Query Context Expansion for Open-Domain Question Answering

WENHAO ZHU, XIAOYU ZHANG, LIANG YE, and QIUHONG ZHAI, Shanghai University, China

Humans are accustomed to autonomously associating prior knowledge with the text in a query when answering questions. However, for machines lacking cognition and common sense, a query is merely a combination of some words. Although we can enrich the semantic information of the given query through language representation or **query expansion (QE)**, the information contained in the query is still insufficient. In this paper, we propose an effective passage retrieval method named **query context expansion-based retrieval (QCER)** for **open-domain question answering (OpenQA)**. QCER associates a query with domain information by adding contextual association information based on the **pseudo-relevance feedback (PRF)**. QCER uses a dense reader to select top-n expansion terms for QE. We implement QCER by appending reader predictions, theoretically present in candidate passages, as contextual information to the initial query to form the new query. QCER with sparse representations (BM25) can improve retrieval efficiency and accelerate query convergence so that the reader can find the desired answer using fewer relevant passages, e.g., 10 passages, as soon as possible. Moreover, QCER can be easily combined with **dense passage retrieval (DPR)** to achieve even better performance, as sparse and dense representations are often complementary. Remarkably, we demonstrate that QCER achieves state-of-the-art performance in three tasks, passage retrieval, passage reading, and passage reranking, on the **Natural Questions (NQ)** and **TriviaQA (Trivia)** datasets under an extractive QA setup.

CCS Concepts: • **Information systems** → **Query reformulation** • **Computing methodologies** → **Lexical semantics**;

Additional Key Words and Phrases: Open-domain question answering, query expansion, passage retrieval, pseudo-relevance feedback

ACM Reference format:
Wenhao Zhu, Xiaoyu Zhang, Liang Ye, and Qiuhong Zhai. 2023. Query Context Expansion for Open-Domain Question Answering. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 22, 8, Article 206 (August 2023), 21 pages.
https://doi.org/10.1145/3603498

1 INTRODUCTION

This work is supported by the National Natural Science Foundation of China (No. 61572434) and the Shanghai Science and Technology Committee (No. 19DZ2204800).
Authors' address: W. Zhu, X. Zhang, L. Ye (corresponding author), and Q. Zhai, Shanghai University, Shanghai, China, 200444; emails: {whzhu, xiaoyu121, liangye}@shu.edu.cn, 18721415231@163.com.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
2375-4699/2023/08-ART206 $15.00
https://doi.org/10.1145/3603498

Open-domain question answering (OpenQA) aims to answer questions without a prespecified domain. In recent years, OpenQA systems have often followed the **retriever-reader (R2)** architecture proposed by Chen et al. in 2017 [5]. The R2 architecture utilizes a simplified two-stage framework. First, a retriever selects a small subset of relevant passages from a large candidate pool (e.g., millions of Wikipedia passages). Then, a machine reader thoroughly examines the retrieval results and identifies or generates the correct answer. Generally, in the first stage, retrieval is implemented to retrieve the first 1,000 or 100 passages from the candidate pool and send them to the reader. However, people tend to look for answers in the smaller number of passages, e.g., 10 passages. In addition, processing too many passages can affect the efficiency of the OpenQA system. To solve this problem, some researchers propose a **retriever-reranker-reader (R3)** architecture to rerank the initial retrieval results. This paper focuses on improving retrieval efficiency, leading to improved QA performance.

In this paper, we propose **Query Context Expansion-based Retrieval (QCER)**.[1] This effective and simple retrieval method associates a query with domain information by adding contextual association information, thus speeding up the convergence of the query and bringing continuous improvements to passage reading and passage reranking tasks. We implemented QCER by using reader predictions as expansion terms for the initial query, further retrieving new relevant passages based on the expanded query. QCER uses a dense reader to extract top-n reader predictions from retrieval results as expansion terms for query expansion.

Our inspiration comes from the fact that humans are accustomed to associating prior knowledge with the query, such as the query's background knowledge, meanings, domains and contextual information when answering questions. However, for a machine that lacks cognition and common sense, a query is merely a simple combination of some words. Although we can use many approaches, such as **query expansion (QE)**, to enrich the semantic information of a query, it is still insufficient for retrieving the most relevant passages [21, 38, 48] from a large candidate pool (more discussion is available in Section 2). From the vector space perspective, the retriever obtains retrieval results by acquiring the relevant passages in the candidate passage vector space with the shortest distances from the query vector. However, the retrieval process does not consider the contextual relationship between the query vector space and candidate passage vector space, which makes the two vector spaces relatively isolated and unrelated. QCER considers reader predictions as the context of the query, which theoretically exists in the candidate passages, thereby connecting the query vector space and the candidate passage vector space. As a result, QCER restricts the domain scope of the retrieved passages and speeds up the retrieval process, enabling the retriever to find the correct relevant passages as quickly as possible.

QCER does a good job of bringing passages that might contain answers to the front of the retrieval results. When the number of retrieved passages is small, e.g., 10 passages, the top-k retrieval accuracy obtained by expanded queries is higher than that of the initial query. This means the reader does not need to equally consider a large number of retrieved passages (e.g., 100 or more passages), which could be computationally expensive. Intuitively, the top predictions of the reader are closely related to the ground-truth answer. Even if the predicted answers are partially correct or incorrect, they may still provide valuable signals for suggesting which passages contain the correct answer [24]. The better the performance of the reader, the higher quality of the expansion terms, and the higher retrieval accuracy of QCER. Moreover, the essence of the QCER method lies in utilizing the reader's predictions as expansion terms for the initial query, which is then followed by a second retrieval and reading process. In theory, the QCER method is independent of language type and model structure. If a different language is employed, the QCER method

[1]Our code is available at https://github.com/XY2323819551/QCER_for_OpenQA.git.

remains applicable, with the only variation being the data itself. Should relevant experiments be conducted in other languages, this approach can be explored and may lead to promising results.

We conduct experiments on the **Natural Questions (NQ)** [15] and TriviaQA (Trivia) [13] datasets. We demonstrate that the R2 architecture with QCER, without additional training or fine-tuning, achieves better **exact match (EM)** gains over the initial QA system when the reader takes fewer relevant passages as input. We also rerank the initial retrieval results using the new reader predictions obtained by QCER. We use the same approach as RIDER [24], i.e., reranking the initial retrieval results solely based on their lexical overlaps with the top predictions of the reader. We achieve better performance than RIDER [24] because of better reader predictions, which represent better ranking signals.

Contributions. (1) We propose QCER, which associates the given query with domain information by adding contextual association information to the query, thus speeding up the retrieval process and continuously improving the QA performance. QCER is a simple and effective **pseudo-relevance feedback (PRF)** method for passage retrieval of OpenQA that can be easily applied to existing R2- or R3-based OpenQA systems to achieve performance improvements without additional training or fine-tuning. (2) We demonstrate that the expanded query can retrieve passages at the top that are more relevant than the initial query, which enables the reader to find the desired answer as soon as possible. We also use new reader predictions obtained by QCER to improve passage reranking task. (3) Notably, QCER achieves performance comparable to or better than state-of-the-art methods in the three tasks of passage retrieval, passage reading, and passage reranking on two benchmark datasets under extractive OpenQA setup when we use a small relevant passage subset (e.g., 10 passages).

2 RELATED WORK

The retrieval process in OpenQA is typically implemented using **term frequency-inverse document frequency (TF-IDF)** or BM25 [35]. Multiple works [10, 34] have recently shown that retrieval systems based entirely on dense representations [52, 53] and approximate nearest neighbor methods are competitive with traditional approaches. Such models can be trained using weak supervision or pretrained using a cloze task and fine-tuned in an end-to-end manner [8, 17]. In 2020, Karpukhin et al. [14] combined sparse retrieval and dense retrieval by computing the linear combination of their respective scores, yielding better performance than that of dense and sparse retrieval individually. Recent studies on OpenQA systems [18, 23, 27] have improved their reading stages and have directly reused DPR [14], which is a dual-encoder architecture model proposed by Karpukhin et al. 2020 [14], as their retrievers. We also use DPR as our retriever, which is implemented by Pyserini [20]. The above methods focus on one step retrieve-and-read OpenQA datasets such as NQ and Trivia, which only need a single hop. However, for open-domain multi-hop problems like HotpotQA, we need more sophisticated reasoning and iterative approaches to solve them, such as Golden Retrieval [32], AISO [54], and MDR [45].

