The CLaC System at the TREC 2020 News Track

Pavel Khloponin and Leila Kosseim

ClaC Lab
Department of Computer Science and Software Engineering
Gina Cody School of Engineering and Computer Science
Concordia University
Montreal QC, Canada
p_khlopo@encs.concordia.ca, leila.kosseim@concordia.ca

Abstract. This paper describes our approach to the TREC 2020 News Track. The goal of the News Track is to provide background links and entity linking to target news articles within a collection of articles. We submitted 4 runs, 2 of which achieved scores higher than the median. The first approach is based on the classic Okapi BM25 using the entire article as a query. This run obtained an nDCG@5 of 0.5924. The second approach is based on a combination of BM25 with GPT-2 embeddings which lead to an nDCG@5 of 0.5873. These top models were chosen after exploring a variety of representations of documents as embedding vectors and proximity measures. Combining document embeddings and Okapi BM25 is shown to diversify the results without much impact on quality, and can help to overcome the limitations of a single model system implementation.

Keywords: News TREC · Text embedding · Proximity measures.

1 Introduction

Given the sheer number of electronic sources of news available today, it is important to develop approaches for the automatic recommendation of contextual information for users to better understand a news article.

In order to address this need, since 2018, the News Track at TREC has proposed two related shared tasks: background linking and entity ranking (Soboroff, Huang, and Harman 2018, Soboroff, Huang, and Harman 2019, Soboroff, Huang, and Harman 2020). The goal of the background linking task is to provide relevant background information to news articles through the identification of related articles. On the other hand, entity ranking focuses on providing a list of names, concepts, artifacts, etc. mentioned in news articles, which will help readers better understand the news. This year, we only participated to the first task: background linking.

For the 2020 background linking task, the data set provided by NIST is an extension of the collection used in the previous years and consists of about

671,000 news articles from the Washington Post. Given a search topic, which itself is an article from the corpus selected by the TREC organizers, participants need to select up to 100 related articles from the corpus and output them as a ranked list from the most related to the least related. For evaluation purposes, only the top 5 articles from each list are considered. A 5 point rank is manually assigned by NIST assessors for each of the top 5 articles during the evaluation phase. The score assigned to each search topic is an integer between 0 (little or no useful information) and 4 (must appear in recommendations or critical context will be missed). The total score of the system is then computed using the nDCG@5 metric (Järvelin and Kekäläinen 2002) with the gain metric 2^{rank}.

One important criterion for judging related articles is diversity. Because we did not have a clear understanding of the notion of article diversity in the news recommendation context, we did not specifically address this aspect of the task, but we discuss this issue in Section 5.

For the submission, 50 search topics were used; however, for the official evaluation, only 49 search topics were considered. One topic with no relevant backlinks was dropped. Prior to the evaluation, the organizers also provided search topics and their corresponding manually evaluated results (that is, all articles evaluated manually with a rank from 0 to 4) from the TREC News 2018 (50 topics) and TREC News 2018 (60 topics), which used the same article collection. We used these two sets of topics (110 in total) and evaluated backlinks (24,163 in total) for validation purposes.

2 Document Collection

The TREC News 2020 organizers provided a document collection of 671,934 news articles from The Washington Post published between 2012 and 2019. This document collection is the same as the one provided at the 2018 and 2019 editions of the task but with duplicate articles removed and with new articles from 2017 to 2019 added. Each news article is stored in "JSON-lines" format and represented as a single line of JSON. Each document contains 8 types of meta-information:

1. id
2. article URL
3. title
4. author
5. publication date
6. type (blog post or article)
7. news source
8. content field

2.1 Preprocessing

Apart from the id (#1) field, which was used for identification purposes, only the article title (#3) and the text extracted from the content field (#8) were

considered. The title was prepended to the content and processed as a single document. The content itself was stored in a form of content blocks, where each content block can be a text paragraph, an image, a video, a tweet, a citation, etc. Each content block itself may contain meta-information (up to 133 different fields), such as MIME-type, type, kicker (category), content, subtype, source, URLs, etc. Based on this meta-information we identified blocks with frequently appearing content types and checked if they have any useful text descriptions, and kept for further processing only blocks with paragraphs, image captions, headers, and quotes. Blocks were further cleaned from embeds, links, images, and other HTML tags, preserving only plain text.

