Generation-Augmented Retrieval for Open-Domain Question Answering

Yuning Mao[1]*, **Pengcheng He**[2], **Xiaodong Liu**[3], **Yelong Shen**[2],
Jianfeng Gao[3], **Jiawei Han**[1], **Weizhu Chen**[2]
[1]University of Illinois, Urbana-Champaign [2]Microsoft Azure AI [3]Microsoft Research
[1]{yuningm2, hanj}@illinois.edu
[2,3]{penhe, xiaodl, yeshe, jfgao,wzchen}@microsoft.com

arXiv:2009.08553v4 [cs.CL] 6 Aug 2021

Abstract

We propose Generation-Augmented Retrieval (GAR) for answering open-domain questions, which augments a query through text generation of heuristically discovered relevant contexts without external resources as supervision. We demonstrate that the generated contexts substantially enrich the semantics of the queries and GAR with sparse representations (BM25) achieves comparable or better performance than state-of-the-art dense retrieval methods such as DPR (Karpukhin et al., 2020). We show that generating diverse contexts for a query is beneficial as fusing their results consistently yields better retrieval accuracy. Moreover, as sparse and dense representations are often complementary, GAR can be easily combined with DPR to achieve even better performance. GAR achieves state-of-the-art performance on Natural Questions and TriviaQA datasets under the extractive QA setup when equipped with an extractive reader, and consistently outperforms other retrieval methods when the same generative reader is used.[1]

1 Introduction

Open-domain question answering (OpenQA) aims to answer factoid questions without a pre-specified domain and has numerous real-world applications. In OpenQA, a large collection of documents (*e.g.*, Wikipedia) are often used to seek information pertaining to the questions. One of the most common approaches uses a retriever-reader architecture (Chen et al., 2017), which first retrieves a small subset of documents *using the question as the query* and then reads the retrieved documents to extract (or generate) an answer. The retriever is crucial as it is infeasible to examine every piece of information in the entire document collection (*e.g.*, millions of Wikipedia passages) and the retrieval accuracy bounds the performance of the (extractive) reader.

Early OpenQA systems (Chen et al., 2017) use classic retrieval methods such as TF-IDF and BM25 with sparse representations. Sparse methods are lightweight and efficient, but unable to perform semantic matching and fail to retrieve relevant passages without lexical overlap. More recently, methods based on dense representations (Guu et al., 2020; Karpukhin et al., 2020) learn to embed queries and passages into a latent vector space, in which text similarity beyond lexical overlap can be measured. Dense retrieval methods can retrieve semantically relevant but lexically different passages and often achieve better performance than sparse methods. However, the dense models are more computationally expensive and suffer from information loss as they condense the entire text sequence into a fixed-size vector that does not guarantee exact matching (Luan et al., 2020).

There have been some recent studies on query reformulation with text generation for other retrieval tasks, which, for example, rewrite the queries to context-independent (Yu et al., 2020; Lin et al., 2020; Vakulenko et al., 2020) or well-formed (Liu et al., 2019) ones. However, these methods require either task-specific data (*e.g.*, conversational contexts, ill-formed queries) or external resources such as paraphrase data (Zaiem and Sadat, 2019; Wang et al., 2020) that cannot or do not transfer well to OpenQA. Also, some rely on time-consuming training process like reinforcement learning (RL) (Nogueira and Cho, 2017; Liu et al., 2019; Wang et al., 2020) that is not efficient enough for OpenQA (more discussions in Sec. 2).

In this paper, we propose Generation-Augmented Retrieval (GAR), which augments a query through text generation of a pre-trained language model (PLM). Different from prior studies that reformulate queries, GAR does not require external resources or downstream feedback via RL as supervision, because it does not *rewrite* the query but *expands* it with heuristically discov-

*Work was done during internship at Microsoft Azure AI.

[1]Our code and retrieval results are available at https://github.com/morningmoni/GAR.

ered relevant contexts, which are fetched from PLMs and provide richer background information (Table 2). For example, by prompting a PLM to generate the title of a relevant passage given a query and appending the generated title to the query, it becomes easier to retrieve that relevant passage. Intuitively, the generated contexts explicitly express the search intent not presented in the original query. As a result, GAR with sparse representations achieves comparable or even better performance than state-of-the-art approaches (Karpukhin et al., 2020; Guu et al., 2020) with dense representations of the original queries, while being more lightweight and efficient in terms of both training and inference (including the cost of the generation model) (Sec. 6.4).

Specifically, we expand the query (question) by adding relevant contexts as follows. We conduct seq2seq learning with the question as the input and various freely accessible in-domain contexts as the output such as *the answer, the sentence where the answer belongs to*, and *the title of a passage that contains the answer*. We then append the generated contexts to the question as the *generation-augmented query* for retrieval. We demonstrate that using multiple contexts from diverse generation targets is beneficial as fusing the retrieval results of different generation-augmented queries consistently yields better retrieval accuracy.