Reranking is a popular method for improving retrieval accuracy by ranking the most relevant passages at the top. Reranking is widely used in information retrieval [30, 33] and was explored in early OpenQA systems [16, 41, 42]. A reranker is a practical component for refining the initial retrieval results, which can improve the retrieval accuracy of the retriever at top positions and allow the reader to achieve comparable performance with fewer input passages. Early efforts on passage reranking for OpenQA used supervised learning [16] or **reinforcement learning (RL)** [41] based on **bidirectional long short-term memory (BiLSTM)**, and current works typically employ **bidirectional encoder representations from transformer (BERT)**-based cross-encoders [30, 33]. However, the training processes of cross-encoders are rather costly. RIDER [24] utilizes downstream signals (i.e., the reader predictions) to rerank the given passages without any training

steps, making it a simple and powerful method. We can perform passage reranking in the same way as RIDER, the main difference is that we use reader predictions extracted from the new retrieval results instead of the initial one. Our approach performs better in sparse and hybrid (BM25+DPR) retrieval scenarios than RIDER because of better ranking signals.

QE is a technique to improve the retrieval process [3, 36] by expanding the initial query with relevant contexts (terms). Most traditional research has been done QE in vector space, probabilistic, and language modeling [22, 50] retrieval models. Besides, the use of additional information, such as Wikipedia or Wikidata graphs [27, 41], WordNet (Miller 1995) [25], and ConceptNet (Speer et al. 2018) [37], as expansion contexts for queries is widespread as well [2]. But these methods may produce topic drift, causing the initial query to deviate from the original semantics. RL and paraphrase generation are also practical approaches for improving retrieval performance when applied to QE. However, these methods are not easily applicable or efficient enough for OpenQA because (1) they require external resources such as paraphrase data [49] to form the reformulated queries, and (2) they involve a time-consuming training process, such as RL [21, 29, 43].

In addition, PRF is considered an effective QE approach that uses information extracted from top-ranked passages to expand queries. As the field has shifted to transformer-based models [9, 19, 39, 40, 47, 51], more and more approaches are combining neural networks and PRF methods to obtain better OpenQA results. PGT [47], a graph-based Transformer that sparsifies attention between graph nodes to enable PRF while avoiding the high computational complexity of most Transformer architectures. Specifically, PGT achieves marginal improvements in reranking and retrieval. Zheng et al. [51] presented BERT-QE, a three-phase BERT-based reranker. Although BERT-QE achieves significant improvements in effectiveness over BERT reranker, it requires 11.01x more computations than BERT [19], making it computationally infeasible in many practical applications; Li et al. [19] investigate how to integrate PRF directly with emergent deep language models, and addresses this gap by investigating methods for integrating PRF signals with rerankers and dense retrievers based on deep language models. However, these methods has not been validated on OpenQA datasets, like NQ and Trivia. Some recent studies, such as those of Nogueira et al. [31] and Mao et al. [23], have expanded the queries with knowledge stored in PLMs and have produced good results.

QCER combines BERT and PRF, specifically, QCER uses a BERT-based dense reader to extract useful information from the top ranked relevant passages. QCER can not only exploit the knowledge stored in BERT, but also improve the efficiency and effectiveness of the retrieval process by adding contextual information to the query, which in turn improves the performance of the question and answer system. QCER is a simple and effective plug-and-play method that works well in current R2 and R3 OpenQA systems without any fine-tuning and training process.

3 METHOD

In this section, we describe our QCER approach for OpenQA. As described in Figure 1, we assume that an OpenQA system with an R2 architecture is available. In the first iteration, we retrieve the top-1,000 support passages based on the initial query and send only the top-p (default p = 500) passages to the reader for inference to obtain the top-n (default n = 10) answer predictions. Next, we select the top-m answer spans from the top-n answer predictions as expansion terms and add them to the initial query to form an expanded query. In the second iteration, the expanded query is used to retrieve relevant passages again, and the reader performs inference as in the first iteration.

3.1 Task Formulation

We denote the initial retrieval results obtained by the initial query as R and the new retrieval results obtained by the expanded query as R'. We denote the top-n reader predictions obtained by

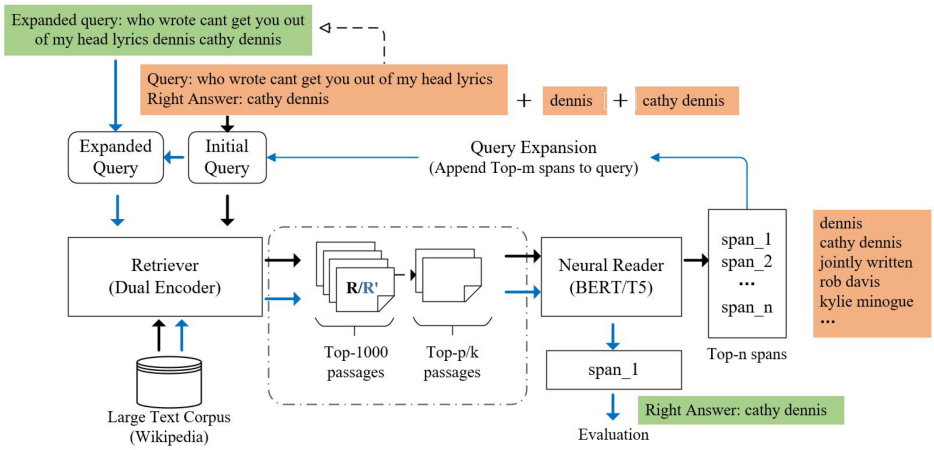

Fig. 1. The framework of QCER. The black lines represent the first iteration of QCER, and the blue lines denote the second iteration based on the top reader predictions of the first iteration. The main difference between the two iterations is that the second iteration uses expanded query to retrieve relevant passages instead of the initial query. In the above example, the value of m is equal to 2.

the top-p passages in R (denoted as $R[:p]$) as $A[:n]$. The goal of QCER is to retrieve R' by using the expanded query, which is formed by selecting the top-m predictions (denoted as $A[:m]$) from $A[:n]$ as expansion terms for the initial query, such that the retrieval efficiency is improved and better end-to-end QA performance is achieved when the reader takes $R'[:k]$ instead of $R[:k]$.

In addition, we obtain the top-n predictions (denoted as $A'[:n]$) when the reader takes $R'[:k]$ as input. We use the top-m predictions (denoted as $A'[:m]$) selected from $A'[:n]$ as reranking signals to rank R. Compared to RIDER [24], our approach achieves better performance.

3.2 Retriever and Reader

Retriever. In the retrieval stage, given a corpus $C = \{D_1, D_2, \ldots D_m\}$, the task for each query q is to return a list of the k most relevant passages (i.e., those most likely to contain the ground-truth answers) from C, where $k << C$. We use Pyserini (Lin et al. [20]) as our retriever. Pyserini is an IR toolkit that supports sparse retrieval (i.e., BM25) via integration with another toolkit called Anserini (Yang et al. [46]), which is built on Lucene with parameters of $b = 0.4$ and $k_1 = 0.9$. Additionally, Pyserini supports dense retrieval via integration with Facebook's Faiss Library [12]. The dense retrieval process involves two independent BERT (Devlin et al. 2019) [7] networks (base and uncased) and takes the representation of the $[CLS]$ token as the output. A dense passage encoder is present, along with a different query encoder that maps the input passages and queries to a d-dimensional vector space as follows:

$$q^* = BERT_Q(q), D_j^* = BERT_D(D_j) \tag{1}$$

During the inference, the relevance score of a passage for a query $Sim(q, D_j)$ is computed by the dot product operation:

$$Sim(q, D_j) = BERT_Q(q)^\top \cdot BERT_D(D_j) \tag{2}$$

This approach supports hybrid retrieval as well by combining dense and sparse retrieval by fusion factor α, $BM25(q, D_j)$ represents the relevance score of a passage for a query based on the BM25 method:

$$Sim(q, D_j) + \alpha \cdot BM25(q, D_j) \tag{3}$$

Reader. Given the top-k retrieved passages (up to 500 passages in our experiments), the reader uses BERT-base for representation learning, where it assigns a passage relevance score to each passage based on the observed [*CLS*] tokens and estimates span relevance scores for each candidate span based on the representations of its start and end tokens.

