Expand, Rerank, and Retrieve: Query Reranking for Open-Domain Question Answering

Yung-Sung Chuang[†] **Wei Fang**[†] **Shang-Wen Li**[‡] **Wen-tau Yih**[‡] **James Glass**[†]
Massachusetts Institute of Technology[†] Meta AI[‡]
yungsung@mit.edu

Abstract

We propose EAR, a query **E**xpansion **A**nd **R**eranking approach for improving passage retrieval, with the application to open-domain question answering. EAR first applies a query expansion model to generate a diverse set of queries, and then uses a *query* reranker to select the ones that could lead to better retrieval results. Motivated by the observation that the best query expansion often is not picked by greedy decoding, EAR trains its reranker to predict the rank orders of the gold passages when issuing the expanded queries to a given retriever. By connecting better the query expansion model and retriever, EAR significantly enhances a traditional sparse retrieval method, BM25. Empirically, EAR improves top-5/20 accuracy by 3-8 and 5-10 points in in-domain and out-of-domain settings, respectively, when compared to a vanilla query expansion model, GAR, and a dense retrieval model, DPR.[1]

1 Introduction

Open-domain question answering (QA) (Chen and Yih, 2020), a task of answering a wide range of factoid questions of diversified domains, is often used to benchmark machine intelligence (Kwiatkowski et al., 2019) and has a direct application to fulfilling user's information need (Voorhees et al., 1999). To provide faithful answers with provenance, and to easily update knowledge from new documents, passage *retrieval*, which finds relevant text chunks to given questions, is critical to the success of a QA system. Retrieval in early open-domain QA systems (Chen et al., 2017) is typically based on term-matching methods, such as BM25 (Robertson et al., 2009) or TF-IDF (Salton et al., 1975). Such methods are sometimes called *sparse* retrievers, as they represent documents and queries with high-dimensional sparse vectors, and can efficiently match keywords with an inverted index and find relevant passages. Despite their simplicity, sparse retrievers are limited by their inability to perform semantic matching for relevant passages that have low lexical overlap with the query. Lately, dense retrievers (Karpukhin et al., 2020), which represent documents and queries with dense, continuous semantic vectors, have been adopted by modern QA systems. Dense retrievers usually outperform their sparse counterparts, especially when there exists enough in-domain training data.

However, dense retrievers have certain weaknesses compared to sparse ones, including: 1) being computationally expensive in training and inference, 2) potential information loss when compressing long passages into fixed-dimensional vectors (Luan et al., 2021), which makes it hard to match rare entities exactly (Sciavolino et al., 2021), and 3) difficulty in generalizing to new domains (Reddy et al., 2021). As a result, dense retrievers and sparse ones are usually complementary to each other and can be combined to boost performance. Recent studies on query expansion, such as GAR (Mao et al., 2021a), have attempted to improve sparse retrievers by adding relevant contexts to the query using pre-trained language models (PLMs), which has been shown effective in closing the gap between sparse and dense retrievers.

In this paper, we introduce a novel query **E**xpansion **A**nd **R**eranking approach, EAR, which enhances generative query expansion with **query reranking**. EAR first generates a diverse set of expanded queries with query expansion models, and then trains a query reranker to estimate the *quality* of these queries by directly predicting the rank order of a gold passage, when issuing these queries to a given retriever, such as BM25. At inference time, EAR selects the most promising query expansion as predicted by the query reranker and issues it to the same retriever to find relevant documents. EAR is motivated by a simple observation—while the greedy decoding output of a query expansion

[1] Source code: https://github.com/voidism/EAR.