We conduct extensive experiments on the Natural Questions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Trivia) (Joshi et al., 2017) datasets. The results reveal four major advantages of GAR: (1) GAR, combined with BM25, achieves significant gains over the same BM25 model that uses the original queries or existing unsupervised query expansion (QE) methods. (2) GAR with sparse representations (BM25) achieves comparable or even better performance than the current state-of-the-art retrieval methods, such as DPR (Karpukhin et al., 2020), that use dense representations. (3) Since GAR uses sparse representations to measure lexical overlap[2], it is complementary to dense representations: by fusing the retrieval results of GAR and DPR (denoted as GAR$^+$), we obtain consistently better performance than either method used individually. (4) GAR outperforms DPR in the end-to-end QA performance (EM) when the same extractive reader is used: EM=41.8 (43.8 for GAR$^+$) on NQ and 62.7 on Trivia, creating new state-of-the-art results for extractive OpenQA. GAR also outperforms other retrieval methods under the generative setup when the same generative reader is used: EM=38.1 (45.3 for GAR$^+$) on NQ and 62.2 on Trivia.

Contributions. (1) We propose Generation-Augmented Retrieval (GAR), which augments queries with heuristically discovered relevant contexts through text generation without external supervision or time-consuming downstream feedback. (2) We show that using generation-augmented queries achieves significantly better retrieval and QA results than using the original queries or existing unsupervised QE methods. (3) We show that GAR, combined with a simple BM25 model, achieves new state-of-the-art performance on two benchmark datasets in extractive OpenQA and competitive results in the generative setting.

2 Related Work

Conventional Query Expansion. GAR shares some merits with query expansion (QE) methods based on pseudo relevance feedback (Rocchio, 1971; Abdul-Jaleel et al., 2004; Lv and Zhai, 2010) in that they both expand the queries with relevant contexts (terms) without the use of external supervision. GAR is superior as it expands the queries with knowledge stored in the PLMs rather than the retrieved passages and its expanded terms are learned through text generation.

Recent Query Reformulation. There are recent or concurrent studies (Nogueira and Cho, 2017; Zaiem and Sadat, 2019; Yu et al., 2020; Vakulenko et al., 2020; Lin et al., 2020) that reformulate queries with generation models for other retrieval tasks. However, these studies are not easily applicable or efficient enough for OpenQA because: (1) They require external resources such as paraphrase data (Zaiem and Sadat, 2019), search sessions (Yu et al., 2020), or conversational contexts (Lin et al., 2020; Vakulenko et al., 2020) to form the reformulated queries, which are not available or showed inferior domain-transfer performance in OpenQA (Zaiem and Sadat, 2019); (2) They involve time-consuming training process such as RL. For example, Nogueira and Cho (2017) reported a training time of 8 to 10 days as it uses retrieval performance in the reward function and conducts retrieval at each iteration. In contrast, GAR uses freely accessible in-domain contexts like passage titles as the generation targets and standard

[2]Strictly speaking, GAR with sparse representations handles semantics before retrieval by enriching the queries, while maintaining the advantage of exact matching.

seq2seq learning, which, despite its simplicity, is not only more efficient but effective for OpenQA.

Retrieval for OpenQA. Existing sparse retrieval methods for OpenQA (Chen et al., 2017) solely rely on the information of the questions. GAR extends to contexts relevant to the questions by extracting information inside PLMs and helps sparse methods achieve comparable or better performance than dense methods (Guu et al., 2020; Karpukhin et al., 2020), while enjoying the simplicity and efficiency of sparse representations. GAR can also be used with dense representations to seek for even better performance, which we leave as future work.

Generative QA. Generative QA generates answers through seq2seq learning instead of extracting answer spans. Recent studies on generative OpenQA (Lewis et al., 2020a; Min et al., 2020; Izacard and Grave, 2020) are orthogonal to GAR in that they focus on improving the reading stage and directly reuse DPR (Karpukhin et al., 2020) as the retriever. Unlike generative QA, the goal of GAR is not to generate perfect answers to the questions but pertinent contexts that are helpful for retrieval. Another line in generative QA learns to generate answers without relevant passages as the evidence but solely the question itself using PLMs (Roberts et al., 2020; Brown et al., 2020). GAR further confirms that one can extract factual knowledge from PLMs, which is not limited to the answers as in prior studies but also other relevant contexts.

3 Generation-Augmented Retrieval

3.1 Task Formulation

OpenQA aims to answer factoid questions without pre-specified domains. We assume that a large collection of documents C (*i.e.*, Wikipedia) are given as the resource to answer the questions and a retriever-reader architecture is used to tackle the task, where the retriever retrieves a small subset of the documents $D \subset C$ and the reader reads the documents D to extract (or generate) an answer. Our goal is to improve the effectiveness and efficiency of the retriever and consequently improve the performance of the reader.

3.2 Generation of Query Contexts

In GAR, queries are augmented with various heuristically discovered relevant contexts in order to retrieve more relevant passages in terms of both quantity and quality. For the task of OpenQA where the query is a question, we take the following three freely accessible contexts as the generation targets. We show in Sec. 6.2 that having multiple generation targets is helpful in that fusing their results consistently brings better retrieval accuracy.

Context 1: The default target (answer). The default target is the label in the task of interest, which is the answer in OpenQA. The answer to the question is apparently useful for the retrieval of relevant passages that contain the answer itself. As shown in previous work (Roberts et al., 2020; Brown et al., 2020), PLMs are able to answer certain questions solely by taking the questions as input (*i.e.*, closed-book QA). Instead of using the generated answers directly as in closed-book QA, GAR treats them as contexts of the question for retrieval. The advantage is that even if the generated answers are partially correct (or even incorrect), they may still benefit retrieval as long as they are relevant to the passages that contain the correct answers (*e.g.*, co-occur with the correct answers).