Regarding the extraction setup, we completely follow the design and implementation of the extractive readers in GAR [23] and PyGaggle [31], which use the passage-level span voting and evidence fusion technique. Let $D = [d_1, d_2, \ldots d_i, \ldots d_k]$ denote the list of retrieved passages with their passage relevance scores in $\vec{D}$. Let $S_i = [s_1, s_2, \ldots s_j \ldots, s_n]$ denote the top-n text spans in passage d_i ranked by their span relevance scores $\vec{S_i}$. Finally, the predictions of the spans in the same surface are aggregated from different retrieved passages, namely, via the normalized answer span scoring technique. PyGaggle [31] uses the evidence fusion technique, it fuses $\vec{D}$ with $\vec{R}$ by $\beta \cdot \vec{D} + \gamma \cdot \vec{R_i}$, where $\vec{R}$ denotes the passage retrieval scores obtained from the retriever for retrieved passages. Depending on the elected retrieval method, $\vec{R}$ can contain the sparse retrieval scores, the dense retrieval scores, or the hybrid retrieval scores. These final fused scores replace $\vec{D}$ as the relevance scores for all contexts in the answer span scoring step. For example, with evidence fusion, the answer span scoring technique of Mao et al. (2020) [23] becomes:

$$p(S_i[j]) = softmax(\beta \cdot \vec{D} + \gamma \cdot \vec{R_i})[i] \times softmax(\vec{S_i})[j] \tag{4}$$

We refer to this reader as GAR*.

3.3 QCER

Given initial retrieval results R and top-n reader predictions $A[:n]$, QCER improves the performance of the QA system as follows. QCER selects the top-m predictions of the reader $A[:m]$ from $A[:n]$, where we set n = 10 and m = 1,5 in our experiment. We use $A[:m]$ as the expansion term(s) for the initial query to form the expanded query. Then we retrieve the new retrieval results R' by using the expanded query based on the same retriever employed during the first iteration. Finally, we select the top-k passages from R' as the input of the reader to obtain predictions $A'[:n]$ and observe the performance of the QA system. We can infer that $A'[:n]$ is of higher quality than $A[:n]$ because QCER obtains significant gains during the retrieval stage and consequently benefits the QA system. Based on this, we can use $A'[:m]$ selected from $A'[:n]$ instead of $A[:m]$ to rerank R with the same method as that used in RIDER [24]. Specifically, we can scan R from the beginning of the list and append every passage $p \in R$ to R if p contains any reader prediction $a \in A'[:m]$ after string normalization (removing articles and punctuation) and tokenization. Then, the remaining passages are appended to R according to their original order. It is not surprising that our results are superior to RIDER due to the higher-quality ranking information.

Intuitively, if reader predictions are perfect, the retrieval accuracy after the second retrieval step is guaranteed to be optimal since reader predictions provide the query with the correct context information. If the reader prediction is incorrect, QCER can still be better (if the predicted answer co-occurs with the correct answer), or worse (if the predicted answer is misleading). If the reader performs reasonably well, QCER will likely retrieve relevant passages effectively. Overall, we quantitatively observe that QCER leads to consistent gains in terms of both retrieval accuracy and QA performance without refining the retriever or reader.

Drawbacks: Retrieval with Expanded Queries. We observed that conducting retrieval with the expanded query is not always effective because (1) some reader predictions are somewhat irrelevant, and (2) a query containing the correct answer may retrieve false positive passages with unrelated contexts containing the answer. Such low-quality passages may lead to potential issues in the subsequent passage reading stage.

Model Generalizability. Most of approaches can rarely work across various systems, but QCER has strong generalizability and can be applied to all kinds of two- or three-stage (R2 or R3) OpenQA systems, regardless of the model architectures of the retriever and reader.

4 EXPERIMENTAL SETUP

4.1 Datasets

We conduct experiments on the open-domain versions of two widely used QA benchmarks: **Natural Questions (NQ)** [15] and **TriviaQA (Trivia)** [13], corresponding to [14]. The open-domain version of NQ was designed for end-to-end QA by discarding answers with more than five tokens. The questions are real Google search queries, and the answers are spans in Wikipedia articles identified by annotators. TriviaQA contains a set of trivia questions with answers gathered from trivia and quiz league websites. The unfiltered version of TriviaQA is used for OpenQA. Each question in both datasets corresponds to an answer list. The question is correctly answered as long as the top-1 reader prediction appears in the answer list.

4.2 Evaluation Metrics

Following prior studies [14, 23], we use the **top-k retrieval accuracy** metric to evaluate the performance of the retriever and the **EM** metric to measure the performance of the reader. Top-k retrieval accuracy is the proportion of questions for which the top-k retrieved passages contain at least one answer span. EM is the proportion of the predicted answer spans being exactly the same as one of the ground-truth answers, after string normalization such as article and punctuation removal.

4.3 Sources of R and R'

We take the initial retrieval results of Pyserini [20] as R, including sparse-, dense-, and sparse-dense hybrid retrieval results obtained on NQ and Trivia, corresponding to R_{Sparse}, R_{Dense} and R_{Hybrid}, respectively. We select R_{Hybrid} as the source of R to produce expansion terms due to its higher retrieval performance. Therefore, the procedure that for obtaining R', including R'_{Sparse} and R'_{Hybrid}, in our experiments is as follows: for sparse retrieval, we regard R_{Hybrid} as the input of the reader and obtain the top-n answer candidates $A[:n]$. We select the top-m predictions from $A[:n]$ as expansion terms for the initial query. Then, we obtain new sparse retrieval results R'_{Sparse} by using the expanded query to conduct sparse retrieval again. We refer to this as QCER. For sparse-dense hybrid retrieval, we obtain new hybrid retrieval results R'_{Hybrid} by combining R'_{Sparse} with the initial dense retrieval results R_{Dense} retrieved by Pyserini. We refer to this as QCER+DPR.

4.4 Sources of $A[:n]$ and $A'[:n]$

To obtain the top-n predicted answers, we use the extractive reader named **GAR*** developed by Mao et al. [23] and Lin et al. [20] with the BERT-base representation, as described in Section 3.2. We send R (i.e., R_{Hybrid}) to the reader and obtain $A[:n]$; if we send R', namely, R'_{Sparse} or R'_{Hybrid}, we obtain $A'[:n]$. For the extractive reader, the top predictions are the text spans with the highest scores, and we set n = 10. QCER can also be applied to generative readers, which achieve better performance than extractive readers, and we leave this for future work.

4.5 Checkpoint

In summary, we use the implementations in Pyserini[2] and PyGaggle.[3] For the NQ dataset, we use RetrieverNQ and ReaderNQ-Single, which are trained only on the NQ dataset. For the Trivia dataset, we use RetrieverMulti and ReaderTQA-Multi; specifically, RetrieverMulti is trained on a combination of datasets (NQ, Trivia, webquestions, and CuratedTREC), and ReaderTQA-Multi is trained on Trivia with negative passages from the retrieval results yielded by RetrieverMulti.[4]

4.6 Hyperparameters

(1) Sparse-dense hybrid parameter α. In our experiments, the weights α for combining the initial dense retrieval results R'_{Sparse} and the new sparse retrieval results are fixed to 1.0 and 0.95 on NQ and Trivia, respectively. **(2) Evidence fusion parameters β and γ.** Following the implementation of GAR* in the PyGaggle library [31], we set the values of β (i.e., the weights for the relevance scores obtained from the reader) and γ (i.e., the weights for the retrieval scores obtained from the retriever) for the sparse retrieval results to 0.46 and 0.308 on NQ and 0.78 and 0.093 on Trivia, respectively. For the hybrid retrieval results, we set β and γ to 0.32 and 0.1952 on NQ and 0.76 and 0.152 on Trivia, respectively. **(3) Default setting of p.** We set up p = 500 in the first iteration for the following experiments except for those in Section 5.5 and Section 5.8. **(4) Default settings of n and m.** The hyperparameter n represents the number of reader predictions we save each time. We fix n = 10 to facilitate more experiments. Actually, our experiments only used the largest m = 5, so setting n = 5 is fine. We set m = 1,5 in Section 5.4 to investigate the effect of the number of reader predictions on the subsequent experiments. If we do not mark the value of m, the default setting is m = 1 in the first iteration.

We do not tune these hyperparameters very carefully. It would not be surprising to achieve better performance after deeply tuning these hyperparameters.

5 EXPERIMENT

We evaluate the effectiveness of QCER in three stages: the retrieval of relevant passages (Section 5.1), the reading of passages for OpenQA (Section 5.2), and the reranking of retrieval results (Section 5.3). We also explore how the value of m affects different tasks in Section 5.4, the relationship between the retrieval accuracy and the EM score of the R2 architecture in Section 5.5, and the best values of k and p for the proposed system in Section 5.6. In addition, we provide case studies in Section 5.7 to help our idea be easily understood. And we also conduct multiple iterations in Section 5.8 to explore when we will get the best results with QCER.

