arXiv:2009.08553v2 [cs.CL] 24 Oct 2020

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 Dynamics 365 AI [3]Microsoft Research
[1]{yuningm2, hanj}@illinois.edu
[2,3]{penhe, xiaodl, yeshe, jfgao,wzchen}@microsoft.com

Abstract

Conventional sparse retrieval methods such as TF-IDF and BM25 are simple and efficient, but solely rely on lexical overlap without semantic matching. Recent dense retrieval methods learn latent representations to tackle the lexical mismatch problem, while being more computationally expensive and insufficient for exact matching as they embed the text sequence into a single vector with limited capacity. In this paper, we present **G**eneration-**A**ugmented **R**etrieval (GAR), a query expansion method that augments a query with relevant contexts through text generation. We demonstrate on open-domain question answering that the generated contexts significantly enrich the semantics of the queries and thus GAR with sparse representations (BM25) achieves comparable or better performance than the state-of-the-art dense methods such as DPR (Karpukhin et al., 2020). We show that generating various contexts of 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. Furthermore, GAR achieves the state-of-the-art performance on the Natural Questions and TriviaQA datasets under the extractive setting when equipped with an extractive reader, and consistently outperforms other retrieval methods when the same generative reader is used.

1 Introduction

Classic retrieval methods such as TF-IDF and BM25 use sparse representations to measure lexical overlap. These sparse methods are lightweight and efficient, but unable to perform semantic matching and fail to retrieve relevant passages without explicit token overlap. To tackle the lexical mismatch problem, the traditional approach is query expansion (QE), which expands a query with relevant terms using *e.g.*, relevance models with (pseudo) relevance feedback (Lavrenko and Croft, 2001; Abdul-Jaleel et al., 2004). More recently, methods based on dense representations (Huang et al., 2013; Guu et al., 2020; Karpukhin et al., 2020) learn to embed queries and passages into a latent vector space, in which text semantics beyond lexical overlap can be measured. These methods can retrieve semantically relevant but lexically different passages and often achieve better performance than the sparse methods. However, the dense models are more computationally expensive and may 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).

In this paper, we propose a novel query expansion method for information retrieval, named Generation-Augmented Retrieval (GAR). At a high level, GAR augments the semantics of a query with relevant contexts (expansion terms) through text generation of a pre-trained language model (LM). For example, by prompting a pre-trained LM to generate the title of a relevant passage given a query and appending the generated title to the query, the *generation-augmented query* becomes semantically richer and thus it is easier to retrieve the relevant passage. Intuitively, the generated contexts of a query explicitly "express" the semantics of the search intent that is not presented in the original query. As a result, GAR with sparse representations can achieve comparable or even better performance than existing approaches with dense representations of the original queries, while being much more lightweight and efficient.

We evaluate the effectiveness of GAR on open-domain question answering (QA), which aims to answer factoid questions without a pre-specified domain and has numerous real-world applications

*Work was done during an internship at Microsoft Dynamics 365 AI.

(Kwiatkowski et al., 2019). In open-domain QA, a large collection of documents (*e.g.*, Wikipedia) are often used to find 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 the 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.

Instead of using questions as queries directly, GAR uses a pre-trained LM to generate contexts relevant to a question and expands the query by adding generated contexts. Specifically, we conduct Seq2Seq learning with the question as input and various generation targets as 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 various 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. First, GAR, combined with BM25, achieves significant gains over the same BM25 model that uses the original queries or conventional query expansion methods. Second, GAR, combined 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. Third, since GAR uses sparse representations to measure lexical overlap[1], it is complementary to dense representations: by fusing the retrieval results of GAR and DPR, we obtain consistently better performance than that of either method used individually. Lastly, GAR outperforms DPR in the end-to-end performance when the same extractive reader is used: EM=41.8 (43.8 when combining with DPR) on NQ and 62.7 on Trivia, creating new state-of-the-art results on the extractive open-domain QA. GAR (+DPR) also outperforms other retrieval methods under the generative setup when the same generative reader is used: EM=38.1 (45.3 when combining with DPR) on NQ and 61.8 on Trivia.

Contributions. (1) We propose Generation-Augmented Retrieval (GAR), a novel query expansion method that augments queries with relevant contexts through text generation. (2) We demonstrate that using generation-augmented queries achieves significantly better retrieval and QA results than using the original queries alone or the baseline query expansion method. (3) We show that GAR, combined with a simple BM25 model, achieves new state-of-the-art performance on two benchmark datasets in extractive open-domain QA and competitive results in the generative setting.

