KASYS at the NTCIR-15 WWW-3 Task

Kohei Shinden*
University of Tsukuba
s1913576@klis.tsukuba.ac.jp

Atsuki Maruta*
University of Tsukuba
s1711567@s.tsukuba.ac.jp

Makoto P. Kato
University of Tsukuba
mpkato@acm.org

ABSTRACT

The KASYS team participated in the English subtask of the NTCIR-15 WWW-3 Task. This paper describes our approach for generating NEW runs and REP (replicated/reproduced) runs in the NTCIR-15 WWW-3 Task. We applied BERT to the WWW-3 task for generating NEW runs, following a recent BERT-based approach for news retrieval. For replicating and reproducing the WWW-2 runs, we used an open-source information retrieval toolkit, Anserini, with some updates. NEW runs achieved the top performance in terms of nDCG, Q-measure, and iRBU, suggesting that BERT-based document ranking is highly effective not only for the other ranking tasks, but also for Web document retrieval. The results of REP runs were not well reproduced in the WWW-3 test collection, for which we discuss possible implementation differences from the original paper.

TEAM NAME

KASYS

SUBTASKS

English

1 INTRODUCTION

The KASYS team participated in the English subtask of the NTCIR-15 WWW-3 task [11]. This paper describes our approach for generating NEW runs, and REP (replicated/reproduced) runs in the NTCIR-15 WWW-3 task.

We applied BERT to the WWW-3 task for generating NEW runs. This approach was proposed by Yilmaz et al. [14], and was shown effective in news retrieval tasks. Applying their approach to the WWW-3 task, we investigated the effectiveness of BERT in the ad-hoc Web document retrieval task. The evaluation results showed that our runs achieved the highest performances in terms of nDCG, Q-measure, and iRBU, and suggests that BERT-based document ranking is highly effective not only for the other ranking tasks, but also for Web document retrieval.

We also tried replicating and reproducing the THUIR runs submitted to the NTCIR-14 WWW-2 task [15]. Their runs were generated by a standard BM25 algorithm and a learning-to-rank approach based on LambdaMART. We used Anserini [13], an open-source information retrieval toolkit, for increasing the reproducibility of our runs[1]. Since our implementation was not (or could not be) the same as the original implementation, we clarify some (possible) differences from the THUIR runs. The evaluation results showed that the results of REP runs were not well reproduced in the WWW-3 test collection, for which we discuss possible implementation differences from the original paper.

*The first two authors contributed equally to this work.
[1]The code of our REP runs is available at https://github.com/kasys-lab/anserini-kasys. See "Regressions for NTCIR-15 WWW-3 REP run" at "README.md" for details.

In the remainder of this paper, Sections 2 and 3 describe the details of the NEW and REP runs, respectively, and Section 4 concludes this paper with some future work.

2 NEW RUNS

BERT [3] has been successfully applied to a broad range of NLP tasks including document ranking tasks. One of the earliest attempts that applied BERT to ad-hoc document retrieval is Yilmaz et al.'s approach [14]. Since there are two problems in the application of BERT to ad-hoc document retrieval, they proposed a solution for these problems in their work. First, common web documents do not fit BERT, as the maximum input length of BERT is fixed at 512 tokens. Second, sentence-level relevance judgments were not usually available in ad-hoc document retrieval tasks, but are required to fine-tune the pre-trained BERT model. To overcome these problems, Yilmaz et al. estimated the document relevance based on the relevance between a query and sentences in a document, and tuned the BERT model with the other tasks such as QA and microblog search task. Followed by their successful application of BERT to news retrieval tasks, we investigated the effectiveness of BERT in Web document retrieval.

2.1 Model

Our NEW runs were generated by Yilmaz et al.'s approach, with some WWW-3 specific settings. We briefly explain their model, and then describe the details of our implementation and datasets used for training.

As mentioned earlier, the pre-trained BERT model was fine-tuned for estimating the sentence-level relevance for a given query. Thus, the input for fine-tuning was created by concatenating query q and sentence s as follows: [[CLS], q, [SEP], s, [SEP]], and padding each sequence in a mini-batch to the maximum length in the batch for efficient implementation. [CLS] and [SEP] are BERT's own tokens, where "[CLS]" represents a classifier embedding and "[SEP]" represents a sentence delimiter. The final hidden layer corresponding to token "[CLS]" was fed into an additional neural network with a single layer, which was trained to predict the relevance of the sentence to the given query. When MS MARCO data [5] was used for fine-tuning, for example, a query and a passage were fed into the BERT model, which is trained to output 1 if the passage is relevant to the query; otherwise 0.