Findings of the Association for Computational Linguistics: ACL 2023, pages 12131–12147
July 9-14, 2023 ©2023 Association for Computational Linguistics

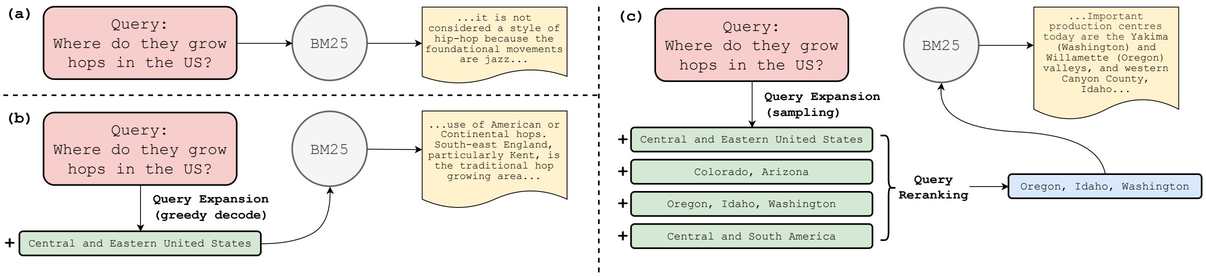

(a)
Query:
Where do they grow
hops in the US?
BM25
...it is not considered a style of hip-hop because the foundational movements are jazz...
(b)
Query:
Where do they grow
hops in the US?
Query Expansion
(greedy decode)
+ Central and Eastern United States
BM25
...use of American or Continental hops. South-east England, particularly Kent, is the traditional hop growing area...
(c)
Query:
Where do they grow
hops in the US?
Query Expansion
(sampling)
+ Central and Eastern United States
+ Colorado, Arizona
+ Oregon, Idaho, Washington
+ Central and South America
Query
Reranking
Oregon, Idaho, Washington
BM25
...Important production centres today are the Yakima (Washington) and Willamette (Oregon) valleys, and western Canyon County, Idaho...

Figure 1: (a) Standard BM25 pipeline (b) Generation-Augmented Retrieval (GAR) with BM25 (c) Our proposed Expand and Rerank (EAR) pipeline.

Model	**Top-1**	**Top-5**	**Top-20**	**Top-100**
1) BM25	22.1	43.8	62.9	78.3
2) DPR	43.0	66.4	78.5	85.0
3) GAR (greedy)	37.0	60.8	73.9	84.7
4) GAR (beam=10)	38.6	61.6	75.2	84.8
5) GAR *best query*	68.8	81.9	88.1	92.0
6) GAR *concat*	39.5	60.3	72.7	83.6

Table 1: The potential top-k retrieval accuracy that can be achieved by query reranking on Natural Questions. GAR uses greedy-decoded/beam-searched queries; GAR *best query* randomly samples 50 queries and picks the oracle one with the best retrieval scores; GAR *concat* simply concatenates all 50 queries as a single long query.

model, such as GAR, could be suboptimal, some randomly sampled query expansions achieve superior performance with BM25 (see Section 2.2). EAR better connects the query expansion model and the underlying retrieval method, and thus can select a more suitable query.

We empirically evaluated EAR in both *in-domain* and *cross-domain* settings. Our *in-domain* experimental results on Natural Questions and TriviaQA show that EAR significantly improves the top-5/20 accuracy by 3-8 points. For the *cross-domain* setting, while the query expansion model suffers from substantial performance degradation when applied to new domains, EAR seems to be more domain-agnostic, and can still find useful queries from a diverse set of query expansions, which leads to a significant improvement over GAR and DPR by 5-10 points for top-5/20 accuracy.

Our contributions can be summarized as follows:

- We proposed EAR to select the best query from a diverse set of query expansions, by predicting which query can achieve the best BM25 result. This improves the connection of query expansion models and BM25, resulting in enhanced performance that surpasses DPR.
- EAR not only performs well on in-domain data, but also shows strong generalization abilities on out-of-domain data, outperforming GAR and DPR by a large margin.
- End-to-end evaluation with a generative reader demonstrates the benefits of EAR in improving the exact match score.
- Lastly, we show that the improvements provided by EAR and passage reranking are complementary, allowing for effective aggregation of performance gains from both methods.

