Leveraging Entities in Background Document Retrieval for News Articles

Kuang Lu[a] and Hui Fang[a]

[a]University of Delaware, Newark DE 19716, USA

Abstract

In this year's News Track, we focus on effectively estimating entity weights by using the words surrounding the entities for background linking task. Analyses on the results reveal the inconsistency of our methods as well as the temporal characteristics of the queries. These observations lead to several interesting research questions about the background document retrieval task.

1 Introduction

The News track is designed with the goal of identifying the search needs of readers and journalists [1]. Two potential needs are identified. The first need is, for a news article, to link a list of other articles that a user should read next in order to get the context and background information of the article. This would help readers better understand the story and evaluate the trustworthiness it. The second need is identified as ranking the entities in a news piece based on whether the external information, such as Wikipedia pages, would help readers better understand the piece.

Based on these needs, two tasks are introduced for the news track, which are background linking and entity ranking. In this year's News Track, we tackle the first task and focus on leveraging entities in the query article to identify background articles. We first define entities as the things and concepts that have their own Wikipedia page, which is similar to that of Daiber et al. [2]. We call these entities *Wiki entities*. Wiki entities are more specific than non-entity words. Moreover, the existence of their Wikipedia pages may suggest that the appearances of Wiki entities are more informative and can be used as strong signals when retrieving background documents. We implemented a method that focuses on correctly estimating the entity weights based on the *context* (e.g. the words surrounding the entities) of the entities. Although our proposed method does not perform as well as a simple baseline that uses all terms in the query document in this year's data, our method outperforms the baseline significantly in last year's data. Moreover, further analyses reveal some interesting insights such as the effect of a time filter. Based on them, several research questions are

[1] trec-news.org

proposed. These research questions are not investigated fully in this paper due to time limit, but we believe that they are worth answering and could lead to better understanding of the problem of background article linking.

2 Data Processing

The collection is the same as last year's Washington Post news collection. We followed the document cleaning process of the top team of last year [1] to remove unwanted documents. More specifically, files were processed according to the lexical order of the file names. When a document in a file was being processed, a 3-tuple (document title, author name, published date) was extracted and saved. If the tuple was the same as that of a previously processed document, the current document was considered a duplicate and was discarded. Otherwise, its document type in "kicker" the field was checked. If it was "Opinion", "Letters to the Editor", or "The PostVíew", it was discarded as well since, according to the guideline, the document is considered to be none-background. If a document was not removed by the previous filtering steps, its id, title, timestamp, and the text of each paragraph as well as title were kept.

In order to leverage Wiki entities, they have to be identified in the documents. We followed our approach last year [3] and used DBpedia Spotlight [2]. DBpedia[2] is a knowledge base that contains structured information about Wiki entities. There is a unique DBpedia entity corresponding to a Wiki entity. DBpedia Spotlight can automatically annotate DBpedia entities from text which accomplishes our entity annotation goal. The entities were subsequently replaced by their canonical forms (e.g. the form appears in DBpedia), which are also provided by the toolkit. For example, "the Red Planet" and "Mars" were all mapped to the canonical form "Mars". We hope that this would help us to more accurately match the entities in the query articles and other articles. The parameter "Confidence" of the tool was set to 0.5. We treat the identified entities, either single-term or multi-term, as single words and used the Indri [4] toolkit to index the documents.

3 Methods

As mentioned before, we focused on using entities to identify background documents. More specifically, we tried to estimate the weights of entities more accurately. If a simple baseline that uses all words in the query document, such as one of the top runs of last year [1], the weights of the words are decided by their occurrences in the document. However, we hypothesize that the importance of an entity is also related to the **context** that it is used in. Given an article, it usually reports an event or discusses a topic. Besides that, some minor aspects of the event or topic are also mentioned in the document. If an entity is often used for describing the event or discussing the main topic, the entity might be more important compared to those for describing minor topics. In order to measure such differences, given an entity, we combine the words before and after each occurrence of the entity (e.g. context

[2]https://wiki.dbpedia.org/

words), generate a language model based on the words, and compute the KL-divergence between this context language model and the document language model. Low divergence might indicate that the entity is frequently used to discuss the core event or topic. On the contrary, if the divergence is high, the entity tends to be about the minor topics. Based on this, we generate a weight distribution of words and entities $P(D)$ for a query document D from its entity weight distribution $P(E)$ and word weight distribution $P(W)$:

$$P(D) = \lambda P(W) + (1 - \lambda)P(E) \tag{1}$$

in which λ is a regulator that balances the weights between the word weight model and the entity weight model. $P(E)$ is generated by computing the entity weights based on their context words whereas $P(W)$ is generated by using the word occurrences. Both of them are normalized so that the weights within each model sum up to 1. $P(D)$ then can be used to retrieve background documents for D.