The relevance of a document is estimated by the relevance of sentences in the document. Assuming that the most relevant sentences in a document are good indicators of the document-level relevance, Yilmaz et al. aggregates a document score (e.g. BM25) as well as the relevance of the top k sentences estimated by the fine-tuned BERT for document ranking:

$$f(d) = f_{\text{BM25}}(d) + \sum_{i=1}^{k} w_i \cdot f_{\text{BERT}}(p_i) \quad (1)$$

Table 1: Evaluation results of KASYS's NEW runs.

Run	Model	nDCG	Q	ERR	iRBU
KASYS-E-CO-NEW-1	BERT(MS MARCO → MB, $k = 3$)	0.6935	0.7123	0.7959	0.9382
KASYS-E-CO-NEW-4	BERT(MS MARCO → MB, $k = 2$)	0.6893	0.7073	0.7893	0.9389
KASYS-E-CO-NEW-5	BERT(CAR → MB, $k = 2$)	0.6812	0.6990	0.7768	0.9298
baselineEng	(A baseline run at WWW-3)	0.5748	0.5850	0.6757	0.8802

Table 2: Preliminary evaluation results of KASYS's NEW runs in the WWW-2 test collection.

Setting	nDCG@10	Q@10	nERR@10
$k = 1$			
MB	0.3097	0.2701	0.4687
CAR	0.3112	0.2832	0.4653
MS MARCO	0.3103	0.2840	0.4716
CAR → MB	0.3243	0.2847	0.4645
MS MARCO → MB	0.3263	0.2939	0.4658
$k = 2$			
MB	0.3186	0.2778	0.4694
CAR	0.3112	0.2832	0.4653
MS MARCO	0.2989	0.2670	0.4611
CAR → MB	0.3259	0.2859	0.4722
MS MARCO → MB	0.3273	0.2948	0.4769
$k = 3$			
MB	0.3266	0.2869	0.4662
CAR	0.3104	0.2832	0.4640
MS MARCO	0.2963	0.2662	0.4611
CAR → MB	0.3312	0.2906	0.4725
MS MARCO → MB	0.3318	0.2998	0.4857

where, $f_{\text{BM25}}(d)$ is the BM25 score of document d, and $f_{\text{BERT}}(p_i)$ is the sentence relevance of the top i-th sentence obtained by BERT. The hyper-parameter w_i is to be tuned with a validation set.

2.2 Implementation Details

Following Yilmaz et al.'s work [14], we used $\text{BERT}_{\text{Large}}$ (uncased, 340M parameters) as a pre-trained BERT model, and four datasets for fine-tuning the pre-trained BERT model: TREC Microblog (MB) [5], MAchine Reading COmprehension (MS MARCO) [8], and TREC Complex Answer Retrieval (CAR) [4]. MB consists of queries and tweets with their relevance from the TREC Microblog track that were organized from 2011 to 2014. MS MARCO is a dataset consisting of queries sampled from Bing's search logs and sentences extracted from Web documents. TREC CAR is a question answering task and provided a dataset including queries and paragraphs. Each query was constructed by concatenating a article title and a section title of a Wikipedia article, and paragraphs were considered relevant for a query if they are in the section whose title is used as the query.

To select appropriate datasets and hyper-parameters for Web document ranking, we conducted a preliminary experiment with the dataset from the Robust Track at TREC 2004 (Robust04) [12], and NTCIR-14 WWW-2 test collection [7]. We trained the BERT models with four datasets explained above, and measured the performance with Robust04. The following hyper-parameter settings were tested: $w_i = 0.0, 0.1, \ldots, 1.0$. The best hyper-parameter for each dataset and the other hyper-parameter setting $k = 1, 2, 3$ was determined by MAP on Robust04. We also measured the performance with the NTCIR-14 WWW-2 test collection (with its original qrels), which is shown in Table 2. Datasets used for training are presented with the hyper-parameter setting for k. Note that "X → Y" indicates that the BERT model was first fine-tuned with X dataset, and further fine-tuned with Y dataset. As we found that the combination of MS MARCO and MB with $k = 2$ and $k = 3$ and that of CAR and MB with $k = 2$ achieved the highest nDCG@10, they were used for our NEW runs, namely, KASYS-E-CO-NEW-1, KASYS-E-CO-NEW-4, and KASYS-E-CO-NEW-5.

To generate our NEW runs, we first retrieved documents from the ClueWeb12-B13 collection by using the BM25 algorithm. This ranked list is exactly the same as that of REP B-run, which is explained later. The documents were then ranked using Equation 1. Birch [1], a document ranking tool based on BERT, was used for our implementation.

