Can a Machine Reading Comprehension Model Improve Ad-hoc Document Retrieval?

Kota Usuha[1](✉), Makoto P. Kato[1], and Sumio Fujita[2]

[1] University of Tsukuba, Tsukuba, Japan
k-ush@klis.tsukuba.ac.jp, mpkato@acm.org
[2] Yahoo Japan Corporation, Tokyo, Japan
sufujita@yahoo-corp.jp

Abstract. We propose a method to solve ad-hoc document retrieval tasks using a reading comprehension model. To solve the ad-hoc retrieval task, the proposed method generates a question for the given query, and a reading comprehension model is employed to determine whether the target document contains a corresponding answer to the generated question, thereby estimating the relevance of the document. Experimental results show that a simple application did not improve the performance in ad-hoc retrieval tasks. Through extensive analysis of the experimental results, however, we found that the proposed method was effective for improving the performance when it was applied to queries containing proper nouns.

Keywords: Information retrieval · Question answering · Reading comprehension

1 Introduction

The ad-hoc retrieval task, which is the central task in information retrieval, involves ranking documents based on their estimated relevance for a given query. On the other hand, the *machine reading comprehension* task attempts to extract an answer to a given question from a given text.

The ad-hoc retrieval and machine reading comprehension algorithms, which we refer to as the *retriever* and *reader*, respectively, have been developed rapidly due to recent advances in neural network models and large-scale datasets, e.g., MS MARCO [1] and SQuAD [18]. State-of-the-arts in those tasks are based on a neural language model pre-trained using a large text corpus, e.g., bidirectional encoder representations from transformers (BERT) [7]. In open-domain question answering tasks, which include the ad-hoc passage retrieval and machine reading comprehension tasks, pre-trained models fine-turned on the same question answering dataset are used as the retriever and reader [6,11,16,22]. Thus, the distinction between ad-hoc retrieval and machine reading comprehension becomes less clear due to the universal models that can be used for various NLP tasks.

However, despite the many similarities between these two tasks, the applicability of employing a model fine-tuned for one task to the other task has not been investigated extensively. If a fine-tuned reader model could be employed in ad-hoc retrieval tasks, task efficiency could be improved in various ways:

Zero Training Time. The training time for a retriever model can be eliminated, because we no longer have to fine-tune a model for the ad-hoc retrieval tasks.

Zero Resource. The preparation of datasets to train retriever models is not required, which is beneficial to developing multi-lingual retrievers. Various multi-lingual resources are available for reading comprehension tasks [3,10]; however, only synthetic multi-lingual datasets are available for ad-hoc retrieval [2].

High Research Efficiency. The performance improvement of reading comprehension tasks can be introduced to ad-hoc retrieval tasks, which may lead to more efficient development of ad-hoc retrieval algorithms.

Thus, in this paper, we propose a method to directly apply a fine-tuned reader model to ad-hoc retrieval tasks. The proposed method, which we refer to as the **A**d-hoc **I**nformation **R**etrieval model based on machine **Read**ing comprehension (AIRRead), transforms a keyword query into the latent questions hidden behind the query. Then, a reader estimates the relevance of each document by determining whether a corresponding answer is contained in a given document. Our experimental results demonstrated that selective application of AIRRead improved the ad-hoc retrieval performance compared to the standard baselines.

2 Related Work

BERT-based ad-hoc retrieval can be divided into two main categories. The first is where BERT is employed to embed documents and queries separately in order to obtain embedded representations. Then, the cosine similarity between the embedded representations of the query and the document is used as the relevance score [8,24]. In the second category, BERT is employed to encode documents and queries jointly. We expect BERT to output a relevance score of the input document for the given query [14,23].

The machine reading comprehension is used in the question answering task, which involves extracting an answer to a given question from a passage. Otsuka et al. proposed a method that transforms input questions to questions with more detailed content prior to inputting them into a reading comprehension model [15]. In question answering tasks, reading comprehension is responsible for extracting answers; however, in an open-domain question answering task, it is necessary to efficiently search for passages to input to the reading comprehension model. Nishida et al. proposed a method that incorporates multi-task learning in an open-domain reading comprehension task, where the same model is employed to retrieve the passages and extract the answers [13].

Similarly, in AIRRead, we employ a reading comprehension model and input documents and queries to BERT jointly. However, the task we tackle is an ad-hoc retrieval task and we directly apply a machine reading model to ad-hoc retrieval tasks. In addition, AIRRead employs a trained model for the machine reading task; thus, additional model training is not required.