2.2 Statistics

NIST required participants to ignore wire articles, editorial content and opinion posts, which have "Opinion", "Letters to the Editor", or "The Post's View" values in the content meta-information block with type "kicker". Due to this, the initial set of 671,947 articles was reduced by 2,057 to 669,890 items. This is shown in Table 1.

As shown in Table 1, many sandbox articles (content previews or articles demonstrating website functionality) were discovered during data exploration: 1,112 articles with a URL path starting with "/test/wp/", presumably indicating content taken from the section of the website not intended to be public (content playground), and 95 documents with "Lorem ipsum..." (common placeholder text) content in the article text. They were preserved in the dataset.

original number of articles	671,947
articles to ignore	2,057
articles used to build the models	669,890
"lorem ipsum" articles	95
sandbox articles	1,112
average size of article before pre-processing (characters)	10,391
average size after pre-processing (characters)	4,533

Table 1. Statistics of the 2020 TREC News document collection

As shown in Table 1, the 671,947 documents considered have an average length of 10,391 characters prior to preprocessing, but only 4,533 after pre-processing. Figure 1 shows the distribution of the article lengths before and after pre-processing. Some articles are composed almost entirely of meta-information and have very little text inside (short statements, video players, cited tweets, etc.), while others have very long texts (transcripts of debates, conferences, testimony, crime reports). As Figure 1 shows, the majority have between 1,000 and 10,000 characters.

3 Our approach

3.1 Document Representation

After pre-processing, we experimented with six different representations to represent each document: Doc2Vec distributed representation (Le and Mikolov 2014), GPT (A. Radford and Narasimhan 2018), GPT-2 (Alec Radford et al. 2019), XLNet (Yang et al. 2020), BERT (Devlin et al. 2019) and RoBERTa (Liu et al. 2019).

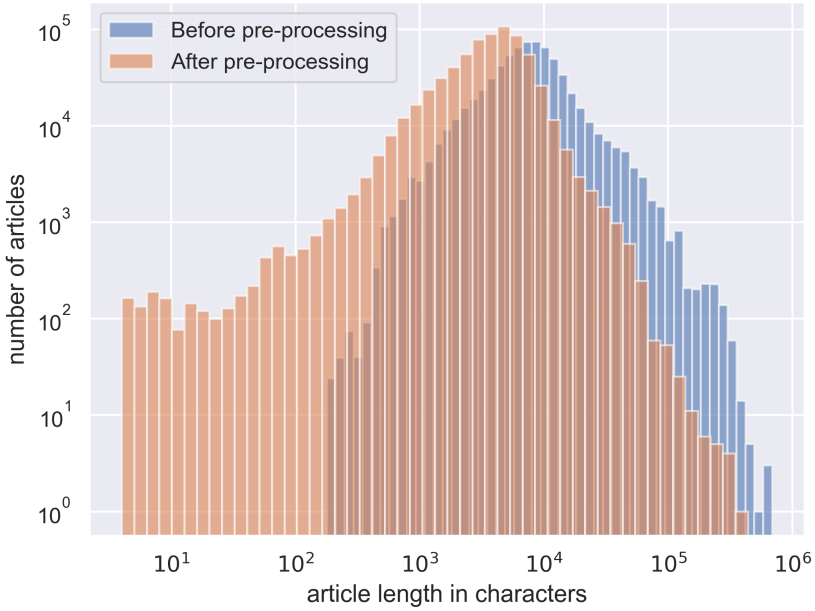

Fig. 1. Length distribution of the articles in the collection

The rationale for using such document representations was our expectation that related articles would be closer to each other in vector space than unrelated articles. Hence, document vector distance would be a good approximation of document content relatedness. Search topics and manually ranked documents from 2018 and 2019 (see Section 1) were used as a validation set to select among different combinations of parameters for text processing and model parameters. Before creating the document vectors, meta-information from the text was removed and Unicode characters were normalized. Pre-trained models were used off the shelf without fine-tuning. Taking into account the deep convolution nature of the models we evaluated embeddings pulled from the last layer as well

as from the hidden layers. Max, min and mean pulling were calculated for each embedding. We also checked if normalizing vector components would make a difference. In total, we experimented with 195 types of embeddings per document.