Context 2: Sentence containing the default target. The sentence in a passage that contains the answer is used as another generation target. Similar to using answers as the generation target, the generated sentences are still beneficial for retrieving relevant passages even if they do not contain the answers, as their semantics is highly related to the questions/answers (examples in Sec. 6.1). One can take the relevant sentences in the ground-truth passages (if any) or those in the positive passages of a retriever as the reference, depending on the trade-off between reference quality and diversity.

Context 3: Title of passage containing the default target. One can also use the titles of relevant passages as the generation target if available. Specifically, we retrieve Wikipedia passages using BM25 with the question as the query, and take the page titles of positive passages that contain the answers as the generation target. We observe that the page titles of positive passages are often entity names of interest, and sometimes (but not always) the answers to the questions. Intuitively, if GAR learns which Wikipedia pages the question is related to, the queries augmented by the generated titles would naturally have a better chance of retrieving those relevant passages.

While it is likely that some of the generated query contexts involve unfaithful or nonfactual information due to hallucination in text generation (Mao et al., 2020) and introduce noise during retrieval, they are beneficial rather than harmful over-

all, as our experiments show that GAR improve both retrieval and QA performance over BM25 significantly. Also, since we generate 3 different (complementary) query contexts and fuse their retrieval results, the distraction of hallucinated content is further alleviated.

3.3 Retrieval with Generation-Augmented Queries

After generating the contexts of a query, we append them to the query to form a *generation-augmented query*.[3] We observe that conducting retrieval with the generated contexts (*e.g.*, answers) alone as queries instead of concatenation is ineffective because (1) some of the generated answers are rather irrelevant, and (2) a query consisting of the correct answer alone (without the question) may retrieve false positive passages with unrelated contexts that happen to contain the answer. Such low-quality passages may lead to potential issues in the following passage reading stage.

If there are multiple query contexts, we conduct retrieval using queries with different generated contexts separately and then fuse their results. The performance of one-time retrieval with all the contexts appended is slightly but not significantly worse. For simplicity, we fuse the retrieval results in a straightforward way: an equal number of passages are taken from the top-retrieved passages of each source. One may also use weighted or more sophisticated fusion strategies such as reciprocal rank fusion (Cormack et al., 2009), the results of which are slightly better according to our experiments.[4]

Next, one can use any off-the-shelf retriever for passage retrieval. Here, we use a simple BM25 model to demonstrate that GAR with sparse representations can already achieve comparable or better performance than state-of-the-art dense methods while being more lightweight and efficient (including the cost of the generation model), closing the gap between sparse and dense retrieval methods.

4 OpenQA with GAR

To further verify the effectiveness of GAR, we equip it with both extractive and generative readers for end-to-end QA evaluation. We follow the reader design of the major baselines for a fair comparison, while virtually any existing QA reader can be used with GAR.

4.1 Extractive Reader

For the extractive setup, we largely follow the design of the extractive reader in DPR (Karpukhin et al., 2020). Let $D = [d_1, d_2, ..., d_k]$ denote the list of retrieved passages with passage relevance scores $\mathbf{D}$. Let $S_i = [s_1, s_2, ..., s_N]$ denote the top N text spans in passage d_i ranked by span relevance scores $\mathbf{S_i}$. Briefly, the DPR reader uses BERT-base (Devlin et al., 2019) for representation learning, where it estimates the passage relevance score $\mathbf{D}_k$ for each retrieved passage d_k based on the [CLS] tokens of all retrieved passages D, and assigns span relevance scores S_i for each candidate span based on the representations of its start and end tokens. Finally, the span with the highest span relevance score from the passage with the highest passage relevance score is chosen as the answer. We refer the readers to Karpukhin et al. (2020) for more details.

Passage-level Span Voting. Many extractive QA methods (Chen et al., 2017; Min et al., 2019b; Guu et al., 2020; Karpukhin et al., 2020) measure the probability of span extraction in different retrieved passages independently, despite that their collective signals may provide more evidence in determining the correct answer. We propose a simple yet effective passage-level span voting mechanism, which aggregates the predictions of the spans in the same surface form from different retrieved passages. Intuitively, if a text span is considered as the answer multiple times in different passages, it is more likely to be the correct answer. Specifically, GAR calculates a normalized score $p(S_i[j])$ for the j-th span in passage d_i during inference as follows: $p(S_i[j]) = \text{softmax}(\mathbf{D})[i] \times \text{softmax}(\mathbf{S_i})[j]$. GAR then aggregates the scores of the spans with the same surface string among all the retrieved passages as the collective passage-level score.[5]

4.2 Generative Reader

For the generative setup, we use a seq2seq framework where the input is the concatenation of the question and top-retrieved passages and the target output is the desired answer. Such generative readers are adopted in recent methods such as SpanSe-

[3] One may create a title field during document indexing and conduct multi-field retrieval but here we append the titles to the questions as other query contexts for generalizability.

[4] We use the fusion tools at https://github.com/joaopalotti/trectools.