2 Background

2.1 Generation-Augmented Retrieval

Generation-Augmented Retrieval (GAR) (Mao et al., 2021a) aims to enhance sparse retrievers by query expansion with text generation from PLMs. Given the initial query, GAR generates relevant contexts including *the answer, answer sentence*, and *title of answer passages*, and then concatenates them to the initial query before performing retrieval with BM25. GAR achieves decent performance close to that of DPR while using the lightweight BM25 retriever. However, a limitation is that GAR is not aware of the existence of BM25, potentially generating suboptimal queries for retrieval. Additionally, GAR is only trained on in-domain data, limiting their ability to transfer to out-of-domain data.

2.2 Preliminary Experiments

Let us first take a look at some preliminary experimental results to better understand the motivation of this paper. In Table 1, we present the top-k retrieval results on Natural Questions (Kwiatkowski et al., 2019) for BM25, DPR, and GAR (greedy/beam search) in rows 1-4. To investigate the potential of GAR, we randomly sampled 50 query expansions from GAR, ran BM25 separately for these queries, and chose the best one by looking at the BM25 results, which requires ground truth labels. The resulting scores are shown in row 5 (GAR *best query*).

From the results, we see that GAR *best query* can lead to a significant improvement of up to 20 points compared to DPR. Since we do not have access to labels for selecting the best query in reality, a naive solution is to concatenate all 50 query expansions together as a single, long query, which will definitely include high-quality expansions if they exist. However, as shown in row 6, the performance of GAR *concat* is even worse than that of GAR alone with greedy decoding outputs. This indicates that the single long query may include too much distracting information, negatively impacting the performance of the BM25 retriever.

From these preliminary results, we reach two conclusions: 1) GAR does have the ability to generate very useful query expansions; 2) however, the useful query expansions may not always be included in the GAR greedy decoding outputs. It is non-trivial to extract these useful query expansions from GAR. Motivated by these findings, we leverage a query reranker to estimate if a query will be beneficial to BM25 retrieval results, so as to unlock the potential of GAR and sparse retrievers.

3 Proposed Method

We illustrate our proposed method, EAR, in Figure 1, along with a comparison with the BM25 and GAR pipelines. Given the original query q, EAR first generates a set of query expansions $E = \{e_1, e_2, ..., e_n\}$ using random sampling. We believe that among these n queries, some may achieve very good retrieval performance. Thus, we train a reranker model $\mathcal{M}$ to re-score all the queries. Here we propose two kinds of rerankers: 1) retrieval-independent (RI) reranker, and 2) retrieval-dependent (RD) reranker. Both rerankers can estimate the quality of a query expansion without using information from answer annotations.

3.1 Retrieval-Independent (RI) Reranker

The inputs to the RI reranker are quite simple: (q, e_i), which consists of the original query q and one of the query expansions e_i. When training this reranker, we first obtain the minimum answer passage ranking among all retrieved passages for each query, when issued to a BM25 retriever. We denote this minimum answer passage ranking as $R = \{r_1, r_2, ..., r_n\}$, which corresponds to each of the expanded queries $\{(q, e_1), (q, e_2), ..., (q, e_n)\}$.

To clarify the concept, let us consider an example with two query expansions, e_1 and e_2. Say the expanded query (q, e_1) retrieves the answer passage as the top result (first position), we assign $r_1 = 1$. Similarly, we assign $r_2 = 15$ if the expanded query (q, e_2) retrieves the answer passage in the 15th position. In this case, we conclude that e_1 is a better query expansion than e_2 since its corresponding ranking value, r_1, is lower than r_2.

r_i can be seen as the *score* that can be obtained by the query of (q, e_i), with smaller r_i corresponding to better quality of (q, e_i). We now train a scoring model to estimate the rank r_i for given inputs (q, e_i). However, considering that the scoring model will be used as a reranker, we only need to ensure the model's *relative* accuracy of estimating r_i, rather than its *absolute* value. Thus, we employ a “contrastive” loss rather than a “regression” loss, which is inspired by the contrastive method in summarization re-scoring (Liu and Liu, 2021).