3.2 Proximity Measures

After obtaining the embeddings for the articles, the proximity between two document vectors is computed. We decided to explore a broad range of proximity measures. Following (Cha 2007) we experimented with 65 different measures. Taking into account different parameters in some of these measures, in total we experimented with 174 proximity measures.

The main measures include cosine, Jacard and Minkowsky L_p. Given two document vectors α and β of size n, distances from Table 2 between α and β are computed as follows:

- cosine distance (inverse of cosine similarity)

$$\frac{||\alpha||||\beta||}{\alpha \cdot \beta} = \frac{\sqrt{\sum_{i=1}^{n} \alpha_i^2}\sqrt{\sum_{i=1}^{n} \beta_i^2}}{\sum_{i=1}^{n} \alpha_i \beta_i}$$

- Jaccard distance (inverse of Jaccard simmilarity for dense vecrors)

$$\frac{\sum_{i=1}^{n} (\alpha_i - \beta_i)^2}{\sum_{i=1}^{n} \alpha_i^2 + \sum_{i=1}^{n} \beta_i^2 - \sum_{i=1}^{n} \alpha_i \beta_i}$$

- Minkowsky L_p distance

$$\sqrt[p]{\sum_{i=1}^{n} |\alpha_i - \beta_i|^p}$$

Using different embedding models, pulling methods, output layers and proximity measures gave us a total of 2,677 runs to evaluate. To speed up the experiments of proximity measures were implemented directly in Elasticsearch.

3.3 Validation

In order to select the best performing combination of parameters, the 2,677 models were evaluated with the 2018 and 2019 data. To evaluate our models, we generated the top 5 backlinks for each topic and computed nDCG@5 when compared with the 2018 and 2019 datasets. If a generated backlink was not in the validation sets, we assigned it a rank of 0 (equivalent to an irrelevant backlink) to calculate nDCG@5.

As shown in Table 2, the model with GPT-2 embedding and the Minkovwsky L_3 distance achieved an average nDCG@5 of 0.466, which is higher than GPT-2 without normalization. Doc2Vec on other hand performed better without normalization with an average nDCG@5 of 0.449, compared to an average nDCG@5 of 0.238.

The top performing embedding algorithm and proximity measure were selected to build the final models. Table 3 displays their performance on the 2019 data.

Embedding	Distance	nDCG@5
GPT-2 with normalization	Minkowsky L_3	0.466
GPT-2	Minkowsky $L_{0.15}$	0.457
Doc2Vec	Jaccard	0.449
GPT	Jaccard	0.440
Doc2Vec	Cosine	0.428
XLNet large cased	Minkowsky $L_{0.1}$	0.354
XLNet base cased	Minkowsky $L_{0.5}$	0.337
RoBERTa base	Minkowsky $L_{0.5}$	0.323
Doc2Vec with normalization	Jaccard	0.238

Table 2. Top performing models on the 2019 data

3.4 Choice of Runs

Given our results with the validation set, the top run was chosen to be submitted to this year's shared task, in addition to the classic Okapi BM25, our model from last year, and a combined approach. In the end, the following four runs were submitted: `clac_gpt2_norm`, `clac_es_bm25`, `clac_combined`, and `clac_d2v2019`.