4 Run Results and Analysis

We submitted two runs. First run UDInfolab_ent is created following the method described in the previous section. We picked the context window size to be 10. Moreover, we tuned the λ in Equation 1 from 0.1 to 1.0 with a step size of 0.1 on last year's data and chose the one with the best performance, which is 0.7. A time-based filtered was employed so that only the documents with a publication date earlier than that of the query documents can be retrieved. The filter was included since it improves the effectiveness significantly based on our experiments on last year's data. The effect of the filter will be investigated and discussed later. Besides this run, we also implemented a baseline run UDInfolab_all in which we simply use all words and entities in the query document for retrieval.

The effectiveness of the submitted runs as well as the TREC median of this year are reported in Table 1 as NDCG@5. Additionally, performances of the methods on last year's data are included. When comparing the baseline UDInfolab_all with the TREC medians, it is clear that even the simple method is reasonably effective for the task and is better than the TREC medians statistically significantly with p value less than 0.05 according to paired t-test in both years. However, although the performance of UDInfolab_ent is higher than the baseline in 2018, it is worse in 2019 and the difference is statistically significant with a p value lower than 0.05. This is a very interesting phenomenon. In order to further investigate this, we tune λ on 2019's queries, which shows that 0.7 is also the optimal value on 2019. It suggests two things. First, our method is stable in that the optimal parameter setting is the same on both years. However it seems to be not a robust solution compared to the baseline since it improves on some queries (e.g those in 2018) but fails on others (e.g. those in 2019). More detailed analysis is required to discover what are those queries, and how our method can be altered for those queries. We did not conduct the analysis due to time limit but we plan to investigate the problem further in the future.

Another line of analyses we did is about the effect of the time filter. We implemented a method called UDInfolab_nf (nf stands for no filter) which removes the time filter of

Table 1: NDCG@5 for our methods and TREC medians on Both Years

Method	2018	2019
UDInfolab_ent	0.467	0.575
UDInfolab_all	0.438	0.606
TREC Median	0.345	0.530

UDInfolab_all. It is applied to both years' data and we compare its performances against that of UDInfolab_all in Figure 1. As can be see, queries in 2019 seem to be easier than that in 2018 as the performances of both methods are considerably better in 2019. However, the performance discrepancy between the methods is significant in 2018 (with a p value lower than 0.05), but is virtually nonexistent in 2019. The usage of the time filter seems to guarantee a better performance nonetheless. To better understand the impact of the time filter, a box plot of the percentages of the differences of top 5 documents between the methods of the same queries are shown in Figure 2. Unsurprisingly, we observe that, in 2019, the simple baseline method tends to retrieve the same set of documents regardless of the existence of the time filter (and the set of documents were published before the time of the query document). As can be seen, there are no differences among 25% of the queries and 75% of the queries have less than or equal to 40% (or 2) documents that are different in the top 5 documents. However, the time filter has a significant impact on what documents are retrieved in 2018. As can be seen, half of the queries observe at least 40% differences and 25% of the queries have at least 60% of (or 3) documents changed in the top 5. To attain more insights about this phenomenon, from the judgement files we compute the percentages of background documents that were published before the query documents in the two years, and show a box plot of them in Figure 3. As can be seen, in general the relevant documents are more likely to be published before the query document. For all queries in the two years, only 5 out of 116 queries[3] have more than half of their relevant documents published after the query documents. We believe this fits the general intuition of background information. Moreover, it explains why methods with the time filter is better or at least the same. Nevertheless, the distribution is slightly more skewed in 2019 than in 2018. In 2019, the bottom whisker is marginally higher than that in 2018. More than 75% of the queries in 2019 have at least 80% of their background documents published before query documents. While in 2018, although it is more top-heavy as the median is around 90%, the third quartile is about 75%, which is lower than that of 2019. Moreover, around 25% of the queries have a difference percentages in the top 5 document between 75% and 50%. According to our conversation with the organizers, this seem to be due to the fact that in 2019, they tried to choose queries that were later in the corpus epoch.

