SU-NLP at TREC NEWS 2021

Kenan Fayoumi and Reyyan Yeniterzi*

Sabancı University, İstanbul, Turkey

Abstract

This paper presents our work and submissions for the TREC 2021 News Track Wikification task. We approach the problem as an entity linking task initially and after identifying the mentions and their corresponding Wikipedia entities, we rank the mentions within the news article based on their usefulness. For the entity linking part, transformer-based architectures are explored for both detecting the mentions, generating the possible candidates and re-ranking them. Finally for the mention ranking, we use previous years' best performing approach which uses the position of the mention within the text.

1 Introduction

Knowledge Bases (KBs) such as Wikipedia, are comprised of a huge collection of entries/articles related to entities, concepts and artifacts. Entity related tasks such as wikification (entity linking), entity disambiguation, entity representation and others are gaining a lot of attention in the literature as these tasks are useful building blocks for more complex NLP tasks such as question answering, information extraction and text summarization. KBs provide a rich source of knowledge that can be incorporated to these tasks. Furthermore, in the context of news articles, automatically linking mentions of entities to KBs can provide useful background and contextual information to readers for a better reading experience.

The terms wikification and entity linking are used interchangeably over the literature. Given a text document and a knowledge base which contains a set of predefined entities, the task is to detect mentions of entities within the text and link them to their corresponding entry in the KB. In the wikification

*Corresponding author: *reyyan.yeniterzi@sabanciuniv.edu*

case, Wikipedia is used as the knowledge base, and mentions of entities within the text are linked to the corresponding Wikipedia pages. In the TREC News Wikification task, the task is not only about linking entities but also ranking them based on their relevance or importance to the news article.

This is the second time wikification is included as a task at the TREC News track. This paper describes SU-NLP's efforts on this year's wikification task. This year, we specifically focus on the impact of utilizing different components for entity linking sub-tasks and analyze the effectiveness of these methods on wikification task.

2 Dataset

TREC 2021 organizers have provided an updated version of the Washington Post news corpus (Version 4) containing 728,626 articles and blog posts dating from January 2012 up until December 2020. Previous version of the Washington Post corpus which was used in previous year's tasks, contains 671,947 articles. Each news article contains a number of content paragraphs. These could consist of text, images or videos. The news articles contain HTML tags and these tags are kept in place to assure the original mention placements/locations while evaluating the system.

This is the second year wikification task is offered at TREC News. Last year's wikification topics consisted of 50 news articles. As for this year's task, 51 news articles/topics are provided. Since the relevance judgments for 2020 topics were provided, we utilize these for evaluating our work. More statistics concerning the topics are displayed in Table 1. According to Table 1, topics from this year and last year are very similar in terms of length. As the knowledge base, previous year's Wikipedia dump of Jan 2020 is also used this year.

	2020	2021
# topics	50	51
Avg. # paragraphs	27	29
Avg. # words in paragraph	43	41
Max. # words in paragraph	174	196

Table 1: Statistics of 2020 and 2021 TREC News Wikification Topics

3 Methodology

We approach the TREC News Wikification task as a two steps procedure. The first step is the entity linking part and the second step is the mention (and corresponding entity) ranking part based on its usefulness for the news article.

In the literature, the task of entity linking is mostly split into a pipeline of 3 sub-tasks: Mention Detection/Recognition (MD), Candidate Generation, Candidate Re-ranking or Entity Disambiguation (ED). In MD, the task is to find spans of texts that mention a specific entity. Detected mentions are given to Candidate Generation step where for each mention a list of candidate entities from the knowledge base are returned. The last step is to re-rank the list of candidate entity-mention pairs in terms of similarity (Entity Disambiguation). This pipeline is depicted in Figure 1 for our case when Wikipedia is used as the knowledge base and Washington Post articles are used as the input text documents.