5.1 Passage Retrieval with QCER

We first evaluate the effectiveness of QCER for passage retrieval tasks. We compare QCER with the classic QE method BM25+RM3 [11], i.e., BM25 with QE (+RM3), to conduct a fair comparison. BM25+RM3 is a classic and commonly used combination for solving open domain problems and does not require external resources. BM25+RM3 and QCER are PRF methods, and both can be used to improve BM25 search results. In Table 1, for sparse retrieval, we show the top-k retrieval accuracy of BM25, BM25+RM3, DPR, GAR, and QCER. For hybrid retrieval, we report the top-k retrieval accuracy of BM25+DPR, GAR+DPR and QCER+DPR.

QCER outperforms vanilla BM25 and BM25+RM3 in sparse retrieval. BM25+RM3 shows marginal improvement over the vanilla BM25 and does not achieve performance comparable to that of QCER or GAR. In addition, QCER helps achieve sparse and hybrid retrieval accuracy comparable

[2] https://github.com/castorini/pyserini/blob/master/docs/experiments-dpr.md.
[3] https://github.com/castorini/pygaggle/blob/master/docs/experiments-dpr-reader.md.
[4] https://github.com/facebookresearch/DPR.

Table 1. Top-k Retrieval Accuracy Achieved on the Test Sets

	Method	NQ					Trivia				
		Top-5	Top-20	Top-100	Top-500	Top-1000	Top-5	Top-20	Top-100	Top-500	Top-1000
Sparse	BM25 (ours)	43.8	62.9	78.3	85.6	88.0	66.3	76.4	83.2	87.3	88.5
	BM25+RM3 [23]	44.6	64.2	79.6	86.8	88.9	67.0	77.1	83.8	87.7	88.9
	GAR [23]	60.9	**74.4**	**85.3**	**90.3**	**91.7**	73.1	80.4	**85.7**	**88.9**	**89.7**
	QCER	**64.3**	73.9	82.8	88.2	89.9	**75.8**	**80.6**	85.2	88.6	**89.7**
Hybrid	BM25+DPR (ours)	71.8	**82.6**	88.6	91.9	92.3	76.0	82.6	86.6	89.1	89.9
	GAR+DPR [23]	70.7	81.6	**88.9**	**92.0**	**93.2**	76.0	82.1	86.6	-	-
	QCER+DPR	**72.6**	82.2	88.3	91.5	92.5	**78.1**	**83.4**	**87.0**	**89.7**	**90.4**

We evaluated the baseline methods BM25, DPR, and BM25+DPR by ourselves using Pyserini [20].

Table 2. End-to-End QA Comparison among the State-of-the-art Methods in EM

	Method	NQ	Trivia
Extractive	Hard EM [26]	28.1	50.9
	Path Retriever [1]	32.6	-
	ORQA [17]	33.3	45.0
	Graph Retriever [27]	34.5	56.0
	REALM [8]	40.4	-
	DPR [14]	41.5	57.9
Generative	GPT3 [4]	29.9	-
	T5 [34]	36.6	60.5
	SpanSeqGen [28]	42.2	-
	RAG [18]	44.5	56.1
	FID [10]	**51.4**	**67.6**
GAR*	BM25 (ours)	27.2	55.4
	QCER	40.1	60.1
	BM25+DPR (ours)	41.9	59.7
	QCER+DPR	**42.6**	**61.2**

Notably, we only select the top-10 most relevant passages (k = 10) from R' for each query as the input of the reader, while the other methods read 50, 100, or more passages. Default m = 1.

to or better than GAR when k is small. The value of k varies across datasets and retrieval methods. For example, as shown in Table 1, QCER outperforms GAR only at top-5 with a 3.4-point gain and QCER+DPR at top-5, 20 on NQ; QCER outperforms GAR at top-5,20 and QCER+DPR at top-5,20,100 on Trivia. Better hybrid results will be obtained if the fusion strategy is carefully refined. The experiment demonstrated that our method effectively ranks the relevant passages that may contain answers at the top.

5.2 Passage Reading with QCER

For passage reading task, we conduct a comparison with both extractive [6, 42, 44] and generative [10, 34] methods when equipping QCER with the GAR*(described in Section 3.2). We report QA performance for four cases, including BM25, QCER, BM25+DPR, and QCER+DPR.

We show the QA performance comparison between QCER methods and the state-of-the-art methods. As shown in Table 2, most methods read 50, 100, or more passages for each query and then compute EM score based on top-1 prediction. When we equip QCER/QCER+DPR with GAR*,

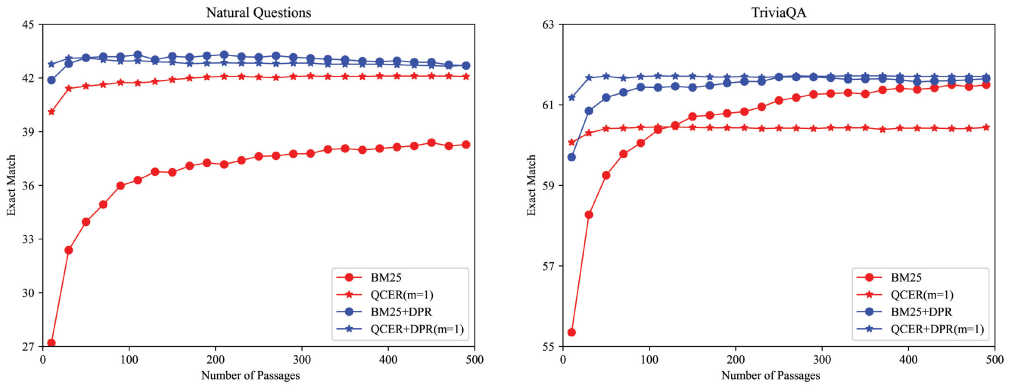

Fig. 2. End-to-end QA performance concerning a more detailed division of the number of retrieval passages. Default m = 1.

we only use the top-10 passages from R' for each query and obtain a comparable or even better performance than most existing methods taking more passages as input except for FID-large [10]. In addition, QCER/QCER+DPR can be regard as the augmented version of BM25/BM25+DPR. We observe that QCER/QCER+DPR exceeds BM25/BM25+DPR on both datasets for both retrieval methods (sparse retrieval and hybrid retrieval) when reader only takes 10 passages as input, even though they use the same extractive reader and without further model training. Specifically, QCER achieves EM = 40.1/60.1 on NQ/Trivia compared to the original EM = 27.2/55.4, QCER+DPR achieves EM = 42.6/61.2 on NQ/Trivia compared to the original EM = 41.9/59.7. Again, such results demonstrate that our approach can achieve improved retrieval efficiency, thereby providing higher-quality input for the reader and leading to finding the desired answer as soon as possible.

The QA performance concerning more cutoffs for k. Figure 2 shows the QA performance of a more detailed division of k. Although QCER performs relatively differently on the two datasets as k increases, the superiority of QCER can be seen when k is small. QCER outperforms vanilla BM25 across the board on NQ, but not on Trivia, especially when $k > 100$. The performance of QCER+DPR seems less desirable on NQ than BM25+DPR but performs well on Trivia. In addition, when the reader reads more passages, the performance of the QA system does not improve much because the more passages that are read, the more noise there is.

5.3 Passage Reranking with QCER

For reranking task, we take RIDER [24] as a comparison. We use the same method as RIDER. It is worth noting that the main difference between QCER reranking and RIDER is that QCER uses $A'[:m]$, which are extracted based on the new retrieval results R'_{Sparse}, as the reranking signals instead of $A[:m]$. In this reranking experiment, QCER(m=1) means we select the top-1 reader prediction as the reranking signal from $A'[:n]$ instead of $A[:n]$, and so does QCER(m=5).

To demonstrate that QCER can frontload relevant passages at the top of the retrieval results and yield continuous improvements to the QA system again, we use only the top-10 passages in R' as the input of the reader to obtain $A'[:m]$ as reranking signals for QCER and the top-100 passages in R as the input of the reader to obtain $A[:m]$ as reranking signals for RIDER.