[5] We find that the number of spans used for normalization in each passage does not have significant impact on the final performance (we take $N = 5$) and using the raw or normalized strings for aggregation also perform similarly.

qGen (Min et al., 2020) and Longformer (Beltagy et al., 2020). Specifically, we use BART-large (Lewis et al., 2019) as the generative reader, which concatenates the question and top-retrieved passages up to its length limit (1,024 tokens, 7.8 passages on average). Generative GAR is directly comparable with SpanSeqGen (Min et al., 2020) that uses the retrieval results of DPR but not comparable with Fusion-in-Decoder (FID) (Izacard and Grave, 2020) since it encodes 100 passages rather than 1,024 tokens and involves more model parameters.

5 Experiment Setup

5.1 Datasets

We conduct experiments on the open-domain version of two popular QA benchmarks: Natural Questions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Trivia) (Joshi et al., 2017). The statistics of the datasets are listed in Table 1.

Dataset	Train / Val / Test	Q-len	A-len	#-A
NQ	79,168 / 8,757 / 3,610	12.5	5.2	1.2
Trivia	78,785 / 8,837 / 11,313	20.2	5.5	13.7

Table 1: Dataset statistics that show the number of samples per data split, the average question (answer) length, and the number of answers for each question.

5.2 Evaluation Metrics

Following prior studies (Karpukhin et al., 2020), we use top-k retrieval accuracy to evaluate the performance of the retriever and the Exact Match (EM) score to measure the performance of the reader.

Top-k retrieval accuracy is defined as the proportion of questions for which the top-k retrieved passages contain at least one answer span, which is an upper bound of how many questions are "answerable" by an extractive reader.

Exact Match (EM) is the proportion of the predicted answer spans being exactly the same as (one of) the ground-truth answer(s), after string normalization such as article and punctuation removal.

5.3 Compared Methods

For passage retrieval, we mainly compare with BM25 and DPR, which represent the most used state-of-the-art methods of sparse and dense retrieval for OpenQA, respectively. For query expansion, we re-emphasize that GAR is the first QE approach designed for OpenQA and most of the recent approaches are not applicable or efficient enough for OpenQA since they have task-specific objectives, require external supervision that was shown to transfer poorly to OpenQA, or take many days to train (Sec. 2). We thus compare with a classic unsupervised QE method RM3 (Abdul-Jaleel et al., 2004) that does not need external resources for a fair comparison. For passage reading, we compare with both extractive (Min et al., 2019a; Asai et al., 2019; Lee et al., 2019; Min et al., 2019b; Guu et al., 2020; Karpukhin et al., 2020) and generative (Brown et al., 2020; Roberts et al., 2020; Min et al., 2020; Lewis et al., 2020a; Izacard and Grave, 2020) methods when equipping GAR with the corresponding reader.

5.4 Implementation Details

Retriever. We use Anserini (Yang et al., 2017) for text retrieval of BM25 and GAR with its default parameters. We conduct grid search for the QE baseline RM3 (Abdul-Jaleel et al., 2004).

Generator. We use BART-large (Lewis et al., 2019) to generate query contexts in GAR. When there are multiple desired targets (such as multiple answers or titles), we concatenate them with [SEP] tokens as the reference and remove the [SEP] tokens in the generation-augmented queries. For Trivia, in particular, we use the value field as the generation target of answer and observe better performance. We take the checkpoint with the best ROUGE-1 F1 score on the validation set, while observing that the retrieval accuracy of GAR is relatively stable to the checkpoint selection since we do not directly use the generated contexts but treat them as augmentation of queries for retrieval.

Reader. Extractive GAR uses the reader of DPR with largely the same hyperparameters, which is initialized with BERT-base (Devlin et al., 2019) and takes 100 (500) retrieved passages during training (inference). Generative GAR concatenates the question and top-10 retrieved passages, and takes at most 1,024 tokens as input. Greedy decoding is adopted for all generation models, which appears to perform similarly to (more expensive) beam search.

6 Experiment Results

We evaluate the effectiveness of GAR in three stages: *generation* of query contexts (Sec. 6.1), *retrieval* of relevant passages (Sec. 6.2), and passage *reading* for OpenQA (Sec. 6.3). Ablation studies are mostly shown on the NQ dataset to understand the drawbacks of GAR since it achieves

Question: when did bat out of hell get released?
Answer: **September 1977** {September 1977}
Sentence: Bat Out of Hell is the second studio album and the major - label debut by American rock singer Meat Loaf ... released in **September 1977** on Cleveland International / Epic Records.
{The album was released in September 1977 on Cleveland International / Epic Records.}
Title: **Bat Out of Hell** {Bat Out of Hell}

Question: who sings does he love me with reba?
Answer: **Brooks & Dunn** {Linda Davis}
Sentence: **Linda Kaye Davis** (born November 26, 1962) is an American country music singer.
{" Does He Love You " is a song written by Sandy Knox and Billy Stritch, and recorded as a duet by American country music artists Reba McEntire and Linda Davis.}
Title: **Does He Love Me** [SEP] **Does He Love Me (Reba McEntire song)** [SEP] I Do (Reba McEntire album)
{Linda Davis [SEP] Greatest Hits Volume Two (Reba McEntire album) [SEP] Does He Love You}

Question: what is the name of wonder womans mother?
Answer: **Mother Magda** {Queen Hippolyta}
Sentence: In the Amazonian myths, she is the daughter of the Amazon queen Sifrat and the male dwarf Shuri, and is the mother of Wonder Woman. {Wonder Woman's origin story relates that she was sculpted from clay by her mother Queen Hippolyta and given life by Aphrodite.}
Title: **Wonder Woman** [SEP] **Diana Prince** [SEP] **Wonder Woman (2011 TV pilot)**
{Wonder Woman [SEP] Orana (comics) [SEP] Wonder Woman (TV series)}

Table 2: **Examples of generated query contexts**. **Relevant** and **irrelevant** contexts are shown in green and red. Ground-truth references are shown in the {braces}. The issue of generating wrong answers is alleviated by generating other contexts highly related to the question/answer.

better performance on Trivia.