2 Related Work

Query / Document Expansion. Query expansion (QE) is widely used in information retrieval. GAR shares some merits with QE methods based on pseudo relevance feedback (Rocchio, 1971; Abdul-Jaleel et al., 2004; Lv and Zhai, 2010) in that both augment the queries with relevant contexts (terms). GAR is unique as it expands the queries with information stored in the pre-trained LMs rather than the retrieved passages and its expanded terms are learned through text generation.

There are also recent studies (Yu et al., 2020; Vakulenko et al., 2020; Lin et al., 2020) that expand (rewrite) queries with generative models. Notably, Yu et al. (2020) *rewrite* concise conversational queries to fully specified, context-independent queries by using sequential queries in the same search session as weak supervision. Alternatively, Nogueira et al. (2019) expand the documents by generating potential queries and appending them to the documents. However, these methods only use one type of generation target and (or) require additional resources such as search logs. Also, it is infeasible to expand every document with all possible queries that are potentially relevant. In contrast, GAR does not rewrite the original queries and leverages *various* query contexts such as passage titles and sentences, which are complementary to each other and freely accessible.

Retrieval for Open-domain QA. Early open-domain QA methods (Chen et al., 2017) use

[1] Strictly speaking, GAR with sparse representations handles semantics before rather than during retrieval by enriching the queries, while maintaining the advantage of exact matching. One can also use GAR with dense retrieval methods.

sparse representations for retrieval, while more recent methods (Guu et al., 2020; Karpukhin et al., 2020) leverage dense representations, *e.g.*, BERT bi-encoders (Devlin et al., 2019), and generally achieve better performance. GAR helps sparse retrieval methods to achieve comparable or better performance than dense methods, while enjoying the simplicity and efficiency of sparse representations. GAR can also be used with dense representations to seek for even better performance.

Generative QA. Generative QA generates answers through Seq2Seq learning instead of extracting answer spans. Recent studies on generative open-domain QA (Lewis et al., 2020; 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 pre-trained LMs (Roberts et al., 2020; Brown et al., 2020). GAR further confirms that one can retrieve factual knowledge from pre-trained LMs, which is not limited to the answers as in prior studies but also other relevant contexts.

3 Generation-Augmented Retrieval

3.1 Task Formulation

Open-domain QA 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 can be augmented with various generated contexts in order to retrieve more relevant passages in terms of both quantity and quality. For the task of open-domain QA where the query is a question, we take the following three contexts as the generation targets. We show in Sec. 5.3.2 that having multiple generation targets is helpful in that fusing their results consistently brings better retrieval accuracy.

Context 1: The default target (answer). The answer to the question is obviously 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), pre-trained LMs are able to answer certain questions solely by taking the questions as input and generating answers. Instead of using the generated answers directly, GAR takes them as contexts of the question for retrieval, the advantage of which 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. For example, we find that the answers of 46.8% questions in the test set of NQ can be found in the outputs of GPT-3 (Brown et al., 2020) when we sample 10 outputs and concatenate them (*i.e.*, the EM upper bound is 46.8), despite that its actual EM score is much lower – 23.3 in our study and 29.9 in Brown et al. (2020).[2]

We also observe that conducting retrieval with the generated answers alone as queries is somewhat ineffective because (1) some of the generated answers are rather irrelevant, and (2) a query with 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.

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. 5.3.1). One can take the relevant sentences in the gold-standard 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

[2]We do not use GPT-3 in the following experiments due to reproducibility.

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 relevant passages.

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] 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 different 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 BPFusion, RRFFusion, or BordaCountFusion.[4]

Next, one can use any off-the-shelf retrieval tool of interest 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 much more lightweight and efficient, closing the gap between sparse and dense retrieval methods.

4 Open-domain QA with GAR

To further verify the effectiveness of GAR, we equip it with both extractive and generative readers for end-to-end evaluation. We use the same reader as the major baselines for a fair comparison, while virtually any existing QA reader can be incorporated into 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 the span relevance scores $\mathbf{S_i}$, 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 the 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.

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 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 several top-retrieved passages and the output is the desired answer. Such generative readers are adopted in recent methods such as SpanSeqGen (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). 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

[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]The tools for the advanced fusion strategies can be found 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.

(FnD) (Izacard and Grave, 2020) since it encodes 100 passages rather than 1,024 tokens.

5 Experiments

5.1 Experimental Setup

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.

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.

5.2 Implementation Details

We use Anserini (Yang et al., 2017) for text retrieval of BM25 and GAR with its default parameters. We conduct grid search for the classic query expansion baseline RM3 (Abdul-Jaleel et al., 2004). 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. The generators on different datasets are trained independently without additional training samples from other datasets (*i.e.*, the single-dataset setting in DPR). Extractive GAR uses the same reader as 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). For all generative models, greedy decoding is adopted.