2.3 Experimental Results

Table 1 shows the results of our NEW runs for WWW-3 topics, together with that of a baseline run provided by the WWW-3 organizers. According to the overview paper of NTCIR-15 WWW-3 [11], our BERT-based approach achieved the highest nDCG@10, Q-measure, and iRBU@10. Significant differences were found between KASYS-E-CO-NEW-1 and 20 runs submitted to WWW-3 in terms of nDCG@10, and suggested that Yilmaz et al.'s approach was also effective for Web document retrieval. Comparing our NEW runs, we found that the combination of MS MARCO and MB performed better than that of CAR and MB, though there was not a significant difference. Since we can hypothesize that BERT is a key component of our runs, it would be interesting to see the performance difference between runs with and without BERT.

3 REP RUNS

This section explains the detailed procedure to replicate the THUIR runs submitted to the NTCIR-14 WWW-2 task. The BM25 run, called *REP B-run* in the overview paper [11], was generated by ranking ClueWeb documents by the BM25 algorithm implemented in Anserini. We followed the regression instruction at the Anserini website[2]. The LambdaMART run (*REP A-run*) was obtained by

[2]https://github.com/castorini/anserini/blob/master/docs/regressions-cw12b13.md

Table 3: Differences between THUIR's run at NTCIR-14 and KASYS'run at NTCIR-15.

Component	THUIR	KASYS (Ours)
Fields from which features were extracted	Whole document, anchor text, title, and URL	Whole document
Feature extraction methods	Their own algorithms	Implementation based on Anserini's feature extractor
Parameters for LMIR features	(Not described)	LMIR.JM $\lambda = 0.1$, LMIR.DIR $\mu = 2000$, LMIR.ABS $\delta = 0.7$
Normalization	(Not described)	Normalized into $[0, 1]$ for each query and feature

re-ranking the top 1,000 documents in REP B-run. We also used Anserini for feature extraction to increase the reproducibility of our replication. An open-source learning-to-rank toolkit, RankLib [2], was used for ranking documents by LambdaMART. The LambdaMART model was trained with MQ2007 and MQ2008, and was tuned with the NTCIR-13 WWW-1 test collection [6].

Our implementation was not exactly the same as the original implementation. Table 3 summarizes the differences from THUIR's implementation. One of the largest differences is the fields from which the features were extracted. THUIR extracted the features from the four fields, while we only processed a single field. Another possible difference can be feature extraction implementation. Ours was mainly based on Anserini's batch program called "FeatureExtractorCli". Hyper-parameter settings and feature normalization could be other possible difference from the original implementation.

3.1 Implementation Details

Below, we further describe the implementation details of our REP-A run.

We extracted the features of the top 1,000 documents in our REP B-run. These features are exactly the same as the features used by THUIR in their LambdaMART run, and are listed in Table 4. As mentioned earlier, we used Anserini's "FeatureExtractorCli" for feature extraction. Since only four features, namely, TF, TF*IDF, DL, and BM25, were implemented in Anserini, we additionally implemented the other features: IDF, LMIR.ABS, LMIR.DIR, and LMIR.JM, by extending "FeatureExtractorCli". Based on the settings described by Qin et al. [10], we set hyper-parameters of the language model based retrieval models as follows: $\lambda = 0.1$ in LMIR.JM, $\mu = 2000$ in LMIR.DIR, and $\delta = 0.7$ in LMIR.ABS. The extracted features were normalized for each query and feature. Normalization squashed values into $[0, 1]$ by using the maximum and minimum values of each feature. If the maximum and minimum values were the same for a feature, the values of the feature were set to 0. When there is a query word that does not exist in any document, the values for LMIR features were also set to 0.

Although THUIR extracted the features from four fields, namely, the whole document, anchor text, title, and URL, we extracted the features only from the whole document since we needed to change several parts of the program in Anserini for dealing with multiple fields.

We used learning-to-rank datasets, MQ2007 and MQ2008 [9], for training a LambdaMART model. While both datasets had been divided into train, validation, and test sets, all of them were combined together and used to train the model. The WWW-1 qrel file was used as a validation set. Since the relevance grades of WWW-1 (a five point-scale) are different from those of MQ2007 and MQ2008 (a three point-scale), the relevance grades L1 and L2 in WWW-1 were converted into L1, and L3 and L4 in WWW-1 were converted into L2. RankLib was used for training the model as well as for ranking the top 1,000 documents in our REP B-run to produce our REP A-run.

Table 4: Features used for LambdaMART.

ID	Features
1	TF (Term frequency)
2	IDF (Inverse document frequency)
3	TF*IDF
4	DL (Document length)
5	BM25
6	LMIR.ABS
7	LMIR.DIR
8	LMIR.JM

3.2 Experimental Results

Tables 5 and 6 show the results of Rep A-run (LambdaMART) and Rep B-run (BM25) for WWW-2 and WWW-3 topics, respectively. As for the evaluation of replicability (WWW-2 topics), since REP B-run outperformed REP A-run in terms of all the evaluation metrics but ERR, we could not replicate the result in WWW-2 that A-run outperformed B-run. As for the evaluation of reproducibility (WWW-3 topics), since REP B-run outperformed REP A-run in terms of all the evaluation metrics, we could not reproduce the result in WWW-2 that A-run outperformed B-run. There are two possible interpretations for this result: the result in WWW-2 could not be generated to that in WWW-3, or the differences between the original implementation and ours caused the different experimental results.