6.1 Query Context Generation

Automatic Evaluation. To evaluate the quality of the generated query contexts, we first measure their lexical overlap with the ground-truth query contexts. As suggested by the nontrivial ROUGE scores in Table 3, GAR does learn to generate meaningful query contexts that could help the retrieval stage. We next measure the lexical overlap between the query and the ground-truth passage. The ROUGE-1/2/L F1 scores between the original query and ground-truth passage are 6.00/2.36/5.01, and those for the generation-augmented query are 7.05/2.84/5.62 (answer), 13.21/6.99/10.27 (sentence), 7.13/2.85/5.76 (title) on NQ, respectively. Such results further demonstrate that the generated query contexts significantly increase the word overlap between the queries and the positive passages, and thus are likely to improve retrieval results.[6]

Case Studies. In Table 2, we show several examples of the generated query contexts and their ground-truth references. In the first example, the correct album release date appears in both the generated answer and the generated sentence, and the generated title is the same as the Wikipedia page title of the album. In the last two examples, the generated answers are wrong but fortunately, the generated sentences contain the correct answer and (or) other relevant information and the generated titles are highly related to the question as well, which shows that different query contexts are complementary to each other and the noise during query context generation is thus reduced.

[6]We use F1 instead of recall to avoid the unfair favor of (longer) generation-augmented query.

Context	ROUGE-1	ROUGE-2	ROUGE-L
Answer	33.51	20.54	33.30
Sentence	37.14	24.71	33.91
Title	43.20	32.11	39.67

Table 3: **ROUGE F1 scores of the generated query contexts** on the validation set of the NQ dataset.

6.2 Generation-Augmented Retrieval

Comparison w. the state-of-the-art. We next evaluate the effectiveness of GAR for retrieval. In Table 4, we show the top-k retrieval accuracy of BM25, BM25 with query expansion (+RM3) (Abdul-Jaleel et al., 2004), DPR (Karpukhin et al., 2020), GAR, and GAR$^+$ (GAR +DPR).

On the NQ dataset, while BM25 clearly underperforms DPR regardless of the number of retrieved passages, the gap between GAR and DPR is significantly smaller and negligible when $k \geq 100$. When $k \geq 500$, GAR is slightly better than DPR despite

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)	43.6	62.9	78.1	85.5	87.8	67.7	77.3	83.9	87.9	88.9
BM25 +RM3	44.6	64.2	79.6	86.8	88.9	67.0	77.1	83.8	87.7	88.9
DPR	68.3	80.1	86.1	90.3	91.2	72.7	80.2	84.8	-	-
GAR	60.9	74.4	85.3	90.3	91.7	73.1	80.4	85.7	**88.9**	**89.7**
GAR$^+$	**70.7**	**81.6**	**88.9**	**92.0**	**93.2**	**76.0**	**82.1**	**86.6**	-	-

Table 4: **Top-k retrieval accuracy on the test sets**. The baselines are evaluated by ourselves and better than reported in Karpukhin et al. (2020). GAR helps BM25 to achieve comparable or better performance than DPR. Best and second best methods are **bold** and underlined, respectively.

that it simply uses BM25 for retrieval. In contrast, the classic QE method RM3, while showing marginal improvement over the vanilla BM25, does not achieve comparable performance with GAR or DPR. By fusing the results of GAR and DPR in the same way as described in Sec. 3.3, we further obtain consistently higher performance than both methods, with top-100 accuracy 88.9% and top-1000 accuracy 93.2%.

On the Trivia dataset, the results are even more encouraging – GAR achieves consistently better retrieval accuracy than DPR when $k \geq 5$. On the other hand, the difference between BM25 and BM25 +RM3 is negligible, which suggests that naively considering top-ranked passages as relevant (*i.e.*, pseudo relevance feedback) for QE does not always work for OpenQA. Results on more cutoffs of k can be found in App. A.

Effectiveness of diverse query contexts. In Fig. 1, we show the performance of GAR when different query contexts are used to augment the queries. Although the individual performance when using each query context is somewhat similar, fusing their retrieved passages consistently leads to better performance, confirming that different generation-augmented queries are complementary to each other (recall examples in Table 2).

Performance breakdown by question type. In Table 5, we show the top-100 accuracy of the compared retrieval methods per question type on the NQ test set. Again, GAR outperforms BM25 on all types of questions significantly and GAR$^+$ achieves the best performance across the board, which further verifies the effectiveness of GAR.