5.3 Experimental Results

We evaluate the effectiveness of GAR in three stages, namely the *generation* of query contexts (Sec. 5.3.1), the *retrieval* of relevant passages (Sec. 5.3.2), and the passage *reading* for open-domain QA (Sec. 5.3.3).

5.3.1 Query Context Generation

Automatic Evaluation. To evaluate the quality of the generated query contexts, an automatic evaluation with ROUGE is first conducted (Table 2). As suggested by the nontrivial ROUGE scores, GAR does learn to generate meaningful query contexts that could help the retrieval stage. 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 for retrieval.

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

Case Studies. In Table 3, we show several examples of the generated query contexts and their gold-standard 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.

5.3.2 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

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 3: **Examples of generated query contexts**. The issue of generating wrong answers is alleviated by generating other contexts highly related to the question/answer. Gold-standard references are shown in the [brackets].

Method	NQ				Trivia			
	Top-20	Top-100	Top-500	Top-1000	Top-20	Top-100	Top-500	Top-1000
BM25 (ours)	62.9	78.1	85.5	87.8	77.3	83.9	87.9	88.9
BM25 +RM3	64.2	79.6	86.8	88.9	77.1	83.8	87.7	88.9
DPR (Karpukhin et al., 2020)	**80.1**	**86.1**	90.3	91.2	80.2	84.8	-	-
GAR	74.4	85.3	**90.3**	**91.7**	**80.4**	**85.7**	**88.9**	**89.7**
GAR +DPR	**81.6**	**88.8**	**92.0**	**93.2**	**82.1**	**86.6**	-	-

Table 4: **Top-k retrieval accuracy of sparse and dense methods** on the test sets of NQ and Trivia. GAR helps BM25 to achieve comparable or better performance than DPR.

of BM25, BM25 with query expansion (+RM3) (Abdul-Jaleel et al., 2004), DPR (Karpukhin et al., 2020), GAR, and 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 that it simply uses BM25 for retrieval. In contrast, the classic query expansion 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 88.8% 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 $k \geq 20$. On the other hand, the difference between BM25 and BM25 +RM3 is negligible, which suggests that naively considering top-ranked passages as relevant (pseudo relevance feedback) for query expansion does not always work. Results on more cutoffs of k can be found in App. A.

Effectiveness of various 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 3).

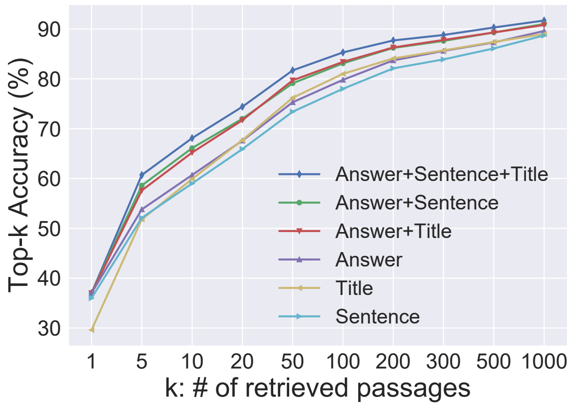

Figure 1: **Top-k retrieval accuracy** on the test set of NQ when fusing retrieval results of different generation-augmented queries.

	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 +DPR	**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., 2020)	44.5	56.1	68.0
	FnD (Izacard and Grave, 2020)	51.4	67.6	80.1
	BM25 (ours)	35.3	58.6	-
	GAR	38.1	**61.8**	-
	GAR +DPR	**45.3**	-	-

Table 5: **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.

5.3.3 Passage Reading with GAR

We show the comparison of end-to-end QA performance of extractive and generative methods in Table 5. Extractive GAR achieves the state-of-the-art performance among extractive methods on both NQ and Trivia datasets, despite that it is much 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 methods or baselines that use DPR as input such as SpanSeqGen (Min et al., 2020) and RAG (Lewis et al., 2020). Furthermore, GAR outperforms the standard BM25 method significantly under both extractive and generative setups, which again indicates the effectiveness of generation-augmented queries.

The best performing generative method FnD (Izacard and Grave, 2020) is not directly comparable as it takes much more (100) passages as input. As an indirect comparison, GAR performs better than FnD when FnD encodes 10 passages (cf. Fig. 2 in Izacard and Grave (2020)). Moreover, since FnD 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 +DPR and further boost the performance. We also observe that, perhaps surprisingly, BM25 (extractive) performs reasonably well when we take 500 passages during reader inference instead of 100 as in Karpukhin et al. (2020), especially on the Trivia dataset, outperforming many recent state-of-the-art methods.[6] BM25 (generative) also performs competitively compared with generative baselines.