As shown in Figure 3, QCER reranking works better than RIDER, whether m = 1 or m = 5. On NQ, the top-k retrieval accuracy of QCER is superior to that of RIDER when the number of relevant passages is small, and both methods achieve higher retrieval accuracy than vanilla BM25. On Trivia, a similar trend as that observed on NQ is found when m = 1. With m = 5, although the

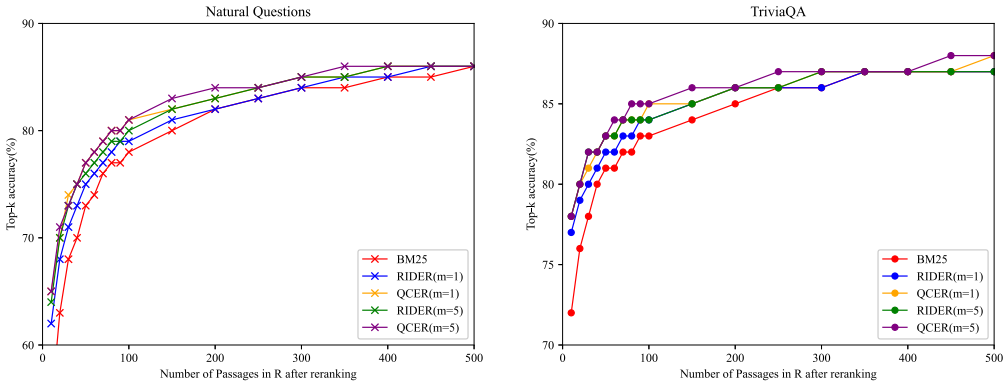

Fig. 3. Top-k retrieval accuracy of the initial retrieval results R after performing RIDER and QCER reranking with different values of m.

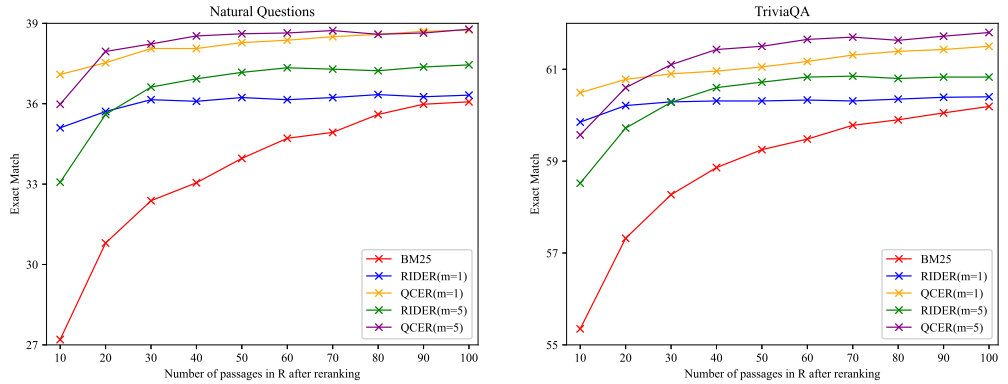

Fig. 4. End-to-end QA performance of RIDER and QCER reranking. The results are measured on the test sets of the two datasets.

performance of QCER is slightly worse than that of RIDER when the number of relevant passages is small, it is somewhat higher than that of RIDER when the number of relevant passages increases. The gaps among QCER, RIDER, and vanilla BM25 gradually decrease as the number of relevant passages increases.

Figure 4 illustrates the EM scores concerning the number of relevant passages in R after reranking. Similar EM score trends are observed on NQ and Trivia as the number of relevant passages increases. Both RIDER and QCER outperform vanilla BM25. QCER outperforms RIDER when sharing the same value of m because of the higher quality of the employed prediction signals ($A'[:m]$), even though QCER uses only the top-10 relevant passages to obtain reader predictions (i.e., reranking signals). With the same k and the same reranking method, QCER with m = 5 is better and consistent with the retrieval stage shown in Figure 3.

5.4 Impact of the Hyperparameter m on Different Tasks

In this part, we analyze the impact of hyperparameter m on three tasks: passage retrieval, passage reading, and passage reranking.

(1) Passage retrieval. Table 3 shows the top-k (k = 1, 5, 20, 100) retrieval accuracy of sparse retrieval and hybrid retrieval on both datasets when m takes different values. Here, we conduct

Table 3. Impact of the Hyperparameter m on Retrieval Accuracy

Method	NQ				Trivia			
	Top-1	Top-5	Top-20	Top-100	Top-1	Top-5	Top-20	Top-100
BM25 (ours)	22.1	43.8	63.0	78.3	46.3	66.3	76.4	83.2
QCER (m=1)	**48.9**	64.3	73.9	82.8	**67.0**	75.8	80.6	85.2
QCER (m=5)	48.6	**67.5**	**76.9**	**83.8**	64.4	**78.2**	**83.4**	**86.8**
BM25+DPR (ours)	48.3	71.8	**82.6**	**88.6**	60.9	76.0	82.6	86.6
QCER+DPR (m=1)	**56.9**	72.7	82.2	88.3	**69.1**	78.1	83.4	87.0
QCER+DPR (m=5)	53.7	**73.1**	82.0	87.7	67.1	**79.7**	**84.3**	**87.7**

We report the top-k retrieval accuracy on the NQ and Trivia test sets.

Table 4. Context Overlap between the Initial Retrieval Results R and New Retrieval Results R'_{Sparse} with GAR* on NQ and Trivia Test Sets

	NQ	Trivia
QCER (m=1)	54.02%	59.93%
QCER (m=5)	23.05%	24.05%

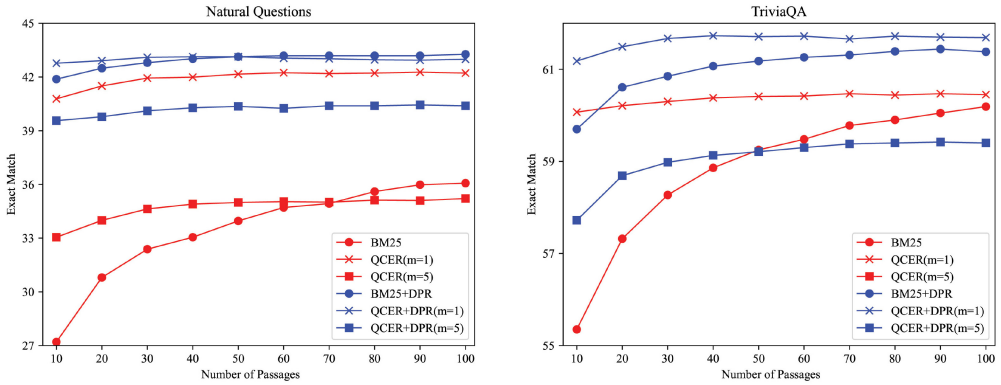

Fig. 5. Impact of the hyperparameter m on the QA system. End-to-end QA effectiveness (EM scores) with respect to the numbers of relevant passages (k) on NQ (left) and Trivia (right). We do not show the results of QCER with m = 5 on the Trivia dataset due to its low performance, below 55.0.

experiments with m = 1,5. In most cases, the retrieval performance of QCER(m=5) outperforms that of QCER(m=1) when k>=5, and vice versa when k = 1.

(2) Passage reading. As shown in Figure 5, we report the QA performance concerning k on two datasets for BM25 retrieval and hybrid retrieval. Notably, the EM score of QCER with m = 5 is lower than that of QCER with m = 1, which is inconsistent with the first retrieval stage. We argue that there are two main reasons for this phenomenon. First, while QCER with m = 5 is more likely to obtain the correct answer, it also introduces more noise. Second, instead of training the reader with new retrieval results, we directly use the checkpoint published by Ma et al. 2021 [20], which means that the reader does not learn the new knowledge in the new retrieval results. Table 4 shows the context overlap between the initial retrieval results R and new retrieval results R'_{Sparse} with GAR* on the NQ and Trivia test sets. The results are consistent with our conjecture: QCER (m=1) overlaps more with the initial retrieval results, and QCER (m=5) overlaps less with

Table 5. Top-k Retrieval Accuracy Achieved when p = 5, 10, 20, 50, 200, and 500

Method	NQ					Trivia				
	Top-5	Top-20	Top-100	Top-500	Top-1000	Top-5	Top-20	Top-100	Top-500	Top-1000
BM25 (ours)	43.77	62.94	78.25	85.57	88.01	66.28	76.41	83.15	87.34	88.50
QCER(p = 5)	63.85	73.46	82.80	88.14	89.81	74.14	79.60	84.52	88.24	89.45
QCER(p = 10)	64.49	73.68	83.13	88.23	89.81	74.98	80.10	84.73	88.28	89.53
QCER(p = 20)	64.63	74.13	**83.27**	**88.42**	89.94	75.41	80.31	84.95	88.44	89.63
QCER(p = 50)	**64.82**	**74.38**	**83.27**	88.39	**89.97**	75.66	80.51	84.99	88.39	89.60
QCER(p = 200)	64.60	73.88	82.94	88.28	89.92	75.82	**80.56**	85.14	88.50	89.68
QCER(p = 500)	64.29	73.85	82.77	88.23	89.89	**75.84**	80.55	**85.20**	**88.55**	**89.69**

Default m = 1.

the initial search results, indicating that QCER (m=5) contains more new knowledge. If we use the new retrieval results R' to retrain the reader, the EM score may be higher than those of QCER with m = 1. However, QCER(m=1) is indeed better than QCER (m=5) without additional training or fine-tuning.