6.3 Passage Reading with GAR

Comparison w. the state-of-the-art. We show the comparison of end-to-end QA performance of extractive and generative methods in Table 6. Extractive GAR achieves state-of-the-art performance among extractive methods on both NQ and Trivia datasets, despite that it is more lightweight and computationally efficient. Generative GAR outperforms most of the generative methods on Trivia but does not perform as well on NQ, which is somewhat expected and consistent with the performance at the retrieval stage, as the generative reader only takes a few passages as input and GAR does not outperform dense retrieval methods on NQ when k is very small. However, combining GAR with DPR achieves significantly better performance than both

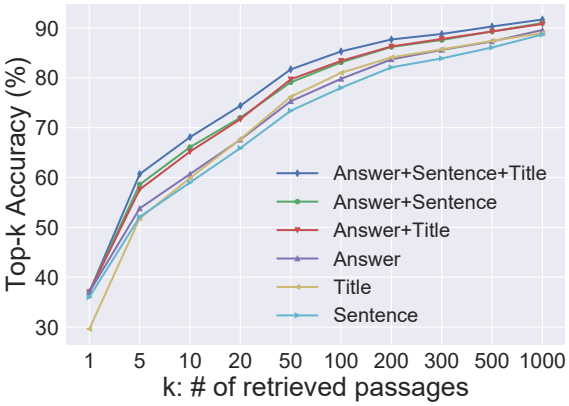

Figure 1: **Top-k retrieval accuracy** on the test set of NQ when fusing retrieval results of different generation-augmented queries.

Type	Percentage	BM25	DPR	GAR	GAR$^+$
Who	37.5%	82.1	88.0	87.5	**90.8**
When	19.0%	73.1	86.9	83.8	**88.6**
What	15.0%	76.5	82.6	81.5	**86.0**
Where	10.9%	77.4	89.1	87.0	**90.8**
Other	9.1%	79.3	78.1	81.8	**84.2**
How	5.0%	78.2	83.8	83.2	**85.5**
Which	3.3%	89.0	90.7	94.1	**94.9**
Why	0.3%	90.0	90.0	90.0	90.0

Table 5: **Top-100 retrieval accuracy breakdown of question type on NQ**. Best and second best methods in each category are **bold** and underlined, respectively.

	Method	NQ	Trivia	
Extractive	Hard EM (Min et al., 2019a)	28.1	50.9	-
	Path Retriever (Asai et al., 2019)	32.6	-	-
	ORQA (Lee et al., 2019)	33.3	45.0	-
	Graph Retriever (Min et al., 2019b)	34.5	56.0	-
	REALM (Guu et al., 2020)	40.4	-	-
	DPR (Karpukhin et al., 2020)	41.5	57.9	-
	BM25 (ours)	37.7	60.1	-
	GAR	**41.8**	**62.7**	**74.8**
	GAR$^+$	**43.8**	-	-
Generative	GPT-3 (Brown et al., 2020)	29.9	-	71.2
	T5 (Roberts et al., 2020)	36.6	60.5	-
	SpanSeqGen (Min et al., 2020)	42.2	-	-
	RAG (Lewis et al., 2020a)	44.5	56.1	68.0
	FID (Izacard and Grave, 2020)	**51.4**	**67.6**	**80.1**
	BM25 (ours)	35.3	58.6	-
	GAR	38.1	**62.2**	-
	GAR$^+$	**45.3**	-	-

Table 6: **End-to-end comparison with the state-of-the-art methods in EM**. For Trivia, the left column denotes the open-domain test set and the right is the hidden Wikipedia test set on the public leaderboard.

methods or baselines that use DPR as input such as SpanSeqGen (Min et al., 2020) and RAG (Lewis et al., 2020a). Also, GAR outperforms BM25 significantly under both extractive and generative setups, which again shows the effectiveness of the generated query contexts, even if they are heuristically discovered without any external supervision.

The best performing generative method FID (Izacard and Grave, 2020) is not directly comparable as it takes more (100) passages as input. As an indirect comparison, GAR performs better than FID when FID encodes 10 passages (cf. Fig. 2 in Izacard and Grave (2020)). Moreover, since FID relies on the retrieval results of DPR as well, we believe that it is a low-hanging fruit to replace its input with GAR or GAR$^+$ and further boost the performance.[7] We also observe that, perhaps surprisingly, extractive BM25 performs reasonably well, especially on the Trivia dataset, outperforming many recent state-of-the-art methods.[8] Generative BM25 also performs competitively in our experiments.

Model Generalizability. Recent studies (Lewis et al., 2020b) show that there are significant question and answer overlaps between the training and test sets of popular OpenQA datasets. Specifically, 60% to 70% test-time answers also appear in the training set and roughly 30% test-set questions have a near-duplicate paraphrase in the training set. Such observations suggest that many questions might have been answered by simple question or answer memorization. To further examine model generalizability, we study the per-category performance of different methods using the annotations in Lewis et al. (2020b).

Method	Total	Question Overlap	Answer Overlap Only	No Overlap
DPR	41.3	**69.4**	34.6	19.3
GAR$^+$ (E)	**43.8**	66.7	**38.1**	**23.9**
BART	26.5	67.6	10.2	0.8
RAG	44.5	**70.7**	34.9	24.8
GAR$^+$ (G)	**45.3**	67.9	**38.1**	**27.0**

Table 7: **EM scores with question-answer overlap category breakdown on NQ.** (E) and (G) denote extractive and generative readers, respectively. Results of baseline methods are taken from Lewis et al. (2020b). The observations on Trivia are similar and omitted.