5 Conclusion

In this year's News Track, we investigated how to leverage entities in retrieving background articles. Although our methods can not consistently outperform our baseline, the usefulness

[3]We include the 59 queries in 2018 that have at least one background document, and the 57 queries in 2019 that have judgements available

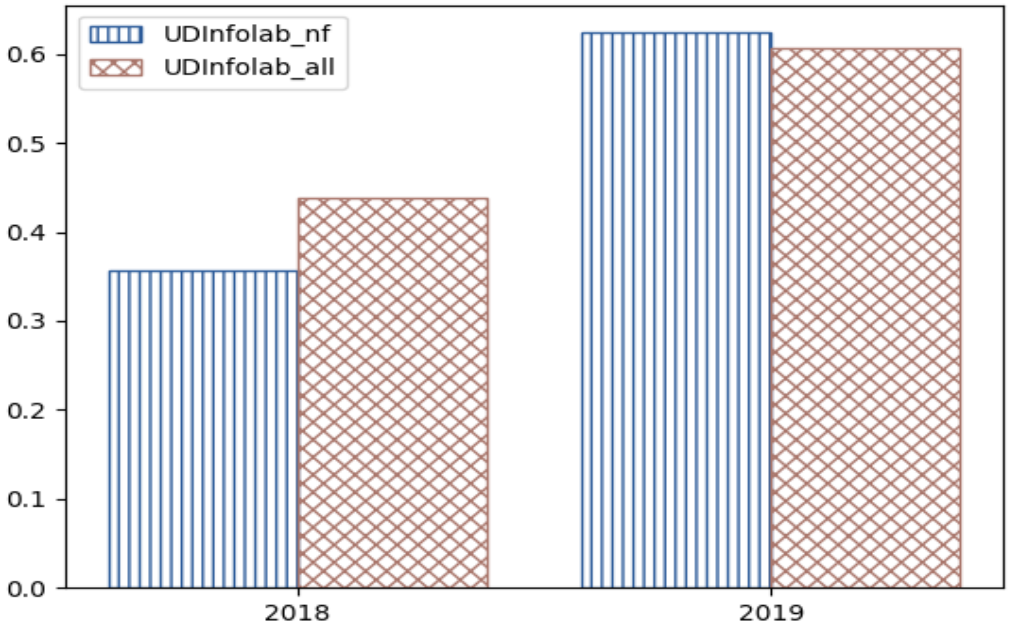

Figure 1: The Effect of The Time Filter

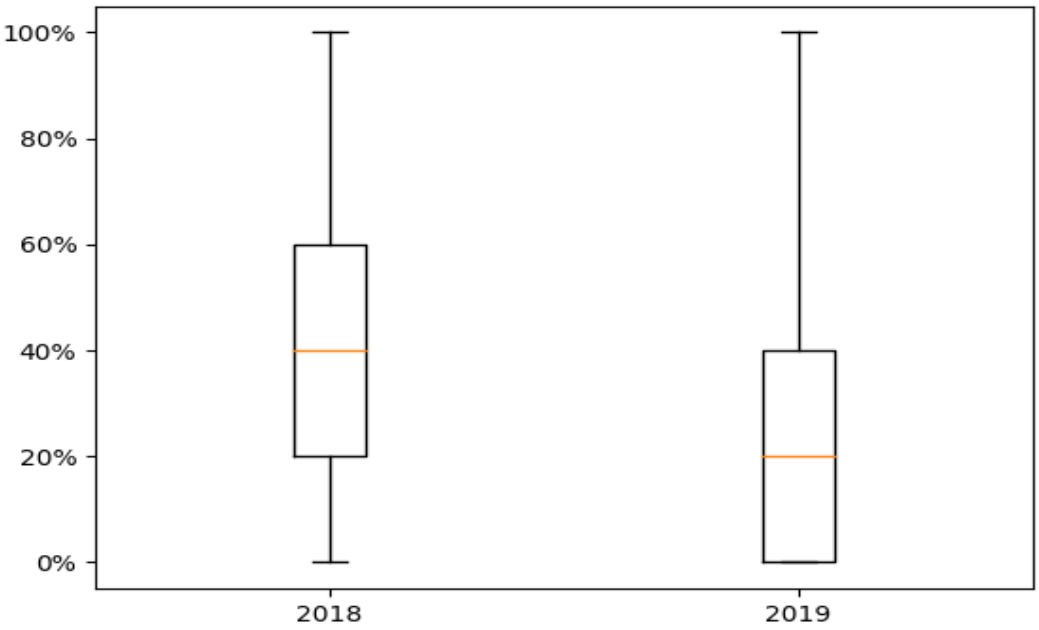

Figure 2: Percentages of Different Top-5 Documents between $UDInfolab_{nf}$ and $UDInfolab_{all}$ Results

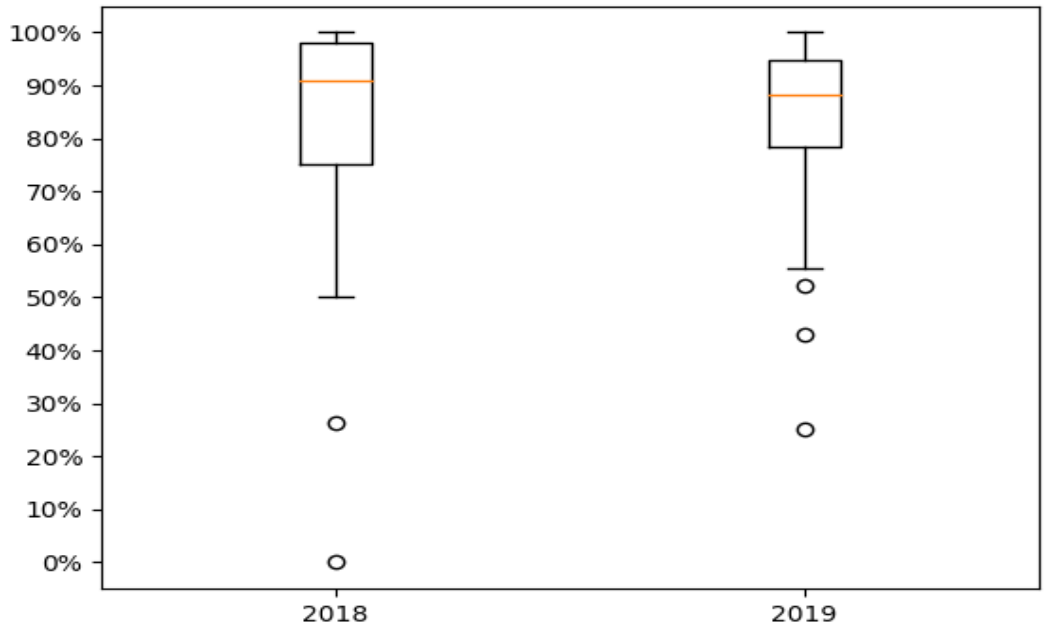

Figure 3: Percentages of Background Documents Published before Query Documents

of it is exhibited in 2018's data. Moreover, time filter seems to be useful in that it would benefit queries published earlier in the collection whereas would not hurt the queries published latter. We plan to more thoroughly investigate the temporal characteristics of queries in the future.

References

[1] Bimantara, A., Blau, M., Engelhardt, K., Gerwert, J., Gottschalk, T., Lukosz, P., Piri, S., Shaft, N.S., Berberich, K.: htw saar @ TREC 2018 news track. In: Proceedings of the Twenty-Seventh Text REtrieval Conference (2018)

[2] Daiber, J., Jakob, M., Hokamp, C., Mendes, P.N.: Improving efficiency and accuracy in multilingual entity extraction. In: Proceedings of the Ninth International Conference on Semantic Systems (2013)

[3] Lu, K., Fang, H.: Paragraph as lead - finding background documents for news articles. In: Proceedings of the Twenty-Seventh Text REtrieval Conference (2018)

[4] Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language model-based search engine for complex queries. In: Proceedings of the 2005 International Conference on Intelligent Analysis (2005)