For all pairs of query expansions (e_i, e_j) such that $r_i < r_j$ (which means e_i is a better expansion than e_j), the ranking loss is calculated as follows:

$$\mathcal{L}_{\text{Rank}} = \sum_{\substack{i,j\in[1,n] \\ r_i<r_j}} \max(0, \mathcal{M}(q,e_i) - \mathcal{M}(q,e_j) + (r_j - r_i) \cdot \alpha)$$

Here, $\mathcal{M}$ is a model that estimates the rank r_i for a given query expansion e_i. Instead of predicting the absolute rank of r_i, the model $\mathcal{M}$ is trained to predict the difference between r_i and r_j for each pair of expansion (e_i, e_j).

The ranking loss $\mathcal{L}_{\text{Rank}}$ forces the model to estimate a lower rank for e_i and a higher rank for e_j, such that the difference between $\mathcal{M}(q, e_i)$ and $\mathcal{M}(q, e_j)$ is greater than the threshold $(r_j - r_i) \cdot \alpha$, where α is a scalar. If some of the expansions do not retrieve the answer passages within the top-k results (e.g. within the top-100 results), we assign a constant value, `MAX_RANK`, to these expansions.

3.2 Retrieval-Dependent (RD) Reranker

The input to the RI Reranker only contains the original query q and the expansion e_i, which may not be sufficient to distinguish good expansions from bad expansions. For example, in Figure 1 (c), for the original query *Where do they grow hops in the US?*, it is easy to tell that *Central and South America* is a bad expansion because the US is not in Central and South America. However, for these two expansions: 1) Colorado, Arizona 2) Oregon, Idaho, Washington, it is very hard to tell which one is better without any external knowledge. To alleviate this problem, we propose the Retrieval-Dependent (RD) Reranker, which is able to see the top-1 passages $D = \{d_1, d_2, ..., d_n\}$ retrieved by each query expansion. [2] The inputs of RD reranker will contain the original query q, the query expansions e_i, and the top-1 passage d_i. We train RD reranker with the same ranking loss $\mathcal{L}_{\text{Rank}}$, but replace the model with $\mathcal{M}(q, e_i, d_i)$.

3.3 Training Examples Construction

To construct training examples, we generate diverse query expansions, run BM25 retrieval on them, and train the rerankers based on the results. However, using the GAR generators directly may not yield diverse sequences and limit the rerankers' learning, since the GAR generators are trained with supervision and may have already overfit on the training set, which would lead to almost identical generation samples. To address this, we propose two alternatives: 1) Split the training set into K subsets, train K different GAR generators on (K-1) subsets and randomly sample from the remaining subset; and 2) Use a large language model (LLM) such as T0 (Sanh et al., 2021) to randomly sample query expansions directly without fine-tuning. Both options performed equally well in our experiments and will be further discussed in Section 6.1.

4 Experiments

4.1 Data

For in-domain experiments, we use two public datasets for training and evaluation: Natural Questions (NQ) (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017). For out-of-domain (cross-dataset) experiments, we directly evaluate our in-domain models on three additional public datasets without using their training sets: WebQuestions (WebQ) (Berant et al., 2013), CuratedTREC (TREC) (Baudiš and Šedivỳ, 2015), and EntityQuestions (EntityQs) (Sciavolino et al., 2021). (See dataset statistics in Appendix A.) All experiments are performed with Wikipedia passages used in DPR (Karpukhin et al., 2020), consisting of 21M 100-word passages from the English Wikipedia dump of Dec. 20, 2018 (Lee et al., 2019).

4.2 Setup

Model For sparse retrieval, we use Pyserini (Lin et al., 2021) for BM25 with its default parameters. For query rerankers, we use the DeBERTa V3 base (He et al., 2021) model from Huggingface Transformers (Wolf et al., 2020). For RI reranker, the input format is: `[CLS] <question> ? <expansion> [SEP]`; for RD reranker, the input format is `[CLS] <question> ? <expansion> [SEP] <top-1 retrieved passage> [SEP]`. Training details can be found in Appendix B.