In particular, vanilla BM25 obtained a 13%/6% improvement on NQ/Trivia when increasing the number of passages from 10 to 100. However, for QCER, there is no significant gap between the results obtained by selecting the top-10 passages or the top-100 passages. Even with the top-20 passages, we can achieve the same performance as that obtained with the top-100 passages, demonstrating that our QCER approach effectively frontloads the most relevant passages. So we can directly use top-10 to get a good EM. QCER(m=1) results in up to 12.9/5.7 EM gains on NQ/Trivia dataset over vanilla BM25 when only 10 passages are sent to the reader, which supports the notion that QCER can accelerate the convergence of the query. QCER+DPR(m=1) also outperforms the vanilla hybrid method BM25+DPR when k less than 50.

(3) Passage reranking. Regarding passage reranking task, as shown in Figure 4, the performance is consistent with that of the retrieval stage in Figure 3. QCER(m=5) has higher retrieval accuracy than QCER(m=1) and RIDER. Different from Section 5.1, where we retrieve the new relevant passages containing the new knowledge, we do not change the contexts of the initial retrieval results but rather rerank them.

5.5 The Relationship Between Retrieval Accuracy and the EM Scores

We cannot state that the performance of QCER(m=5) is necessarily worse than that of QCER with m = 1. After all, the reader has not been trained on the new retrieval results R'. However, if we fix the value of m, we can argue that the performance between the retrieval accuracy and the EM score is not necessarily consistent. In other words, higher retrieval accuracy does not necessarily lead to a higher EM score if we do not retrain the reader. We fix m = 1 and conduct more detailed experiments on BM25 retrieval in this section. Unlike the above experiments, we set p = 5, 10, 20, 50, 200, and 500 in the first iteration, generating different expansion terms, forming various expansion queries, and obtaining new sparse retrieval results R' with different top-k retrieval accuracy. Then, we send R' to the reader and record the corresponding changes of EM. As shown in Tables 5 and 6, retrieval accuracy is almost consistent with QA performance on Trivia, both reaching the optimum at p = 500. However, on NQ, QCER obtained the best retrieval accuracy at p = 50, and the best QA performance was presented when p = 500. In summary, improvements in the retrieval stage may not improve the QA performance.

5.6 The Best Values of k and p for the Proposed System

Table 6. The QA Performance Achieved when p = 5, 10, 20, 50, 200, and 500

Method	NQ				Trivia			
	Top-10	Top-20	Top-50	Top-100	Top-10	Top-20	Top-50	Top-100
BM25 (ours)	27.20	30.80	33.96	36.07	55.35	57.32	59.25	60.19
QCER(p = 5)	32.11	32.80	33.46	33.80	58.08	58.45	58.92	59.11
QCER(p = 10)	32.91	33.49	34.10	34.35	58.91	59.12	59.42	59.64
QCER(p = 20)	33.30	33.80	34.38	34.63	59.35	59.53	59.78	59.94
QCER(p = 50)	33.46	33.96	34.68	34.85	59.72	59.91	60.10	60.24
QCER(p = 200)	33.30	33.85	34.71	34.99	59.92	60.08	60.25	60.34
QCER(p = 500)	**40.11**	**40.94**	**41.55**	**41.75**	**60.07**	**60.21**	**60.41**	**60.45**

Default m = 1. The experiments use the GAR* reader, detailed in Section 3.2.

The k and p values will have different optimal values in different systems. Different quality reader models, retriever models, and datasets affect the choice of k and p values. For the value of k, top-k retrieval accuracy increases as the value of k increases. Researchers generally set value k up to 1,000, and we follow this setting. For the value of p, if we want good results, we can set p = 500. If we want to obtain faster inference, we can set p = 50, reducing model performance but speeding up the QA pipeline. We set the default p = 500 except for Tables 5 and 6. From the Table 5, we know that the retrieval accuracy performs well for p = 10, 20, 30, and 40, and is not even worse than p = 50. Why set p = 500? This is because considering the QA performance, the reader can achieve better EM at p = 500. So from the whole QA system, it is necessary to set p = 500 to get the best performance.

5.7 Case Studies

We list four cases in Table 7, representing four different types of cases. The first type of case: the expansion terms are correct, and the retrieval result contains the correct answer. The second type of case: the expansion terms are correct, but the retrieval result does not contain the correct answer. The third type of case: the expansion terms are incorrect, but the retrieval result contains the correct answer, and the fourth type of case: the expansion terms are incorrect, and the retrieval result does not contain the correct answer.

The "Old passage" field and "New passage" field in Table 7 refer to the retrieval results of the initial query and the extended query, respectively. Specifically, in the first and third types of cases, the correct answers are not found in the retrieval results before implementing QCER; however, they are included in the retrieval results after using QCER. The opposite was true in the second and fourth type of cases. Therefore, not all QEs can lead to improvement. In Tables 8 and 9, "Incorrect→Correct" represents the comparison of retrieval results before and after using QCER, mainly including the first and third types of cases; "Correct→Incorrect" follows the same logic, mainly including the second and fourth types of cases. Therefore, the final retrieval accuracy improvement is calculated as follows:

$$\frac{[num(case1) + num(case3)] - [num(case2) + num(case4)]}{num(total)} \tag{5}$$

where $num(case1)$ represents the number of samples in the first type of case, and so on. $num(total)$ represents the total number of samples in the current dataset.

For example, we want to know how QCER improves the top-5 retrieval results on NQ. In Table 8, we can see that in the retrieval results before and after using QCER, the proportion of incorrect-to-correct cases and correct-to-incorrect cases in the dataset accounts for the entire dataset 24.02%

Table 7. Four Types of Cases from NQ

Case 1	**Query**: when is the next deadpool movie being released **Old passage**: Deadpool's world premiere was held at the Grand Rex in Paris on February 8, 2016, before its initial theatrical release in Hong Kong the next day.... **Expansion terms**: may 18 2018 {['May 18, 2018']} **New query**: when is the next deadpool movie being released may 18 2018 **New passage**: ...score was released by Columbia Records on May 18, 2018, coinciding with the film's release. A soundtrack album covering...
Case2	**Query**: when is the last time the philadelphia won the superbowl **Old Passage**: Philadelphia's Super Bowl 52 win at the end of the 2017 season. **Expansion terms**: 2017 {['Super Bowl LII', '2017']} **New query**: when is the last time the philadelphia won the superbowl 2017 **New passage**: ...Superbowl of Wrestling The Superbowl of Wrestling was an event held in the 1970s. It was one of the first professional wrestling ""Supercards""...
Case3	**Query**: who got the first nobel prize in physics **Old Passage**: It is one of the five Nobel Prizes established by the will of Alfred Nobel in 1895 and awarded since 1901. **New query**: who got the first nobel prize in physics albert einstein **New passage**: ...Germany dominated the world of physics before 1933, led by Hermann von Helmholtz, Joseph von Fraunhofer, Daniel Gabriel Fahrenheit, Wilhelm Conrad Röntgen, Albert Einstein, Max Planck and Werner Heisenberg...
Case4	**Query**: who was the first lady nominated member of the rajya sabha **Old passage**: Mary Kom was nominated for the Rajya Sabha in 2016 along with former cricketer Navjot Singh Sidhu, Malayalam film actor Suresh Gopi, Narendra Jadhav and Swapan ... **Expansion terms**: rukmini devi arundale {['Mary Kom']} **New query**: who was the first lady nominated member of the rajya sabha rukmini devi arundale **New passage**: ...started in India, when Dr George Arundale invited Dr Maria Montessori to start courses in the 'Besant Theosophical High School' in 1939...Rukmini Devi was nominated as a member of the Indian Parliament's Council of States (the Rajya Sabha) in April 1952 and re-nominated in 1956. She was the first Indian woman to be nominated in Rajya.