As listed in Table 7, for the *No Overlap* category, GAR$^+$ (E) outperforms DPR on the extractive setup and GAR$^+$ (G) outperforms RAG on the generative setup, which indicates that better end-to-end model generalizability can be achieved by adding GAR for retrieval. GAR$^+$ also achieves the best EM under the *Answer Overlap Only* category. In addition, we observe that a closed-book BART model that only takes the question as input performs much worse than additionally taking top-retrieved passages, *i.e.*, GAR$^+$ (G), especially on the questions that require generalizability. Notably, all methods perform significantly better on the *Question Overlap* category, which suggests that the high *Total* EM is mostly contributed by question memorization. That said, GAR$^+$ appears to be less dependent on question memorization given its lower EM for this category.[9]

6.4 Efficiency of GAR

GAR is efficient and scalable since it uses sparse representations for retrieval and does not involve time-consuming training process such as RL (Nogueira and Cho, 2017; Liu et al., 2019). The only overhead of GAR is on the generation of query contexts and the retrieval with generation-

[7] This claim is later verified by the best systems in the NeurIPS 2020 EfficientQA competition (Min et al., 2021).

[8] We find that taking 500 passages during reader inference instead of 100 as in Karpukhin et al. (2020) improves the performance of BM25 but not DPR.

[9] The same ablation study is also conducted on the retrieval stage and similar results are observed. More detailed discussions can be found in App. A.

	Training	Indexing	Retrieval
DPR	24h w. 8 GPUs	17.3h w. 8 GPUs	30 min w. 1 GPU
GAR	3 $\sim$ 6h w. 1 GPU	0.5h w. 35 CPUs	5 min w. 35 CPUs

Table 8: **Comparison of computational cost between DPR and GAR at different stages.** The training time of GAR is for one generation target but different generators can be trained in parallel.

augmented (thus longer) queries, whose computational complexity is significantly lower than other methods with comparable retrieval accuracy.

We use Nvidia V100 GPUs and Intel Xeon Platinum 8168 CPUs in our experiments. As listed in Table 8, the training time of GAR is 3 to 6 hours on 1 GPU depending on the generation target. As a comparison, REALM (Guu et al., 2020) uses 64 TPUs to train for 200k steps during pre-training alone and DPR (Karpukhin et al., 2020) takes about 24 hours to train with 8 GPUs. To build the indices of Wikipedia passages, GAR only takes around 30 min with 35 CPUs, while DPR takes 8.8 hours on 8 GPUs to generate dense representations and another 8.5 hours to build the FAISS index (Johnson et al., 2017). For retrieval, GAR takes about 1 min to generate one query context with 1 GPU, 1 min to retrieve 1,000 passages for the NQ test set with answer/title-augmented queries and 2 min with sentence-augmented queries using 35 CPUs. In contrast, DPR takes about 30 min on 1 GPU.

7 Conclusion

In this work, we propose Generation-Augmented Retrieval and demonstrate that the relevant contexts generated by PLMs without external supervision can significantly enrich query semantics and improve retrieval accuracy. Remarkably, GAR with sparse representations performs similarly or better than state-of-the-art methods based on the dense representations of the original queries. GAR can also be easily combined with dense representations to produce even better results. Furthermore, GAR achieves state-of-the-art end-to-end performance on extractive OpenQA and competitive performance under the generative setup.

8 Future Extensions

Potential improvements. There is still much space to explore and improve for GAR in future work. For query context generation, one can explore multi-task learning to further reduce computational cost and examine whether different contexts can mutually enhance each other when generated by the same generator. One may also sample multiple contexts instead of greedy decoding to enrich a query. For retrieval, one can adopt more advanced fusion techniques based on both the ranking and score of the passages. As the generator and retriever are largely independent now, it is also interesting to study how to jointly or iteratively optimize generation and retrieval such that the generator is aware of the retriever and generates query contexts more beneficial for the retrieval stage. Last but not least, it is very likely that better results can be obtained by more extensive hyper-parameter tuning.

Applicability to other tasks. Beyond OpenQA, GAR also has great potentials for other tasks that involve text matching such as conversation utterance selection (Lowe et al., 2015; Dinan et al., 2020) or information retrieval (Nguyen et al., 2016; Craswell et al., 2020). The default generation target is always available for supervised tasks. For example, for conversation utterance selection one can use the reference utterance as the default target and then match the concatenation of the conversation history and the generated utterance with the provided utterance candidates. For article search, the default target could be (part of) the ground-truth article itself. Other generation targets are more task-specific and can be designed as long as they can be fetched from the latent knowledge inside PLMs and are helpful for further text retrieval (matching). Note that by augmenting (expanding) the queries with heuristically discovered relevant contexts extracted from PLMs instead of reformulating them, GAR bypasses the need for external supervision to form the original-reformulated query pairs.

Acknowledgments

We thank Vladimir Karpukhin, Sewon Min, Gautier Izacard, Wenda Qiu, Revanth Reddy, and Hao Cheng for helpful discussions. We thank the anonymous reviewers for valuable comments.