5.4 Efficiency of GAR

GAR is efficient and scalable since it uses sparse representations (BM25) for retrieval. The only overhead of GAR is on the generation of query contexts and the retrieval with generation-augmented (thus longer) queries, whose computational complexity is significantly lower than other methods with comparable retrieval accuracy (Table 6).

	Training	Indexing	Inference
DPR	24h w. 8 GPUs	17.3h w. 8 GPUs	$\sim$ 30 min w. 1 GPU
GAR	3 $\sim$ 6h w. 1 GPU	0.5h w. 35 CPUs	$<$ 4 min w. 35 CPUs

Table 6: Comparison of computational complexity between DPR and GAR at different stages.

The training time of the generator in GAR is 3 to 6 hours on 1 V100 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), although more efficient, still takes about 24 hours to train with 8 V100 GPUs. To build indices of the Wikipedia passages (21 million), GAR only takes around 30 min with 35 CPUs,

[6]In contrast, increasing the number of retrieved passages for DPR does not improve its performance.

while DPR takes 8.8 hours on 8 GPUs to generate the dense representations and another 8.5 hours to build the FAISS index (Johnson et al., 2017). For retriever inference, GAR takes less than 1 min to retrieve 1,000 passages for the test set of NQ with answer/title-augmented queries and 2 min with sentence-augmented queries using 35 CPUs (*i.e.*, less than 4 min in total). In contrast, DPR takes about 30 min on 1 V100 GPU.

6 Discussions

We note that there is still much space to explore and improve for GAR in future work despite its promising results. 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. For passage 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 optimize generation and retrieval such that the generator is aware of the retriever and generates query contexts more beneficial for the retrieval stage.

Beyond open-domain QA, 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 broad search in information retrieval (Nguyen et al., 2016; Craswell et al., 2020). The default generation target is always available for different (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 pre-trained LMs and are helpful for further text retrieval (matching).

7 Conclusion

In this work, we propose Generation-Augmented Retrieval (GAR) and demonstrate on open-domain QA that the relevant contexts generated by pre-trained LMs can significantly enrich the 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. Furthermore, GAR achieves the state-of-the-art end-to-end performance on extractive open-domain QA and competitive performance under the generative setup.

Acknowledgments

We thank Vladimir Karpukhin, Sewon Min, Gautier Izacard, and Hao Cheng for helpful discussions.

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.

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*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

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.

Victor Lavrenko and W Bruce Croft. 2001. Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 120–127.

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. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

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*.

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.

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, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

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*.

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*.

A Details of Retrieval Performance

We show the detailed results of top-k retrieval accuracy of the compared methods in Figs. 2 and 3.

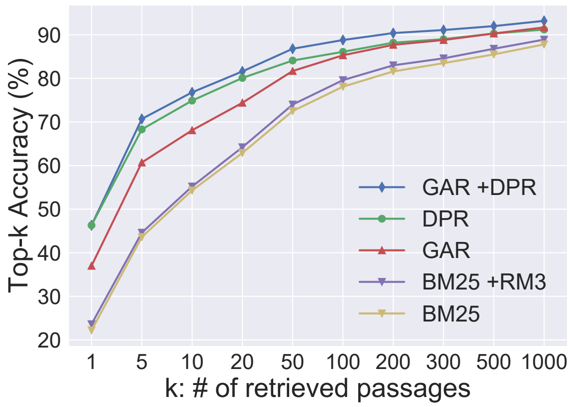

Figure 2: **Top-k retrieval accuracy of sparse and dense methods on the test set of NQ.** GAR improves BM25 and achieves comparable or slightly better performance than DPR when $k \geq 100$.

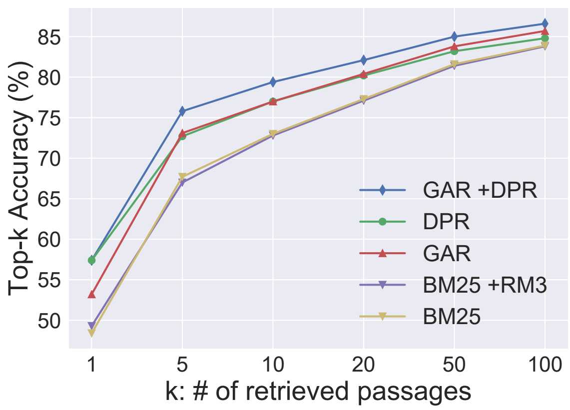

Figure 3: **Top-k retrieval accuracy on the test set of Trivia.** GAR achieves better performance than DPR when $k \geq 5$.