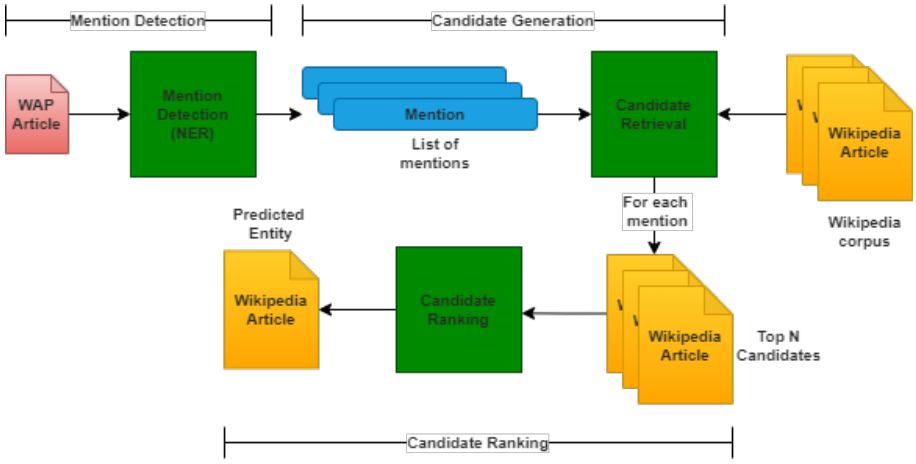

Figure 1: Wikification Entity Linking Pipeline

Last year's best performing model (Ak et al., 2020), used a simple entity linking model. They used Stanford CoreNLP's NER tool (Manning et al., 2014) to identify the mentions and then searched over an Elastic Search index built over the Wikipedia articles' titles. The first Wikipedia page returned from the search engine was used to represent the entity. Finally, ranking the different entities within the same article based on their order in the news article returned the highest scores.

This year we extend on our work from previous year (Ak et al., 2020)

by employing more recent neural network-based architectures in the entity linking part of the task. Namely, we experiment with utilizing two different mention detection models, two different candidate retrieval models and a transformer-based model for candidate ranking.

3.1 Mention Detection

In mention detection step, we treat the task as a named-entity recognition task. For that purpose as our first approach we follow the work in last year's approach (Ak et al., 2020) and utilize Stanford CoreNLP's NER tool (Manning et al., 2014). As for our second NER model, we use Flair NLP tool (Akbik et al., 2019) which utilizes transformer architectures in the NER modeling. These two tools are used to extract the entity mentions which are passed to our candidate retrieval system as the next step.

3.2 Candidate Retrieval

Two different approaches are explored for the candidate retrieval step. Our first method is inspired by Ak et al. (2020) as we utilize an Elasticsearch search engine built on top of the Wikipedia. We follow Ak et al. (2020) in both the index and query formulation as we query the extracted entity mentions over an index built by only using the title field of the Wikipedia articles.

As of our second method, we follow the Bi-Encoder approach of Wu et al. (2020) for dense entity retrieval. In their approach candidate retrieval is handled with vector similarity search. Mentions and entities are represented as vectors using a BERT encoder. To encode the Wikipedia entities, the following input is generated with the addition of the special token $[ENTITY]$ to separate the Wikipedia title and description from each other:

$$[\boldsymbol{CLS}]\ title\ [\boldsymbol{ENTITY}]\ description\ [\boldsymbol{SEP}]$$

Similarly, mentions are encoded with BERT using the following input structure which uses both the left and the right context of the mention:

$$[\boldsymbol{CLS}]\ left\ context\ [\boldsymbol{M_{start}}]\ mention\ text\ [\boldsymbol{M_{end}}]\ right\ context\ [\boldsymbol{SEP}]$$

In both of these models, $[CLS]$ token representation is extracted and used for representing the Wikipedia entities in the first one and mentions in the second one. Later on these vectors are passed into a single layer neural network that performs dot-product to determine the similarity between each

mention and candidate entity pairs. The parameters are optimized by maximizing the dot-product similarity of golden mention-entity pairs. In this Bi-Encoder model, all inputs are cropped or padded into 256 tokens. For Wikipedia the first 256 tokens are used and for mention text, the included left and right context sizes are dependant on the mention length and are limited into 128 tokens.

A dataset of all linked mentions in the Wikipedia dump was used for training the encoder. After training, all entity vectors are calculated and used for building a FAISS (Johnson et al., 2017) vector search index. At inference time, each mention is first encoded and then the query is sent to FAISS to retrieve the nearest N neighbors in terms of vector similarity.

