BERT, ELMo, USE and InferSent Sentence Encoders: The Panacea for Research-Paper Recommendation?

Hebatallah A. Mohamed Hassan
Roma Tre University, Department of Engineering, Rome, Italy
hebatallah.mohamed@uniroma3.it

Giuseppe Sansonetti
Roma Tre University, Department of Engineering, Rome, Italy
gsansone@dia.uniroma3.it

Fabio Gasparetti
Roma Tre University, Department of Engineering, Rome, Italy
gaspare@dia.uniroma3.it

Alessandro Micarelli
Roma Tre University, Department of Engineering, Rome, Italy
micarel@dia.uniroma3.it

Joeran Beel
Trinity College Dublin, School of Computer Science and Statistics, ADAPT Centre, Dublin, Ireland
beelj@tcd.ie

ABSTRACT

Content-based approaches to research paper recommendation are important when user feedback is sparse or not available. The task of content-based matching is challenging, mainly due to the problem of determining the semantic similarity of texts. Nowadays, there exist many sentence embedding models that learn deep semantic representations by being trained on huge corpora, aiming to provide transfer learning to a wide variety of natural language processing tasks. In this work, we present a comparative evaluation among five well-known pre-trained sentence encoders deployed in the pipeline of title-based research paper recommendation. The experimented encoders are USE, BERT,

ACM RecSys 2019 Late-breaking Results, 16th-20th September 2019, Copenhagen, Denmark

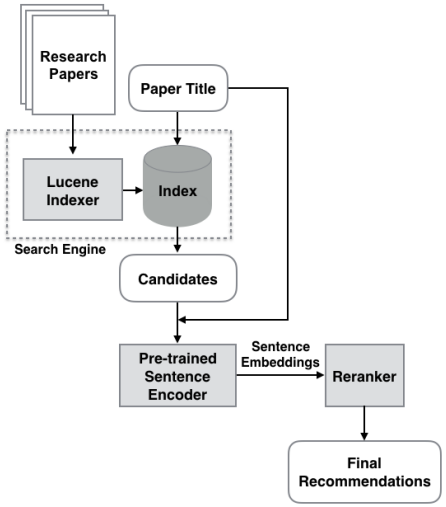

Figure 1: Architecture of the proposed approach.

InferSent, ELMo, and SciBERT. For our study, we propose a methodology for evaluating such models in reranking BM25-based recommendations. The experimental results show that the sole consideration of semantic information from these encoders does not lead to improved recommendation performance over the traditional BM25 technique, while their integration enables the retrieval of a set of relevant papers that may not be retrieved by the BM25 ranking function.

KEYWORDS

Pre-trained sentence embeddings, semantic similarity, reranking, research paper recommendation

INTRODUCTION

Sentence encoders such as Google's BERT and USE, Facebook's InferSent, and AllenAI's SciBERT and ELMo, have received significant attention in recent years. These pre-trained machine learning models can encode a sentence into deep contextualized embeddings. They have been reported to outperform previous state-of-the-art approaches such as traditional word embeddings, for many natural language processing tasks [2, 3, 5, 7, 8]. Such tasks also include the calculation of semantic similarity and relatedness, which is key in developing effective research-paper recommender systems. If sentence encoders performed for calculating relatedness of research articles as good as for other tasks, this would mean a great advancement for research-paper recommender systems.

There are some work on using document embeddings for ranking research papers based on semantic relatedness (e.g., [4]), but - as far as we know - no work on exploiting pre-trained sentence embeddings for the same task. Existing work on sentence encoders focused on different domains such as social media posts [10], news [6, 10], and web pages [6]. Our goal is to find how well some of the most common sentence encoders perform for calculating document relatedness in the scenario of a research-paper recommender system. To the best of our knowledge, we are the first to conduct such an evaluation in the field of research paper recommendations.

METHODOLOGY

We focus on the task of related-article recommendations, where a recommender system receives one paper as input, and returns a list of related papers. In our experiments, research papers are represented by their title only. While this may not be ideal to leverage the full potential of sentence encoders, using only the title is a realistic scenario as many research-paper recommender systems do not use full-texts but only the title (and sometimes the abstract) [1].