Context Generator At training time, we use T0-3B (Sanh et al., 2021) to randomly sample 50 query expansions per question, as we mentioned in Section 3.3. We add a short prompt, *To answer this question, we need to know*, to the end of the original question, and let T0-3B complete the sentence. During inference, we still use the GAR generators to randomly sample 50 query expansions per question on the testing set, since the examples are not seen during GAR training and the generations are diverse enough. To speed up the inference process, we de-duplicate the query expansions that appear more than once. The average number of query expansions we use is 25 for Natural Questions and 34 for TriviaQA, respectively.

4.3 Baselines

We compare EAR with 1) DPR (Karpukhin et al., 2020): a standard BERT-based dense retriever; 2) BM25 (Robertson et al., 2009): a standard sparse retriever based on term matching; 3) GAR (Mao et al., 2021a): generation-augmented retrieval with BM25; 4) Liu et al. (2022): a concurrent work that uses a GAR-like generative model to perform beam search decoding, followed by filtering to obtain multiple expanded queries for performing multiple retrievals with BM25, and then fusion of the results; and 5) SEAL (Bevilacqua et al., 2022): an autoregressive search engine, proposing constrained decoding with the FM-index data structure that enables autoregressive models to retrieve passages.

4.4 Result: In-Domain Dataset

We first train and evaluate EAR on NQ and TriviaQA. In Table 2, we see that both EAR-RI and EAR-RD improve the performance of GAR significantly. EAR-RI improves the top-5/20/100 accuracy of GAR by 1-2 points, while EAR-RD improves the top-5 accuracy of GAR by 6-8 points, and the top-20 accuracy by 3-5 points on both datasets. More-

[2] For RD reranker, we need additional computational costs to retrieve the top-1 passage. However, this process can be efficiently parallelized for all queries using a lightweight BM25 retriever, so the required time is not excessive. We will discuss the latency of EAR further in Section 7.

Model	Natural Questions			TriviaQA		
	Top-5	Top-20	Top-100	Top-5	Top-20	Top-100
Dense Retrieval						
DPR	68.3	80.1	86.1	72.7	80.2	84.8
Lexical Retrieval						
BM25	43.8	62.9	78.3	67.7	77.3	83.9
GAR	60.8	73.9	84.7	71.8	79.5	85.3
SEAL	61.3	76.2	86.3	-	-	-
Liu et al. (2022)	63.9	76.8	**86.7**	72.3	80.1	85.8
EAR-RI	63.2	76.4	85.9	73.4	80.8	85.9
EAR-RD	**69.3**	**78.6**	86.5	**77.6**	**82.1**	**86.4**
GAR *best query*	*81.9*	*88.1*	*92.0*	*85.0*	*88.1*	*90.1*
Fusion (Dense + Lexical) Retrieval						
BM25 + DPR	69.7	81.2	88.2	71.5	79.7	85.0
GAR + DPR	72.3	**83.1**	88.9	75.7	82.2	86.3
Liu et al. (2022) + DPR	72.7	83.0	89.1	76.1	82.5	86.4
EAR-RI + DPR	71.1	82.5	89.1	76.4	83.0	87.0
EAR-RD + DPR	**74.2**	**83.1**	**89.3**	**79.0**	**83.7**	**87.3**

Table 2: Top-k retrieval accuracy (%) on the NQ and TriviaQA test sets. Numbers for prior work are cited from Liu et al. (2022).

over, EAR-RD is significantly better than DPR except for the top-20 accuracy on NQ. These results show that it is possible for BM25 to beat dense retrieval with the help of an optimized process to generate high-quality query expansions. Additional qualitative studies in Appendix E provide further insight into how EAR works. We also report the results of *the best query from* GAR, which presents the potential performance upper bound that could be achieved by query reranking. It suggests that there is still room for EAR to improve if mechanisms for more effective query selection are developed. At the bottom of Table 2, we present the fusion retrieval results of combining EAR and DPR. EAR-RD+DPR outperforms the fusion results of BM25/GAR/Liu et al. (2022), showing the complementarity between EAR-RD and DPR.