3 Methodology

Here, we describe the methodology used to generate a question from a query, and how relevance estimation is performed using a trained reading comprehension model (or *reader*).

3.1 Problem Setting

Let D be a document collection. We estimate the relevance score s_i of documents $d_i \in D$ for the given q_r, and rank documents in descending order of the relevance score. To estimate the relevance score, we do not train a relevance estimation model, but employ a trained reader. Thus, a training process is not required for the relevance estimation.

3.2 Framework

Here, given a query, we first retrieve an initial ranked list of documents D' from the document set D with a search model that can be retrieved rapidly via indexing, e.g., BM25. Query q_r is transformed to a question q_s by the question generation model g, which is used as input to the reader, and the relevance score is estimated for each document in D' by the reader f. A question-document pair is an input to the reader to obtain the relevance score. Formally, the relevance score s_i of the i-th document d_i in D' is estimated as follows:

$$q_s = g(q_r),$$
$$s_i = f(q_s, d_i).$$

When translating a query into a question, the information needs must be shared between query and question to capture the original information needs. If the information needs of the query are ambiguous or underspecified, multiple questions may be necessary to represent hidden information needs. In such cases, the relevance of a document is estimated based on multiple questions, and the relevance scores for each question are aggregated into a single score. A document can be highly relevant if it covers major questions behind a query. Thus, when multiple questions are generated from a query, the maximum value is used as the relevance score as follows:

$$Q_s = g(q_r),$$
$$s_i = \max_{q \in Q_s} f(q, d_i),$$

where Q_s is the set of questions generated by $g(q_r)$.

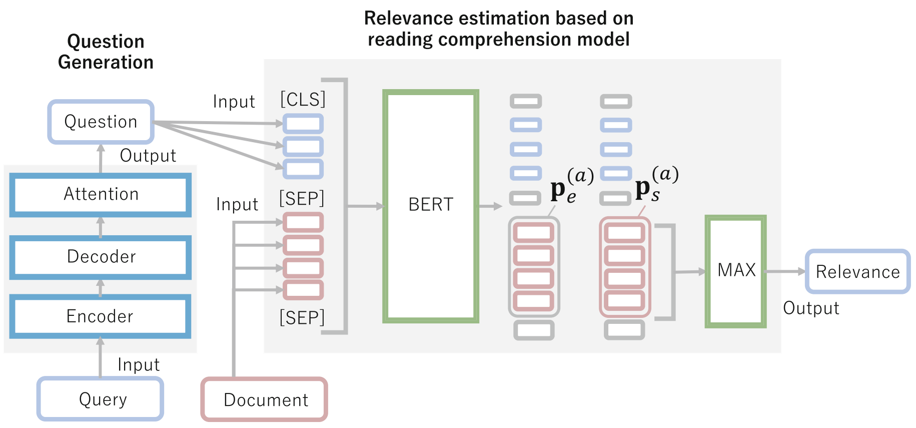

Fig. 1. Flow of the proposed method

Algorithm 1. Generating a query from a question

```
tokens ← Tokenize(question)
length ← get uniformly distributed random number from 1 to 3
length ← max(length, GetLength(question))
query ← {}
for 1 ... length do
    token ← $\arg\max_{token \in tokens} \mathrm{idf}(token)$
    query ← query ∪ {token}
end for

return query
```

3.3 AIRRead

In this section, we describe the proposed method, AIRRead, in detail. Figure 1 shows the flow of the proposed method.

3.4 Question Generation

To generate questions from queries, several methods have been proposed [4,9,20]. In AIRRead, We treat the process of generating a question from a query as a text-to-text translation process [9]; thus, we employed a machine translation model to generate the query into a question. We constructed a dataset in which queries and questions are paired by generating queries from questions to train a machine translation model.

Algorithm 1 describes the procedure used to generate a query from a question. For Generally, queries are shorter than questions, and approximately 76% of queries include three words or less [21]. Thus, the length of the query to be generated is determined randomly to be uniformly distributed in the range of 1–3. As stated previously, the query to be generated must share the information needs of the question; thus, the words used as the query are extracted from words included in the question. The words to be extracted are determined by their inverse document frequency (IDF) under the assumption that a word with a lower occurrence frequency contains more information. Then, as many times as the query length, one of the tokens in the question with the highest IDF value is extracted to form a query.

3.5 Relevance Estimation Based on Reading Comprehension

Here, we describe the method used in AIRRead to estimate relevance in ad-hoc document retrieval using a trained reading comprehension model (or *reader*).