The correct answers are highlighted in green, and the incorrect answers are highlighted in red. Ground-truth references are shown in the {braces}.

Table 8. The Proportion of Changes Accounts for the Entire Dataset in whether the Retrieval Results Contain Answers before and after using QCER

		Top-1	Top-5	Top-10	Top-20	Top-50	Top-100	Top-500	Top-1000
NQ	Incorrect→Correct	29.61	24.02	18.56	14.35	9.72	6.95	4.07	3.13
	Correct→Incorrect	2.83	3.49	4.07	3.43	2.96	2.44	1.41	1.25
Trivia	Incorrect→Correct	22.51	11.57	8.39	5.99	4.09	3.22	1.88	1.57
	Correct→Incorrect	1.79	2.01	1.89	1.85	1.41	1.17	0.67	0.37

Table 9. The Proportion of Two Kinds of Changes on Top-5 Retrieval Accuracy

Top-5 retrieval accuracy	Incorrect→Correct		Correct→Incorrect	
	Case1	Case3	Case2	Case4
NQ	19.73/24.02 (82.14%)	4.29/24.02 (17.86%)	0.25/3.49 (7.16%)	3.24/3.49 (92.84%)
Trivia	11.40/11.57 (98.53%)	0.17/11.57 (1.47%)	0.03/2.01 (1.49%)	1.98/2.01 (98.51%)

Table 10. Top-k Retrieval Accuracy and EM Score of Different Iterations on NQ

Method	Retrieval					Reading				
	Top-5	Top-10	Top-20	Top-50	Top-100	Top-5	Top-10	Top-20	Top-50	Top-100
Iteration1	43.77	54.46	62.94	72.74	78.25	22.60	27.20	30.80	33.96	36.07
Iteration2(p = 50)	**64.82**	**69.39**	**74.38**	**79.86**	**83.27**	32.99	33.46	33.96	34.68	34.85
Iteration2(p = 500)	64.29	68.95	73.85	79.50	82.77	**39.42**	**40.11**	**40.94**	**41.55**	**41.75**
Iteration3(p = 50)	63.60	67.76	71.77	76.23	79.61	29.86	31.30	31.55	32.24	32.60
Iteration3(p = 500)	62.44	66.98	71.00	75.46	78.81	29.72	31.16	31.36	31.97	32.38
Iteration4(p = 50)	61.99	65.82	69.14	73.91	77.09	28.03	28.39	29.61	30.30	30.89
Iteration4(p = 500)	60.69	64.74	68.25	73.16	76.29	27.81	28.17	29.53	30.22	30.80

and 3.49%, respectively. Therefore we can say that when using QCER, the top-5 retrieval accuracy can be improved by 20.53% on NQ. Positive feedback trends are shown on different values of k and different datasets.

From 9 we can know that the improvement brought by QCER is mainly brought by the first type of case and the third type of case, of which the first type of case is the main reason for the improvement, and the fourth type of case is the main reason for the decline. We experimented on top-5 retrieval accuracy in Table 9 and found that 82.14%/98.53% of the incorrect-to-correct improvement on NQ/Trivia was caused by the first type of case. A total of 92.84%/98.51% of the correct-to-incorrect decline on NQ/Trivia, respectively, was caused by the fourth type of case. The results of the experiment are consistent with our hypothesis.

In conclusion, although not all QEs are effective, OCER is still effective, given that our evaluation is based on the entire dataset. QCER can bring improvements based on the existing retriever model and reader model.

5.8 Multiple Iterations

We do not recommend multiple iterations because: (1) multiple iterations are too expensive. Each iteration needs to use the reader predictions produced by the previous iteration, so it can only be performed serially. (2) After many iterations, the query has been overextended, and the meaning may change compared with the original query. (3) Our retriever and reader were not trained on the expanded query and the new retrieval results, so the neural model did not learn new knowledge, resulting in performance degradation. Tables 10 and 11 are our supplementary experiments for BM25 retrieval on NQ and Trivia. It can be seen that the performance of the two datasets is consistent. The best results are still at the second iteration, whether the retrieval or the reading stage. The effect of the third and fourth iterations begins to decline.

The value of p refers to selecting top-p passages from the previous iteration to generate reader predictions as the expansion terms of this iteration, which has the same meaning as the value of p in the paper. The process of iteration 3 and iteration 4 is the same as described in the paper. Each query expansion is based on the query of the previous iteration. As the iteration increases, the query becomes longer.

Table 11. Top-k Retrieval Accuracy and EM Score of Different Iterations on Trivia

Method	Retrieval					Reading				
	Top-5	Top-10	Top-20	Top-50	Top-100	Top-5	Top-10	Top-20	Top-50	Top-100
Iteration1	66.28	71.74	76.41	80.56	83.15	52.11	55.35	57.32	59.25	60.19
Iteration2(p = 50)	75.66	77.95	80.51	83.09	84.99	59.37	59.72	59.91	60.10	60.24
Iteration2(p = 500)	**75.84**	**78.24**	**80.55**	**83.24**	**85.20**	**59.76**	**60.07**	**60.21**	**60.41**	**60.45**
Iteration3(p = 50)	75.04	77.05	79.02	81.68	83.57	56.75	57.39	57.72	58.13	58.18
Iteration3(p = 500)	74.86	77.04	79.12	81.85	83.84	57.01	57.60	58.05	58.45	58.48
Iteration4(p = 50)	74.27	76.20	78.20	80.58	82.30	53.39	54.15	54.74	54.98	55.14
Iteration4(p = 500)	74.17	76.22	78.30	80.62	82.52	53.73	54.45	55.09	55.39	55.52

6 CONCLUSION

In this work, we propose QCER, which associates a query with domain information by adding contextual association information, thus speeding up the process of finding relevant passages and improving the passage reading and passage reranking tasks. We demonstrate that the expanded query, formed by using reader predictions as expansion terms for the initial query, can effectively retrieve the most relevant passages at the top and make the reader find the desired answer as soon as possible. Remarkably, QCER is a simple passage retrieval method for OpenQA that does not involve additional training or fine-tuning and can be easily integrated into existing R2 or R3 systems to improve performance. QCER improves the retrieval accuracy, reranking accuracy, and QA performance on two benchmark datasets with fewer input passages. In future work, we plan to explore more simple and powerful query context expansion strategies for passage retrieval to further improve the performance of the OpenQA system.

REFERENCES

[1] Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over Wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=SJgVHkrYDH.

[2] Hiteshwar Kumar Azad and Akshay Deepak. 2019. A new approach for query expansion using Wikipedia and WordNet. *Inf. Sci.* 492 (2019), 147–163. DOI: https://doi.org/10.1016/j.ins.2019.04.019

[3] Hiteshwar Kumar Azad and Akshay Deepak. 2019. Query expansion techniques for information retrieval: A survey. *Inf. Process. Manag.* 56, 5 (2019), 1698–1735. DOI: https://doi.org/10.1016/j.ipm.2019.05.009

[4] Tom B. Brown, Benjamin Mann, and Nick Ryder. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[5] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, Regina Barzilay and Min-Yen Kan (Eds.). Association for Computational Linguistics, 1870–1879. DOI: https://doi.org/10.18653/v1/P17-1171

[6] Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 845–855. DOI: https://doi.org/10.18653/v1/P18-1078

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. DOI: https://doi.org/10.18653/v1/n19-1423

[8] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. *CoRR* abs/2002.08909 (2020). arXiv:2002.08909. https://arxiv.org/abs/2002.08909.

[9] Kai Hui, Ben He, Tiejian Luo, and Bin Wang. 2011. A comparative study of pseudo relevance feedback for ad-hoc retrieval. In *Advances in Information Retrieval Theory - Third International Conference, ICTIR 2011, Bertinoro, Italy, September 12-14, 2011. Proceedings (Lecture Notes in Computer Science)*, Giambattista Amati and Fabio Crestani (Eds.), Vol. 6931. Springer, 318–322. DOI: https://doi.org/10.1007/978-3-642-23318-0_30

[10] Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19-23, 2021*, Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty (Eds.). Association for Computational Linguistics, 874–880. DOI: https://doi.org/10.18653/v1/2021.eacl-main.74

[11] Nasreen Abdul Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. In *Proceedings of the Thirteenth Text REtrieval Conference, TREC 2004, Gaithersburg, Maryland, USA, November 16-19, 2004 (NIST Special Publication)*, Ellen M. Voorhees and Lori P. Buckland (Eds.), Vol. 500-261. National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec13/papers/umass.novelty.hard.pdf.