Sentence Embeddings, Baseline, and Hybrid Approaches

We experiment with five pre-trained sentence encoders to transform the input paper and candidates papers in the corpus into sentence embeddings. The encoders are as follows:

- USE. It has two models available to download from Tensorflow Hub: the former trained with a Deep Averaging Network (DAN)[1], the latter with a Transformer[2] network. The models are trained on a variety of web sources, such as Wikipedia, web news, web question-answer pages, and supervised data from the Stanford Natural Language Inference (SNLI) corpus. Both models return vectors of 512 dimension as output [3].
- InferSent. It adopts a bi-directional Long-Short Term Memory (LSTM) with a max-pooling operator as sentence encoder, and it is trained on the SNLI corpus. We experimented with two models of InferSent[3]: the former trained using Glove word embeddings, the latter using fastText word embeddings. The output is an embedding of 4096 dimension [5].
- ELMo. It uses a bi-directional LSTM to compute contextualized character-based word representations. We used TensorFlow Hub implementation of ELMo[4], trained on the 1 Billion Word Benchmark. It returns a representation of 1024 dimension [8].
- BERT. It is a sentence embedding model that learns vector representations by training a deep bi-directional Transformer network. We used the uncased BERT-Base model[5] trained on Wikipedia, through bert-as-service[6] to obtain a vector of 768 dimension [7].
- SciBERT. It is a BERT model, but trained on a corpus of 1.14M scientific papers [2]. We used the recommended uncased scibert-scivocab version[7]. Similar to BERT, we obtain a vector of 768 dimension using bert-as-service[6] .

As strong baseline we used BM25, a common approach for document ranking. Today's sentence encoders have a major drawback, that is, the high execution time on large corpora. Using sentence embeddings on large corpora seems hardly feasible in a production recommender system, which needs to return recommendations within a few seconds or less. Hence, we do not only compare the embeddings with BM25, but additionally experimented with hybrid approaches in which we first used Apache Lucene's BM25 to retrieve a list of top-20, 50 or 100 recommendation candidates, and re-ranked that list with the sentence encoders. Figure 1 shows the overall architecture of the proposed approach. In the re-ranking step, we calculate the cosine similarity between the sentence embedding of the input paper title and embeddings of all the candidate paper titles. This similarity metric expresses the semantic similarity. We perform a linear combination between the initial scores from BM25 after being normalized, and the semantic similarity scores from sentence embeddings, by summing up the scores (with uniform weights set to 0.5) to generate the final ranked recommendations.

Dataset

For the evaluation, we used the CiteULike dataset [9]. It contains paper collections of 5,551 researchers, i.e. lists of which documents researchers added to their personal document collection. Table 1 shows the statistics of the dataset.

Table 1: Statistics of the CiteULike dataset.

#Users	#Papers	User-Paper
5,551	16,980	204,986

[1]https://tfhub.dev/google/universal-sentence-encoder/2

[2]https://tfhub.dev/google/universal-sentence-encoder-large/3

[3]https://github.com/facebookresearch/InferSent

[4]https://tfhub.dev/google/elmo/2

[5] https://github.com/google-research/bert

[6]https://github.com/hanxiao/bert-as-service

[7]https://github.com/allenai/scibert

Evaluation

Using 5-fold cross-validation, we split the data by randomly selecting one of the research paper titles that a user has in her library as input paper, and all the remaining paper titles are used for evaluating if the recommended papers were actually in the user's library. As evaluation metrics, we calculated Recall, Precision, and Mean Average Precision (MAP) at rank 10. We also measured the execution time. Our tests were performed on a computer with Intel Core i7-6700 CPU and 16GB RAM.

RESULTS AND DISCUSSION

In Table 2, we report the performance of: (i) BM25 without any reranking as baseline; (ii) Only sentence embeddings; (iii) BM25 scores are combined with sentence embeddings similarity score for reranking. The results show that none of the sentence embedding models alone is able to outperform BM25. Furthermore, we observe that USE, BERT and SciBERT outperform ELMo and InferSent, on average. One possible reason is that USE, BERT, and SciBERT are trained on corpora that contain technical and scientific terms (i.e., USE and BERT on Wikipedia, SciBERT on scientific papers), whereas ELMo and InferSent are trained on a news crawl and a natural language inference corpus, respectively.

On the other hand, the hybrid sentence embeddings + BM25 ranking outperforms all other single approaches. Apparently, in some cases, BM25 fails to assign the right ranking scores to papers, while sentence embeddings could capture the semantic similarity between them. In this case, the ranking performance increases with the top-N number of papers retrieved by BM25, which means that more relative papers can be found. BM25 + USE (Transformer) performs best. Compared to BM25, it relatively increases MAP@10 by +5.29% when reranking 20 titles, by +6.47% when reranking 50, and by +7.35% when reranking 100 titles.

In our experiments, BM25 queries took around 5 milliseconds to retrieve up to 100 results. The extra time taken to calculate embeddings and reranking 20, 50 and 100 titles through the different models is shown in Figure 2. USE (DAN) is the fastest, taking around 0.02 seconds to rerank 20 or 50 titles, and 0.03 seconds to rerank 100 titles. ELMo is the slowest in reranking 20 or 50 titles. Finally, BERT and SciBERT using bert-as-server are the slowest in reranking 100 titles, taking around 4.0 seconds. This means that they could not be used for real-time reranking recommendations, unless higher computing resources (e.g., GPU or TPU) were provided.