For both of our candidate retrieval methods, namely (1) ElasticSearch and (2) Bi-Encoder, we obtain the top 100 candidates and pass them to final stage in the entity linking pipeline: candidate re-ranking.

3.3 Candidate Re-ranking

After obtaining the top 100 candidate entities for each detected mention, the next step is to re-rank these candidates and select the most likely prediction for each mention. This step is an optional step as the ranking scores (if available) from the previous retrieval step can be used as well.

Furthermore we explore the Cross-Encoder approach of Wu et al. (2020) for additional re-ranking. Cross-Encoder is similar to Bi-Encoder but this time both the Wikipedia entity and the mention are encoded together in the BERT architecture. They are concatenated into an input as shown below:

$$[\boldsymbol{CLS}]\ \textit{left context}\ [\boldsymbol{M_{start}}]\ \textit{mention text}\ [\boldsymbol{M_{end}}]\ \textit{right context}\ [\boldsymbol{SEP}]\ \textit{title}\ [\boldsymbol{ENTITY}]\ \textit{description}\ [\boldsymbol{SEP}]$$

This method is used to further apply deep attention between the mention context and entity text. Following this representation, the embedding of $[CLS]$ token is obtained and sent to a linear layer to produce a final ranking score for the mention-entity pair. Similar to Bi-Encoder, the parameters are optimized by maximizing the scores of golden mention-entity pairs. Unlike Bi-Encoder, this approach does not use the FAISS, instead a score is inferred for a mention and its context together with a candidate Wikipedia entry. For each mention the highest scoring entity out of the 100 retrieved candidates is selected.

3.4 Mention Ranking

After obtaining a list of mentions and their linked entities in each news article, final step of the TREC Wikification task is to rank these mentions (and entities) in terms of their importance and relevance to the article. For this purpose we use previous years' top performing method Text Order in this step. The position of the mentions is used for the ranking. Based on previous years' experiments the position of the entities seem to be good indication of their relevance to the news article.

As for our second method, we utilize the Cross-Encoder model scores for ranking the linked mentions in a document. With this approach we like to see if modeling both the Wikipedia article and news article together can be useful for estimating the relevant entities of a news article as well.

4 Experiments and Discussion

In order to analyze the effects of different components of the proposed pipeline, 2020 topics are used in the pre-submission experiments.

4.1 Mention Detection

For mention detection, in our experiments with Stanford Core-NLP and FLAIR, FLAIR consistently outperforms ($\approx$ 10% boost in final wikification score) Core-NLP in experiments with 2020 topics. Furthermore, we investigate the effectiveness of these tools by calculating recall for the mention detection subtask using 2020 topics which contain 422 gold mentions. As expected, FLAIR with a score of 78% out-performs CoreNLP's 64% score. As a results of these results FLAIR is used for detecting mentions in the submitted runs. It should be noted that both of these tools perform much worse precision-wise with a high rate of false positives mentions detected.

4.2 Candidate Retrieval

As for candidate retrieval, we report the recall at 1, 10, 50 and 100 candidates for our two retrieval methods on 2020 topics in Table 2. Elastic-Search performs slightly better overall for our retrieval purpose compared to Bi-Encoder. For both methods, we observe a 1-2% boost in recall when retrieving 100 candidates compared to 10 candidates, therefore 100 candidates are retrieved for the submissions.

Model	R@1	R@10	R@50	R@100
Bi-Encoder	0.67	0.73	0.73	0.74
Elastic-Search	0.72	0.75	0.76	0.77

Table 2: Recall Scores of Retrieval Models on 2020 Topics

4.3 Candidate Re-Ranking

Candidate re-ranking step is an optional step. Normally both Elastic-Search and Bi-Encoder approaches used in the previous retrieval step return a ranked list of results with relevance scores. This ranking can be used directly for linking the entities. With 2020 topics using the Bi-Encoder retrieval list directly without any other re-ranking returned 0.3211 NDCG@5 score. Similarly, Elastic-Search returned 0.3400, which is slightly higher.