1. **clac_gpt2_norm** is based on GPT-2 (normalized) embeddings. Each article's text was split in chunks 512 words, embeddings were computed and averaged for each article. Minkowski L_3 norm was used for proximity measure. This configuration was the best one among the embedding methods and proximity measures we explored (see Table 2).
2. **clac_es_bm25** is based on the Elasticsearch (Lucene) implementation of the Okapi BM25 ranking algorithm. The entire article text was used as a query. The parameter `indices.query.bool.max_clause_count` value was increased to 10240 to allow a longer query article. This approach was selected as it is widely used in the text retrieval field, and its implementation in Elasticsearch already has a reasonable configuration.
3. **clac_combined** is a combination of two previous runs, where the BM25 score between the query and each target document is multiplied by the inverted distance between corresponding GPT-2-embeddings.
4. **clac_d2v2019** is based on Dov2Vec embeddings computed from News TREC 2019 and cosine similarity as a proximity measure. This model was used because we used this approach last year in our participation to the track, and, for this year, we wished to compare last year's method to novel ones.

4 Results and Analysis

4.1 Official Scores

As indicated in Section 3.4, we have submitted four runs. For each topic, NIST provided us with our official score as well as the minimum, maximum and median scores across all submitted runs. Table 4 shows the overall scores for all topics. As

Run	nDCG@5
clac_es_bm25	0.5689
clac_combined	0.5668
clac_gpt2_norm	0.4660
clac_d2v2019	0.4280
TREC max	0.7737
TREC median	0.5295
TREC min	0.1002

Table 3. Results with the 2019 data

shown in Table 4, two of our runs outperformed the collective median nDCG@5 of 0.5250. Among our submissions `clac_es_bm25` achieved the highest score with an nDCG@5 of 0.5924, `clac_combined` performed slightly worse reaching an nDCG@5 of 0.5873, while `clac_gpt2_norm` and `clac_d2v2019` performed below the collective median with nDCG@5 of 0.4541 and 0.4481 respectively. These results are inline with our expectations from our validation runs on the 2019 data (see Section 3.3).

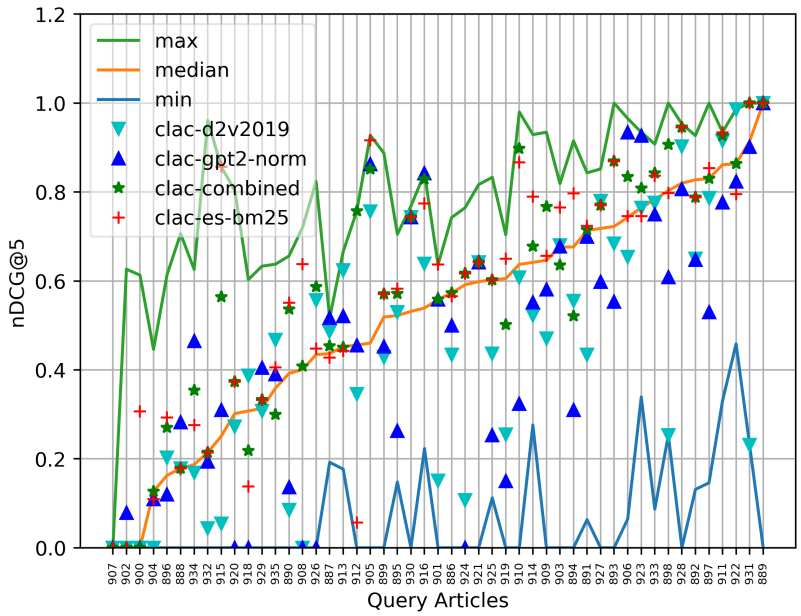

Fig. 2. 2020 results per topic in increasing order of median nDCG@5

Run	nDCG@5
clac_es_bm25	**0.5924**
clac_combined	0.5873
clac_gpt2_norm	0.4541
clac_d2v2019	0.4481
TREC max	0.7914
TREC median	0.5250
TREC min	0.0660

Table 4. 2020 overall results of our runs

Figure 2 shows the scores for each topic, in decreasing order of the median score. Difficult topics for all participants appear on the left-hand side of Figure 2, while easier topics are on the right-hand side. For a few topics our approach had the lowest performance among all runs (7 topics for `clac_d2v2019` and `clac_gpt2_norm`, 3 topics for `clac_combined`, and only 2 topics for `clac_es_bm25`), they are all situated on the minimum line. On the other hand, for a few other topics, all points are situated on the maximum line (4 topics for `clac_d2v2019`, 6 for `clac_gpt2_norm` and for `clac_combined`, and 8 topics for `clac_es_bm25`) hence we achieved the best performance among all News TREC runs on these topics.