In conclusion, our results show that the sentence encoders – that perform so well in other domains – are not performing well in the domain of research paper recommendations. When combined in a hybrid approach with BM25, they perform better than BM25 alone or any of the encoders alone. However, the improvement of up to 7.35% is still small compared to the exceptional results sentence encoders achieve in other domains. A limitation of our research is that we only worked with the titles of papers. Presumably the encoders work better with longer text. In future research, we will use at

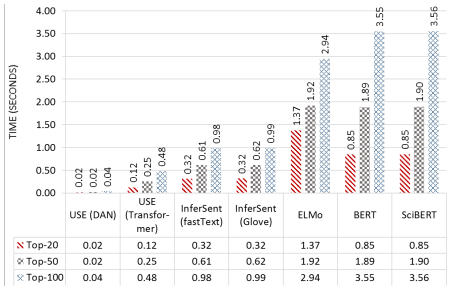

Figure 2: Reranking time in seconds.

Table 2: Recall, Precision and MAP. Results are statistically significant (with $p < 0.05$ in ANOVA test).

Methods	Top-20 R@10	Top-20 P@10	Top-20 MAP@10	Top-50 R@10	Top-50 P@10	Top-50 MAP@10	Top-100 R@10	Top-100 P@10	Top-100 MAP@10
BM25 (no reranking)	0.0197	0.0540	0.0340	0.0197	0.0540	0.0340	0.0197	0.0540	0.0340
USE (DAN)	0.0194	0.0515	0.0313	0.0181	0.0480	0.0286	0.0171	0.0449	0.0267
USE (Transformer)	0.0199	0.0532	0.0328	0.0191	0.0504	0.0310	0.0183	0.0483	0.0294
InferSent (Glove)	0.0188	0.0510	0.0307	0.0174	0.0477	0.0289	0.0165	0.0457	0.0273
InferSent (FastText)	0.0188	0.0522	0.0311	0.0172	0.0495	0.0286	0.0162	0.0473	0.0273
ELMo	0.0179	0.0480	0.0276	0.0151	0.0395	0.0231	0.0134	0.0352	0.0206
BERT	0.0196	0.0528	0.0319	0.0186	0.0507	0.0295	0.0175	0.0463	0.0272
SciBERT	0.0196	0.0534	0.0317	0.0183	0.0495	0.0295	0.0177	0.0474	0.0282
BM25 + USE (DAN)	0.0205	0.0561	0.0353	0.0207	0.0566	0.0356	0.0208	0.0568	0.0357
BM25 + USE (Transformer)	**0.0208**	**0.0567**	**0.0358**	**0.0211**	**0.0574**	**0.0362**	**0.0213**	**0.0580**	**0.0365**
BM25 + InferSent (Glove)	0.0200	0.0551	0.0349	0.0205	0.0563	0.0353	0.0205	0.0564	0.0353
BM25 + InferSent (FastText)	0.0201	0.0551	0.0346	0.0201	0.0553	0.0347	0.0201	0.0555	0.0347
BM25 + ELMo	0.0201	0.0552	0.0346	0.0200	0.0552	0.0345	0.0200	0.0550	0.0344
BM25 + BERT	0.0203	0.0547	0.0342	0.0205	0.0557	0.0350	0.0205	0.0560	0.0351
BM25 + SciBERT	0.0202	0.0552	0.0346	0.0204	0.0563	0.0355	0.0203	0.0556	0.0348

least the abstracts. However, this will probably further increase the running time, and hence decrease the applicability of sentence encoders in systems that require recommendations in real-time.

REFERENCES

[1] Joeran Beel, Bela Gipp, Stefan Langer, and Corinna Breitinger. 2016. Research Paper Recommender Systems: A Literature Survey. *International Journal on Digital Libraries* 4 (2016), 305–338. https://doi.org/10.1007/s00799-015-0156-0

[2] Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. SciBERT: Pretrained Contextualized Embeddings for Scientific Text. *arXiv preprint arXiv:1903.10676* (2019).

[3] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder. *arXiv preprint arXiv:1803.11175* (2018).

[4] Andrew Collins and Joeran Beel. 2019. Document Embeddings vs. Keyphrases vs. Terms: A Large-Scale Online Evaluation in Digital Library Recommender Systems. In *Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL)*.

[5] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364* (2017).

[6] Zhuyun Dai and Jamie Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. *arXiv preprint arXiv:1905.09217* (2019).

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

[9] Chong Wang and David M Blei. 2011. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 448–456.

[10] Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Simple Applications of BERT for Ad Hoc Document Retrieval. *arXiv preprint arXiv:1903.10972* (2019).