This year we also experiment with Cross-Encoder as an additional re-ranking mechanism. Re-ranking the retrieved 100 entities with this method provided significant improvements in both Bi-Encoder and Elastic-Search experiments, NDCG@5 score of 0.4244 and 0.4380 respectively. Modeling the Wikipedia entry and the news article together seems to be useful for identifying the more relevant entities. In order to see whether this holds for 2021 topics as well, all these four settings are submitted as part of the official runs.

4.4 Mention Ranking

For mention ranking, we continue to use the text order (position of the mention) for most of the runs. As an addition experiment we also use Cross-Encoder model's score directly for mention ranking, as we would like to analyze the correlation between Cross-Encoder's score and mention's relevance to the article.

4.5 TREC 2021 Submissions

This year five runs are submitted, and all of our submissions utilize FLAIR for the mention detection. FLAIR model detected a total of 2830 mentions in 1198 paragraph across 51 news article.

The details of the five submissions are as following:

- SU-BiEnc: Utilizes Bi-Encoder for retrieval, also uses Bi-Encoder score for candidate ranking and finally text-order for mention ranking.

System	NDCG@5
SU-BiEnc	0.6422
SU-ES	0.6483
SU-BiEnc-CrsEnc	0.7542
SU-ES-CrsEnc	0.7482
SU-ES-CrsEnc-NF	0.7476

Table 3: NDCG@5 Scores on 2021 Topics

- SU-BiEnc-CrsEnc: Utilizes Bi-Encoder for retrieval, Cross-Encoder for candidate re-ranking and text-order for mention ranking.
- SU-ES-CrsEnc: Utilizes Elastic-Search for retrieval, Cross-Encoder for candidate re-ranking and text-order for mention ranking.
- SU-ES: Utilizes Elastic-Search for retrieval, Elastic-Search scores are used for candidate ranking and text-order is used for mention ranking.
- SU-ES-CrsEnc-NF: Utilizes Elastic-Search for retrieval and Cross-Encoder for both candidate and mention ranking. NF stands for no text order ranking for mentions.

The NDCG@5 scores of our submitted runs are presented in Table 3. We observe that 2021 results are mostly consistent with our experiments on 2020 topics. Using Bi-Encoder or Elastic-Search directly without any candidate re-ranking returned very similar performances. When the Cross-Encoder is introduced as an additional re-ranking mechanism, a significant improvement (around 0.10-0.11) is observed again. Using Bi-Encoder and Cross-Encoder together slightly outperformed the Elastic-Search and Cross-Encoder combination this time.

As specified by TREC submission rules, only 100 detected entities can be submitted per news article. For 4 of our 5 submissions, text order (appearance in text) is used to create the top 100 ranked mentions. As our last submission (SU-ES-CrsEnc-NF) we use the Cross-Encoder scores to create the final ranking for submission. This one also performed similar to the text order ranking approach.

Even though the high scores (around 0.75), we believe there is one pitfall in our architecture. Utilizing separate models in the wikification pipeline causes errors to propagate from one step to the other. Our first step, mention detection model detects a large amount of candidate mention spans with a low precision score. This calls for a method to filter out or reject unlikely

mention spans instead of trying to link them to entities. This will be analyzed as a future work.

5 Conclusion

In this paper, we present our work for TREC 2021 News Track Wikification task. This year, we explore different methods for the different subtasks in the pipeline. In addition to trying successful methods from previous years' wikification experiments, more recent neural network-based entity linking approaches are also explored. Based on our experiments, we observe high effectiveness of neural-based architecture, especially in mention detection and candidate ranking subtasks. Furthermore, based on our results, search engines are still competent options for high-recall candidate retrieval systems.

References

A. E. Ak, Çağhan Köksal, K. Fayoumi, and R. Yeniterzi. Su-nlp at trec news 2020. In *TREC*, 2020.

A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf. Flair: An easy-to-use framework for state-of-the-art nlp. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, 2019.

J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The stanford corenlp natural language processing toolkit. 01 2014. doi: 10.3115/v1/P14-5010.

L. Wu, F. Petroni, M. Josifoski, S. Riedel, and L. Zettlemoyer. Zero-shot entity linking with dense entity retrieval. In *EMNLP*, 2020.