Table 5 shows over all topics the percentage of times that each run was above or equal to the median.

Run	$\geq$ median	$<$ median	percentage $\geq$ median
clac-combined	42	7	85.71%
clac-es-bm25	39	10	79.59%
clac-d2v2019	22	27	44.90%
clac-gpt2-norm	20	29	40.81%

Table 5. The number of topics ranked $\geq$ or $<$ the overall median for each run

Figure 3 shows the scores per topic in increasing order of nDCG@5. This figure helps to compare our runs with the collective median and show that `clac_es_bm25` and `clac_combined`, in general, perform better than `clac_gpt2_norm` and `clac_d2v2019`.

4.2 Analysis

As shown in Table 4, `clac_es_bm25` and `clac_combined` are rather similar in terms of overall median nDCG@5, however when looking at individual topics, their results are significantly different. Figure 4 shows their difference graphically. The model `clac_combined` returned the best result over all runs submitted for the topics #912[1] when `clac_es_bm25` missed the most relevant article[2]. The

[1] How to ditch the boring trail mix and eat well while camping out
[2] Move beyond skewers and s'mores on your next camping trip

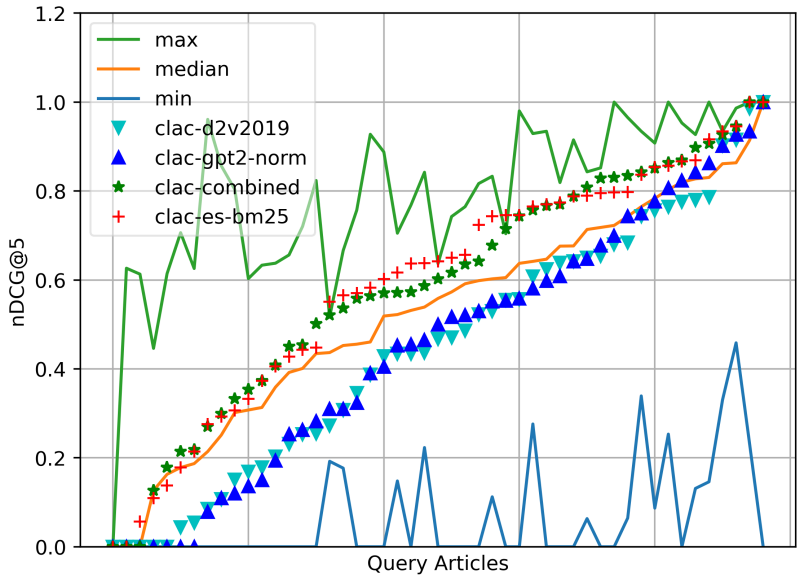

Fig. 3. Results sorted individually by their nDCG@5. The topics do not necessarily correspond to the data points with the same abscissa.

query article and the most relevant backlink have less word overlapping compared to query article and the top backlink returned by `clac_es_bm25`. `clac_combined`, on other hand, functioning on a different principle, was able to return the most relevant backlink in the first position.

All our models failed to return any relevant backlinks for topic #907. Only one relevant link was returned on topic #902 by the `clac_gpt2_norm` model. Topics #900, #904, #896 and #888 were also challenging for our systems and for other participants as well.

5 Conclusions and Future Work

Despite being a classic approach to information retrieval, Okapi BM25, returned good results on the backlinking task, but there is still an opportunity for improvement, since it is not providing the ideal order for backlinks and is missing some articles entirely.