[12] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with GPUs. *IEEE Trans. Big Data* 7, 3 (2021), 535–547. DOI: https://doi.org/10.1109/TBDATA.2019.2921572

[13] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, Regina Barzilay and Min-Yen Kan (Eds.). Association for Computational Linguistics, 1601–1611. DOI: https://doi.org/10.18653/v1/P17-1147

[14] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 6769–6781. DOI: https://doi.org/10.18653/v1/2020.emnlp-main.550

[15] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Trans. Assoc. Comput. Linguistics* 7 (2019), 452–466. https://transacl.org/ojs/index.php/tacl/article/view/1455.

[16] Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking paragraphs for improving answer recall in open-domain question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, 565–569. DOI: https://doi.org/10.18653/v1/d18-1053

[17] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 6086–6096. DOI: https://doi.org/10.18653/v1/p19-1612

[18] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

[19] Hang Li, Ahmed Mourad, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. 2021. Pseudo relevance feedback with deep language models and dense retrievers: Successes and pitfalls. *CoRR* abs/2108.11044 (2021). arXiv:2108.11044. https://arxiv.org/abs/2108.11044.

[20] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: An easy-to-use Python toolkit to support replicable IR research with sparse and dense representations. *CoRR* abs/2102.10073 (2021). arXiv:2102.10073. https://arxiv.org/abs/2102.10073.

[21] Ye Liu, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip S. Yu. 2019. Generative question refinement with deep reinforcement learning in retrieval-based QA system. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 1643–1652. DOI: https://doi.org/10.1145/3357384.3358046

[22] Yuanhua Lv and ChengXiang Zhai. 2014. Revisiting the divergence minimization feedback model. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, Jianzhong Li, Xiaoyang Sean Wang, Minos N. Garofalakis, Ian Soboroff, Torsten Suel, and Min Wang (Eds.). ACM, 1863–1866. DOI: https://doi.org/10.1145/2661829.2661900

[23] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Generation-augmented retrieval for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, 4089–4100. DOI: https://doi.org/10.18653/v1/2021.acl-long.316

[24] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Reader-guided passage reranking for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021 (Findings of ACL)*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.), Vol. ACL/IJCNLP 2021. Association for Computational Linguistics, 344–350. DOI: https://doi.org/10.18653/v1/2021.findings-acl.29

[25] George A. Miller. 1995. WordNet: A lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41. DOI: https://doi.org/10.1145/219717.219748

[26] Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard EM approach for weakly supervised question answering. 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 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 2851–2864. DOI: https://doi.org/10.18653/v1/D19-1284

[27] Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering. *CoRR* abs/1911.03868 (2019). arXiv:1911.03868. http://arxiv.org/abs/1911.03868.

[28] Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 5783–5797. DOI: https://doi.org/10.18653/v1/2020.emnlp-main.466

[29] Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, 574–583. DOI: https://doi.org/10.18653/v1/d17-1061

[30] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with BERT. *CoRR* abs/1901.04085 (2019). arXiv:1901.04085. http://arxiv.org/abs/1901.04085.

[31] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020 (Findings of ACL)*, Trevor Cohn, Yulan He, and Yang Liu (Eds.), Vol. EMNLP 2020. Association for Computational Linguistics, 708–718. DOI: https://doi.org/10.18653/v1/2020.findings-emnlp.63

[32] Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering complex open-domain questions through iterative query generation. 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 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 2590–2602. DOI: https://doi.org/10.18653/v1/D19-1261

[33] Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of BERT in ranking. *CoRR* abs/1904.07531 (2019). arXiv:1904.07531. http://arxiv.org/abs/1904.07531.

[34] Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model?. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, 5418–5426. DOI: https://doi.org/10.18653/v1/2020.emnlp-main.437

[35] Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.* 3, 4 (2009), 333–389. DOI: https://doi.org/10.1561/1500000019

[36] Dilip Kumar Sharma, Rajendra Pamula, and D. S. Chauhan. 2021. Semantic approaches for query expansion. *Evol. Intell.* 14, 2 (2021), 1101–1116. DOI: https://doi.org/10.1007/s12065-020-00554-x

[37] Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-first AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco,*

California, USA, Satinder P. Singh and Shaul Markovitch (Eds.). AAAI Press, 4444–4451. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972.

[38] Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question rewriting for conversational question answering. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*, Liane Lewin-Eytan, David Carmel, Elad Yom-Tov, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 355–363. DOI: https://doi.org/10.1145/3437963.3441748

[39] Junmei Wang, Min Pan, Tingting He, Xiang Huang, Xueyan Wang, and Xinhui Tu. 2020. A pseudo-relevance feedback framework combining relevance matching and semantic matching for information retrieval. *Inf. Process. Manag.* 57, 6 (2020), 102342. DOI: https://doi.org/10.1016/j.ipm.2020.102342

[40] Le Wang, Ze Luo, Canjia Li, Ben He, Le Sun, Hao Yu, and Yingfei Sun. 2020. An end-to-end pseudo relevance feedback framework for neural document retrieval. *Inf. Process. Manag.* 57, 2 (2020), 102182. DOI: https://doi.org/10.1016/j.ipm.2019.102182

[41] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R^3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of the Thirty-second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 5981–5988. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16712.

[42] Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018. Evidence aggregation for answer re-ranking in open-domain question answering. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=rJl3yM-Ab.

[43] Xiao Wang, Craig MacDonald, and Iadh Ounis. 2020. Deep reinforced query reformulation for information retrieval. *CoRR* abs/2007.07987 (2020). arXiv:2007.07987. https://arxiv.org/abs/2007.07987.

[44] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. 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 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 5877–5881. DOI: https://doi.org/10.18653/v1/D19-1599

[45] Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick S. H. Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2021. Answering complex open-domain questions with multi-hop dense retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=EMHoBG0avc1.

[46] Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the use of Lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White (Eds.). ACM, 1253–1256. DOI: https://doi.org/10.1145/3077136.3080721

[47] HongChien Yu, Zhuyun Dai, and Jamie Callan. 2021. PGT: Pseudo relevance feedback using a graph-based transformer. In *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part II (Lecture Notes in Computer Science)*, Djoerd Hiemstra, Marie-Francine Moens, Josiane Mothe, Raffaele Perego, Martin Potthast, and Fabrizio Sebastiani (Eds.), Vol. 12657. Springer, 440–447. DOI: https://doi.org/10.1007/978-3-030-72240-1_46

[48] Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul N. Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 1933–1936. DOI: https://doi.org/10.1145/3397271.3401323

[49] Salah Zaiem and Fatiha Sadat. 2019. Sequence to sequence learning for query expansion. In *The Thirty-third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-first Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 10075–10076. DOI: https://doi.org/10.1609/aaai.v33i01.330110075

[50] ChengXiang Zhai and John D. Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management, Atlanta, Georgia, USA, November 5-10, 2001*. ACM, 403–410. DOI: https://doi.org/10.1145/502585.502654

[51] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. 2020. BERT-QE: Contextualized query expansion

for document re-ranking. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020 (Findings of ACL)*, Trevor Cohn, Yulan He, and Yang Liu (Eds.), Vol. EMNLP 2020. Association for Computational Linguistics, 4718–4728. DOI: https://doi.org/10.18653/v1/2020.findings-emnlp.424

[52] Wenhao Zhu, Xin Jin, Shuang Liu, Zhiguo Lu, Wu Zhang, Ke Yan, and Baogang Wei. 2020. Enhanced double-carrier word embedding via phonetics and writing. *ACM Trans. Asian Low Resour. Lang. Inf. Process.* 19, 2 (2020), 20:1–20:18. DOI: https://doi.org/10.1145/3344920

[53] Wenhao Zhu, Shuang Liu, and Chaoming Liu. 2021. Learning multimodal word representation with graph convolutional networks. *Inf. Process. Manag.* 58, 6 (2021), 102709. DOI: https://doi.org/10.1016/j.ipm.2021.102709

[54] Yunchang Zhu, Liang Pang, Yanyan Lan, Huawei Shen, and Xueqi Cheng. 2021. Adaptive information seeking for open-domain question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7–11 November, 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics, 3615–3626. DOI: https://doi.org/10.18653/v1/2021.emnlp-main.293

Received 3 March 2022; revised 25 April 2023; accepted 1 June 2023