation step.

Acknowledgements

The research leading to these results was carried out as part of the READERS project (`http://nlp.uned.es/readers-project/`) funded by European CommunityâĂŹs in the framework of ERA-NET CHIST-ERA. The work has been also funded by the Spanish Government (project SKaTer, TIN2012-38584-C06-02). Ander Barrena enjoys an scholarship from the Basque Government.

8. REFERENCES

[1] E. Agirre, O. L. de Lacalle, and A. Soroa. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 40(1):57–88, 2014.

[2] E. Agirre and A. Soroa. Personalizing PageRank for Word Sense Disambiguation. In *Proceedings of 14th Conference of the European Chapter of the Association for Computational Linguistics*, Athens, Greece, 2009.

[3] S. Brin and L. Page. 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, 1998. Elsevier Science Publishers B. V.

[4] 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).

[5] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems*, page 121âĂŞ124. ACM, 2013.

[6] X. Han and L. Sun. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, page 945âĂŞ954. Association for Computational Linguistics, 2011.

[7] T. Haveliwala. Topic-sensitive PageRank. In *Proceedings of the 11th international conference on World Wide Web (WWW'02)*, pages 517–526, New York, NY, USA, 2002.