On the other hand, our approach of using document embeddings based on pre-trained models such as BERT, GPT, GPT-2 or XLNet with various proximity measures is showing potential, even though by itself these performed below the median. In combination with BM25 these showed very similar nDCG@5 but

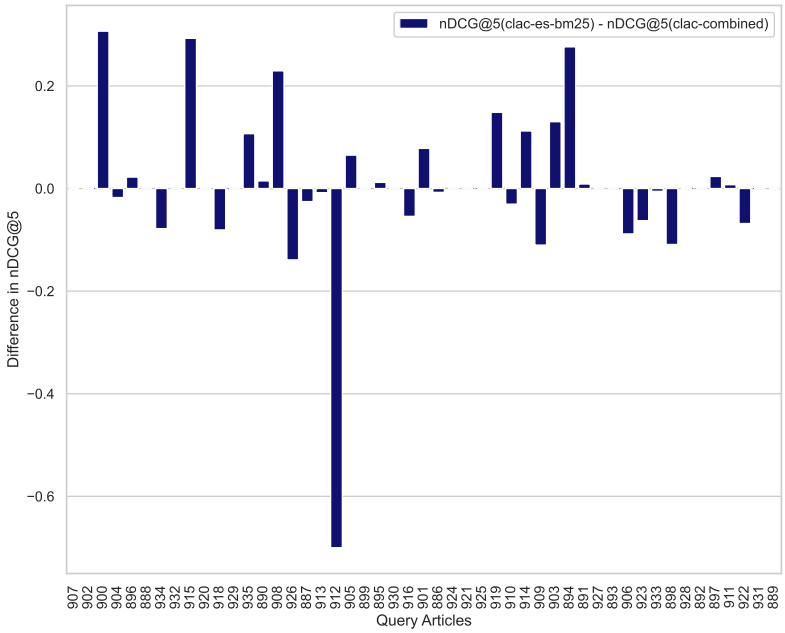

Fig. 4. The difference in nDCG@5 scores for `clac_es_bm25` and `clac_combined`

on closer look we can see significant variety on results from `clac_es_bm25` and `clac_combined`. This fact shows a potential step towards returning more diverse backlinks, by relying on different models with different implementations we can return topics potentially not visible to a single model system.

In this work we did not do any fine-tuning of the models we used. Tuning for our specific dataset and task itself might improve the representation of the documents in vector-space, potentially providing a better ranking for backlinks.

Acknowledgements

This work was financially supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Soboroff, Ian, Shudong Huang, and Donna Harman (2018). "2018 News Track Overview". In: *NIST Special Publication 500-331: The Twenty-Seventh Text*

REtrieval Conference Proceedings (TREC 2018). URL: https://trec.nist.gov/pubs/trec27/papers/Overview-News.pdf.

— (2019). "2019 News Track Overview". In: *NIST Special Publication 1250: The Twenty-Eighth Text REtrieval Conference Proceedings (TREC 2019)*. URL: https://trec.nist.gov/pubs/trec28/papers/OVERVIEW.N.pdf.

— (2020). "2020 News Track Overview (Notebook version)". In: *The Twenty-Ninth Text REtrieval Conference (TREC 2020) Notebook*. URL: https://trec.nist.gov/act_part/conference/papers/OVERVIEW.N.PDF.

Järvelin, Kalervo and Jaana Kekäläinen (Oct. 2002). "Cumulated Gain-based Evaluation of IR Techniques". In: *ACM Transactions on Information Systems (TOIS)* 20.4, pp. 422–446. ISSN: 1046-8188.

Le, Quoc and Tomas Mikolov (22–24 Jun 2014). "Distributed Representations of Sentences and Documents". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, pp. 1188–1196. URL: http://proceedings.mlr.press/v32/le14.html.

Radford, A. and Karthik Narasimhan (2018). "Improving Language Understanding by Generative Pre-Training". In: *Preprint*. URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

Radford, Alec et al. (2019). "Language Models are Unsupervised Multitask Learners". In: *Preprint*. URL: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Yang, Zhilin et al. (2020). *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. arXiv: 1906.08237 [cs.CL].

Devlin, Jacob et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv: 1810.04805 [cs.CL].

Liu, Yinhan et al. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. arXiv: 1907.11692 [cs.CL].

Cha, Sung-Hyuk (Jan. 2007). "Comprehensive Survey on Distance/Similarity Measures Between Probability Density Functions". In: *Int. J. Math. Model. Meth. Appl. Sci.* 1.