Typically, a reader is employed for the question answering task, which takes a passage and a question, and then extracts the answer to the question from the given passage [5]. Here, the answer is presented as a span in the passage. Thus, a reader outputs two probabilities for each token. i.e., one is the probability that the answer span begins with that token, and the other is the probability that the answer span ends with that token.

We employ BERT as the reader, where a sequence of questions and passages is input, and the probability of the beginning and end of the answer span for each input token is output. When inputting a question q_s and passage a, we add [SEP] between the question and the passage and at the end of the input sequence, and we add [CLS] at the beginning of the input sequence. In this case, the output is two probability distributions over each token in the input sequence. As we are especially interested in whether an answer exists in the given passage, let $\mathbf{p}_s^{(a)}$ be the probability that the passage token is the beginning of the answer span, and $\mathbf{p}_e^{(a)}$ be the probability that the passage token is the end of the answer span. More specifically, the answer beginning probability $\mathbf{p}_s^{(a)}$ is defined as $\mathbf{p}_s^{(a)} = (p_{s,1}^{(a)}, p_{s,2}^{(a)}, \ldots, p_{s,|a|}^{(a)})$ where $|a|$ is the length of the passage a. The answer end probability $\mathbf{p}_e^{(a)}$ is defined similarly.

A passage is considered relevant if it contains at least an answer to a generated question. Thus, the relevance score of the passage a in the document d_i, denoted by $s_{i,a}$, is defined as the maximum beginning probability of passage tokens:

$$s_{i,a} = \max_{1 \leq j \leq |a|} p_{s,j}^{(a)}$$

We fine-tune BERT on SQuAD 2.0 [17], which is a dataset for reading comprehension tasks. Unlike SQuAD 1.0, SQuAD 2.0 includes questions cannot be answered from the passage. If a question is determined to be unanswerable, BERT is trained such that the position of the [CLS] token at the beginning of the input sequence becomes the answer interval. As a result, when the reader

Table 1. Statistics of the constructed query-question dataset.

Data size	Average query length	Average question length
727,858	1.99	6.38

finds a passage not able to answer a question, we expect the reader to output low probabilities for all the passage tokens and, accordingly, a low relevance score for the given passage.

4 Experiments

In this section, we describe the experimental settings and results.

4.1 Datasets

To train a model that can translate a query to a question, we constructed a dataset of query-question pairs using the questions in the MS MARCO dataset. For the constructed dataset, the construction process is as described in Sect. 3.4, and the statistics are given in Table 1.

To evaluate AIRRead, we employed the English NTCIR WWW-2 [12] and WWW-3 [19] test collections, which are standard test collections for ad-hoc document retrieval tasks.

4.2 Experimental Settings

The following methods were used as baselines in our experiments: *BM25 (WWW)*, which is provided as a baseline in NTCIR WWW-2 and WWW-3; *BM25 (Ours)*, which is BM25 used in our experiment (slightly different from BM25 (WWW) due to some configuration differences); and *Birch* [23], which achieved the best performance in the NTCIR-15 WWW-3 English subtask. Birch is a BERT-based ad-hoc retrieval model that estimates the document relevance by aggregating sentence-level evidence. We used three standard evaluation metrics for retrieval tasks, i.e., nDCG@10, Q@10, and nERR@10.

For the question generation model, we used an encoder-decoder with an attention mechanism trained on the constructed query-question dataset (Sect. 4.1). While we generated multiple questions and performed relevance estimation, we found that this did not contribute to performance improvement. Thus, we opt to use a single question for each query.

4.3 Initial Results

Table 2 shows the experimental results obtained by the baselines and AIRRead. As can be seen, on the WWW-3 test collection, AIRRead outperformed BM25

Table 2. Experimental results of the baselines and AIRRead.

Method	WWW-3			WWW-2		
	nDCG	Q	nERR	nDCG	Q	nERR
BM25 (WWW)	0.575	0.585	0.676	0.326	**0.304**	0.478
BM25 (Ours)	0.628	0.639	0.744	0.317	0.291	0.459
Birch	**0.694**	**0.712**	**0.796**	**0.334**	0.300	**0.486**
AIRRead	0.627	0.636	0.735	0.303	0.281	0.424

(WWW) in terms of all considered metrics but did not outperform Birch. However, when we look at the results for BM25 (Ours), we see that BM25 (Ours) outperformed AIRRead for all metrics. These results suggest that a simple application of the reader cannot improve the performance of ad-hoc retrieval baselines. We then hypothesized that AIRRead is effective for a special type of queries, and devised a selective application of AIRRead based on extensive analysis of the experimental results.