4 CONCLUSIONS

The KASYS team participated in the English subtask of the NTCIR-15 WWW-3 Task. Our NEW runs are mainly based on a BERT-based document ranking method proposed in an existing work, and achieved the top performance in terms of nDCG, Q-measure, and iRBU. This result suggested that BERT-based document ranking is highly effective not only for the other ranking tasks, but also for Web document retrieval. The results of REP runs were not well reproduced in the WWW-3 test collection, for which we discussed possible implementation differences from the original paper.

Our future work includes per-query analysis of the evaluation results of our NEW runs, and further failure analysis on our REP runs.

Table 5: Evaluation results of KASYS's REP runs for WWW-2 topics (Replicability evaluation).

Run	Model	nDCG	Q	ERR	iRBU
KASYS-E-CO-REP-2	Rep A-run (LambdaMART)	0.6742	0.6904	0.7904	0.9256
KASYS-E-CO-REP-3	Rep B-run (BM25)	0.6803	0.6979	0.7904	0.9334

Table 6: Evaluation results of KASYS's REP runs for WWW-3 topics (Reproducibility evaluation).

Run	Model	nDCG	Q	ERR	iRBU
KASYS-E-CO-REP-2	Rep A-run (LambdaMART)	0.6131	0.6256	0.7213	0.8876
KASYS-E-CO-REP-3	Rep B-run (BM25)	0.6275	0.6402	0.7410	0.9037

REFERENCES

[1] Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Cross-Domain Modeling of Sentence-Level Evidence for Document Retrieval. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3490–3496. https://doi.org/10.18653/v1/D19-1352

[2] V Dang. 2013. The lemur project-wiki-ranklib. *Lemur Project* (2013).

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*. 4171–4186.

[4] Laura Dietz, Manisha Verma, Filip Radlinski, and Nick Craswell. 2017. TREC Complex Answer Retrieval Overview. In *TREC*.

[5] Jimmy Lin, Yulu Wang, Miles Efron, and Garrick Sherman. 2014. Overview of the TREC-2014 Microblog Track. In *Proceedings of The Twenty-Third Text REtrieval Conference, TREC 2014, Gaithersburg, Maryland, USA, November 19-21, 2014 (NIST Special Publication, Vol. 500-308)*, Ellen M. Voorhees and Angela Ellis (Eds.). National Institute of Standards and Technology (NIST). https://trec.nist.gov/pubs/trec23/papers/overview-microblog.pdf

[6] Cheng Luo, Tetsuya Sakai, Yiqun Liu, Zhicheng Dou, Chenyan Xiong, and Jingfang Xu. 2017. Overview of the ntcir-13 we want web task. (2017).

[7] Jiaxin Mao, Tetsuya Sakai, Cheng Luo, Peng Xiao, Yiqun Liu, and Zhicheng Dou. 2019. Overview of the NTCIR-14 We Want Web Task. In *The 14th NTCIR Conference*.

[8] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *CoRR* abs/1611.09268 (2016). arXiv:1611.09268 http://arxiv.org/abs/1611.09268

[9] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 datasets. *arXiv preprint arXiv:1306.2597* (2013).

[10] Tao Qin, Tie-Yan Liu, Jun Xu, and Hang Li. 2010. LETOR: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval* 13, 4 (2010), 346–374.

[11] Tetsuya Sakai, Sijie Tao, Zhaohao Zeng, Yukun Zheng, Jiaxin Mao, Zhumin Chu, Yiqun Liu, Maria Maistro, Zhicheng Dou, Nicola Ferro, and Ian Soboroff. 2020. Overview of the NTCIR-15 We Want Web with CENTRE (WWW-3) Task. In *Proceedings of NTCIR-15*. to appear.

[12] Ellen M Voorhees. 2005. The TREC robust retrieval track. In *ACM SIGIR Forum*, Vol. 39. 11–20.

[13] Peilin Yang, Hui Fang, and Jimmy Lin. 2018. Anserini: Reproducible Ranking Baselines Using Lucene. *Journal of Data and Information Quality* 10, 4, Article 16 (Oct. 2018), 20 pages. https://doi.org/10.1145/3239571

[14] Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Cross-domain modeling of sentence-level evidence for document retrieval. In *EMNLP-IJCNLP*. 3481–3487.

[15] Yukun Zheng, Zhumin Chu, Xiangsheng Li, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2019. THUIR at the NTCIR-14 WWW-2 Task. In *NII Conference on Testbeds and Community for Information Access Research*. 165–179.