References

Nasreen Abdul-Jaleel, James Allan, W Bruce Croft, Fernando Diaz, Leah Larkey, Xiaoyan Li, Mark D Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard. *Computer Science Department Faculty Publication Series*, page 189.

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2019. Learning to retrieve reasoning paths over wikipedia

graph for question answering. *arXiv preprint arXiv:1911.10470*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

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 (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 758–759.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. 2020. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition*, pages 187–208. Springer.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Mandar Joshi, Eunsol Choi, Daniel 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 (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur 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. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020a. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2020b. Question and answer test-train overlap in open-domain question answering datasets. *arXiv preprint arXiv:2008.02637*.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2020. Query reformulation using query history for passage retrieval in conversational search. *arXiv preprint arXiv:2005.02230*.

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*, pages 1643–1652.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *arXiv preprint arXiv:1506.08909*.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2020. Sparse, dense, and attentional representations for text retrieval. *arXiv preprint arXiv:2005.00181*.

Yuanhua Lv and ChengXiang Zhai. 2010. Positional relevance model for pseudo-relevance feedback. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 579–586.

Yuning Mao, Xiang Ren, Heng Ji, and Jiawei Han. 2020. Constrained abstractive summarization: Preserving factual consistency with constrained generation. *arXiv preprint arXiv:2010.12723*.

Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, et al. 2021. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. *arXiv preprint arXiv:2101.00133*.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. 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)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. *arXiv preprint arXiv:2004.10645*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human-generated machine reading comprehension dataset.

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*, pages 574–583, Copenhagen, Denmark. Association for Computational Linguistics.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.

Joseph Rocchio. 1971. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, pages 313–323.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2020. Question rewriting for conversational question answering. *arXiv preprint arXiv:2004.14652*.

Xiao Wang, Craig Macdonald, and Iadh Ounis. 2020. Deep reinforced query reformulation for information retrieval. *arXiv preprint arXiv:2007.07987*.

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*, pages 1253–1256.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. *arXiv preprint arXiv:2006.05009*.

Salah Zaiem and Fatiha Sadat. 2019. Sequence to sequence learning for query expansion. In *Proceedings of the AAAI Conference on Artificial Intelligence, Student Abstract Track*, volume 33, pages 10075–10076.

A More Analysis of Retrieval Performance

We show the detailed results of top-k retrieval accuracy of the compared methods in Figs. 2 and 3. GAR performs comparably or better than DPR when $k \geq 100$ on NQ and $k \geq 5$ on Trivia.

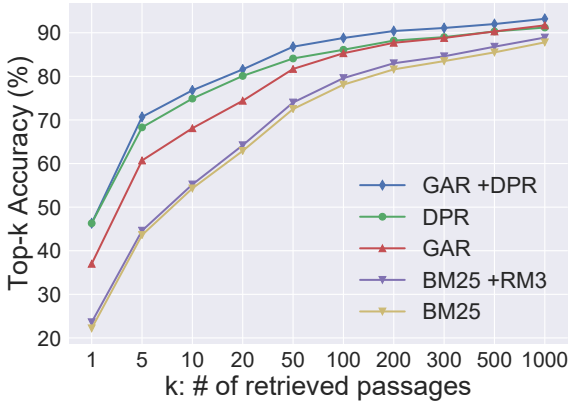

Figure 2: **Top-k retrieval accuracy of sparse and dense methods on the test set of NQ.** GAR improves BM25 and achieves comparable or better performance than DPR when $k \geq 100$.

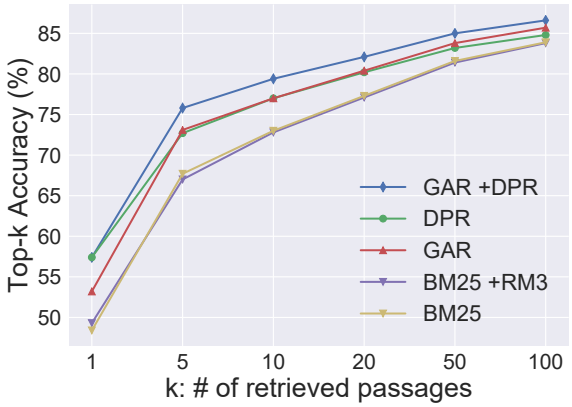

Figure 3: **Top-k retrieval accuracy on the Trivia test set.** GAR achieves better results than DPR when $k \geq 5$.

We show in Table 9 the retrieval accuracy breakdown using the question-answer overlap categories. The most significant gap between BM25 and other methods is on the *Question Overlap* category, which coincides with the fact that BM25 is unable to conduct question paraphrasing (semantic matching). GAR helps BM25 to bridge the gap by providing the query contexts and even outperform DPR in this category. Moreover, GAR consistently improves over BM25 on other categories and GAR$^+$ outperforms DPR as well.

Method	Total	Question Overlap	Answer Overlap Only	No Overlap
BM25	78.8	81.2	85.1	70.6
DPR	86.1	93.2	89.5	76.8
GAR	85.3	94.1	87.9	73.7
GAR$^+$	**88.9**	**96.3**	**91.7**	**79.8**

Table 9: **Top-100 retrieval accuracy by question-answer overlap categories on the NQ test set.**