4.4 Selective Application of AIRRead

From the document rankings obtained by BM25 (Ours), for each query, we examined how much reranking by the reader improved the rankings. We define the improvement rate of reranking by the reader as follows.

$$\text{Improvement rate} = \frac{\text{nDCG}_{\text{RC}}}{\text{nDCG}_{\text{BM25}}}$$

where nDCG_{RC} is the nDCG of the document rankings obtained by AIRRead, and $\text{nDCG}_{\text{BM25}}$ is the nDCG of the document rankings obtained by BM25 (Ours). When $\text{nDCG}_{\text{BM25}}$ is 0, nDCG_{RC} is also 0; thus, the improvement rate is set to 0. An improvement rate greater than 1 indicates that AIRRead improved the ranking of BM25 in terms of nDCG.

For the WWW-3 queries, we sorted the rankings obtained by AIRRead (Manual)[1] in descending order of improvement rate and examined the percentage of parts of speech in the top-20 and bottom-20 queries. Table 3 shows the results of sorting in descending order by the percentage of parts-of-speech in the overall, top-20, and bottom-20 queries, as well as the difference between the percentage of each part-of-speech in the top-20 and bottom-20 queries. As can be seen, the difference between the top-20 and bottom-20 in terms of the improvement rate for proper nouns was the largest; thus, we consider that reranking via AIRRead is effective for queries containing proper nouns. For nouns, the difference between the top-20 and bottom-20 was the smallest; however, considering that the ratio of nouns to all queries is as high as 0.459, more research is required to conclude that the AIRRead method's reranking process has a negative effect on performance improvement for queries containing nouns.

[1] Here, we used *Manual* configuration to isolate the effect of the question generation.

Table 3. Percentage of POS in the top-20 queries when queries were sorted in descending order of improvement rate. The difference is the percentage of parts of speech in the top-20 queries minus the percentage of parts of speech in the bottom-20 queries. Only the top three and bottom three of POS are shown.

POS	All	Top-20	Bottom-20	Top-20 − Bottom-20
Proper noun	0.238	0.486	0.245	**+0.241**
Adverb	0.022	0.057	0.019	+0.038
Other	0.006	0.029	0.000	+0.029
Determiner	0.022	0.000	0.019	−0.019
Adposition	0.050	0.029	0.075	−0.046
Noun	0.459	0.257	0.472	−0.215

Table 4. Experimental results of selective application of AIRRead.

Method	WWW-3			WWW-2		
	nDCG	Q	nERR	nDCG	Q	nERR
AIRRead	0.627	0.636	0.735	0.303	0.281	0.424
Selective	0.627	0.635	0.745	**0.320**	**0.295**	**0.474**
Manual and Selective	**0.629**	**0.639**	**0.749**	0.319	**0.295**	0.472
WhatIs and Selective	0.627	0.635	0.737	0.319	**0.295**	0.469

Since AIRRead was particularly effective for queries containing proper nouns, we devised *Selective* approach, which only applies AIRRead to queries that contain at least a proper noun. For the other queries, the documents list ranked by BM25 was output without reranking. Table 4 compared Selective methods with following different question generation strategies. *WhatIs*, which generated questions by simply adding "What is" to the beginning of the given query. *Manual*, which manually transformed the queries given in the English subtask of NTCIR15 WWW-3 and WWW-2 into questions. The transformation was performed by the authors, who read the description field describing the information needs of the query. From Table 4, we found that AIRRead (Selective) outperformed AIRRead for all evaluation metrics, except for Q of the WWW-3 test collection. While the selective application alone was effective for the WWW-2 test collection, high-quality questions (Manual) were necessary to achieve decent performance improvements for the WWW-3 test collection. Comparing the best performances achieved by the selective approaches with the baselines in Table 2, we can observe some improvements over the BM25 baselines in the WWW-3 test collection. These results indicate that the reading comprehension model contributed to the performance improvement via selective reranking.

5 Conclusion

In this paper, we have proposed a method to address the problem of generating questions from queries and handle ad-hoc document retrieval tasks using trained machine reading comprehension models. We found that, compared to the BM25 method, the trained reading comprehension model worked well in terms of document reranking for queries containing proper nouns. In future, we would like further investigate the tendency of effective queries.

References

1. Bajaj, P., et al.: MS MARCO: a human generated machine reading comprehension dataset (2018). https://arxiv.org/abs/1611.09268
2. Bonifacio, L.H., Campiotti, I., Jeronymo, V., Lotufo, R., Nogueira, R.: MMARCO: a multilingual version of the MS MARCO passage ranking dataset. arXiv preprint arXiv:2108.13897 (2021)
3. Clark, J.H., Choi, E., Collins, M., Garrette, D., Kwiatkowski, T., Nikolaev, V., Palomaki, J.: TYDI QA: a benchmark for information-seeking question answering in typologically diverse languages. Trans. Assoc. Comput. Linguist. **8**, 454–470 (2020)
4. Dror, G., Maarek, Y., Mejer, A., Szpektor, I.: From query to question in one click: suggesting synthetic questions to searchers. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 391–402 WWW 2013, Association for Computing Machinery, New York, NY, USA (2013). https://doi.org/10.1145/2488388.2488423
5. Hermann, K.M., et al.: Teaching machines to read and comprehend. In: Proceedings of the 28th International Conference on Neural Information Processing Systems, vol. 1, pp. 1693–1701. NIPS2015, MIT Press, Cambridge, MA, USA (2015)
6. Karpukhin, V., et al.: Dense passage retrieval for open-domain question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6769–6781 (2020)
7. Kenton, J.D.M.W.C., Toutanova, L.K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT, pp. 4171–4186 (2019)
8. Khattab, O., Zaharia, M.: ColBERT: efficient and effective passage search via contextualized late interaction over BERT. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 39–48 (2020)
9. Kumar, A., Dandapat, S., Chordia, S.: Translating web search queries into natural language questions. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan (2018). https://aclanthology.org/L18-1151
10. Longpre, S., Lu, Y., Daiber, J.: MKQA: A linguistically diverse benchmark for multilingual open domain question answering. arXiv preprint arXiv:2007.15207 (2020)
11. Luan, Y., Eisenstein, J., Toutanova, K., Collins, M.: Sparse, dense, and attentional representations for text retrieval. Trans. Assoc. Comput. Linguist. **9**, 329–345 (2021)

12. Mao, J., Sakai, T., Luo, C., Xiao, P., Liu, Y., Dou, Z.: Overview of the NTCIR-14 we want web task. In: NTCIR-14 Conference (2019)
13. Nishida, K., Saito, I., Otsuka, A., Asano, H., Tomita, J.: Retrieve-and-read: multi-task learning of information retrieval and reading comprehension. ACM (2018). https://doi.org/10.1145/3269206.3271702
14. Nogueira, R., Yang, W., Cho, K., Lin, J.: Multi-stage document ranking with BERT (2019). https://arxiv.org/abs/1910.14424
15. Otsuka, A., Nishida, K., Saito, I., Asano, H., Tomita, J., Satoh, T.: Reading comprehension based question answering technique by focusing on identifying question intention. vol. 34, pp. 1–12 (2019).https://doi.org/10.1527/tjsai.A-J14
16. Qu, Y., et al.: Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5835–5847 (2021)
17. Rajpurkar, P., Jia, R., Liang, P.: Know what you don't know: unanswerable questions for SQuAD. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 784–789 Association for Computational Linguistics, Melbourne, Australia (2018). https://doi.org/10.18653/v1/P18-2124
18. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100,000+ questions for machine comprehension of text. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2383–2392 Association for Computational Linguistics, Austin, Texas (2016). https://doi.org/10.18653/v1/D16-1264
19. Sakai, T., et al.: Overview of the NTCIR-15 we want web with centre (WWW-3) task. In: NTCIR-15 Conference (2020)
20. Shiqi, Z., Haifeng, W., Chao, L., Ting, L., Yi, G.: Automatically generating questions from queries for community-based question answering. In: Proceedings of 5th International Joint Conference on Natural Language Processing, pp. 929–937. Asian Federation of Natural Language Processing, Chiang Mai, Thailand (2011)
21. Spink, A., Wolfram, D., Jansen, M.B.J., Saracevic, T.: Searching the web: the public and their queries. vol. 52, pp. 226–234
22. Xiong, L., et al.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: International Conference on Learning Representations (2020)
23. Yilmaz, Z.A., Yang, W., Zhang, H., Lin, J.: 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), pp. 3490–3496. Association for Computational Linguistics (2019)
24. Zhan, J., Mao, J., Liu, Y., Zhang, M., Ma, S.: RepBERT: contextualized text embeddings for first-stage retrieval (2020). https://arxiv.org/abs/2006.15498