4.5 Result: Cross-Dataset Generalization

To better evaluate the robustness of these models for out-of-domain examples, we train our models only on NQ or TriviaQA, and then test them on WebQ, TREC, and EntityQs in a *zero-shot* manner. The results are shown in Table 3. We observe that when transferring from NQ or TriviaQA, DPR experiences a decline in performance compared to in-domain supervised training on WebQ.[3] GAR performs even worse than DPR on both WebQ and TREC. However, GAR performs better than DPR on EntityQs, which is designed to challenge dense retrieval by including many rare entities. Here we also present the performance of GAR *best query*. We see that although GAR transfers poorly on cross-domain datasets, it still has the ability to generate high-quality query expansions by random sampling. This provides an opportunity for EAR to improve performance. After adopting EAR, we see that EAR-RI improves the performance of GAR by 2-4 points for top-5/20 accuracy, and EAR-RD further boosts the performance of GAR by 5-10 points for top-5/20 accuracy. Overall, EAR-RD outperforms DPR except when transferring from TriviaQA to WebQ.

These results suggest that query reranking is a general technique that can work well even on out-of-domain examples, showing that *generating relevant contexts* (GAR) is largely dependent on the domains, while *judging which contexts may be more beneficial to retriever* is a more domain-agnostic skill.

4.6 Result: End-to-end QA with FiD

To fully understand whether EAR can benefit end-to-end QA systems, we further evaluate the exact match scores with Fusion-in-Decoder (FiD) (Izacard and Grave, 2021), a generative reader model trained from T5-large (Raffel et al., 2020). We take the FiD models that were pre-trained on

[3] The in-domain DPR performs poorly on TREC since it only has 1,125 training examples.

Model	WebQuestions			TREC			EntityQuestions		
	Top-5	Top-20	Top-100	Top-5	Top-20	Top-100	Top-5	Top-20	Top-100
BM25	41.8	62.4	75.5	64.3	80.7	89.9	60.6	70.8	79.2
In-Domain Supervised									
DPR†	62.8	74.3	82.2	66.6	81.7	89.9	-	-	-
Transfer from NQ									
DPR†	52.7	68.8	78.3	74.1	85.9	92.1	38.1	49.7	63.2
GAR	50.0	66.0	79.0	70.9	83.9	92.4	59.7	71.0	79.8
EAR-RI	53.7	69.6	81.3	73.5	85.9	92.9	62.7	73.3	81.4
EAR-RD	**59.5**	**70.8**	**81.3**	**80.0**	**88.9**	**93.7**	**65.5**	**74.1**	**81.5**
GAR *best query*	*78.9*	*85.4*	*90.3*	*93.1*	*95.5*	*97.1*	*78.6*	*85.2*	*90.9*
Transfer from TriviaQA									
DPR†	**56.8**	**71.4**	**81.2**	78.8	87.9	**93.7**	51.2	62.7	74.6
GAR	45.5	61.8	76.7	71.5	84.0	91.5	58.2	68.9	78.7
EAR-RI	49.6	67.1	79.6	74.2	86.2	92.5	62.1	72.0	80.4
EAR-RD	54.5	68.0	79.7	**79.8**	**88.5**	93.1	**64.9**	**73.0**	**80.5**
GAR *best query*	*78.4*	*84.6*	*89.3*	*92.5*	*95.2*	*96.8*	*79.1*	*85.9*	*91.8*

Table 3: Top-k retrieval accuracy on the test sets of three datasets for cross-dataset generalization settings. GAR *best query* represents the performance upper bound we can achieve by selecting the best query according to the labels. Numbers in bold are the best scores for each setting. †Results are provided by Ram et al. (2022).

Model	NQ	TriviaQA
Top-100 passages as input to FiD		
DPR + Extractive	41.5	57.9
RAG	44.5	56.1
DPR + FiD	51.4	67.6
GAR + FiD	50.6	70.0
SEAL + FiD	50.7	-
Liu et al. (2022) + FiD	51.7	70.8
EAR RI + FiD	51.4	71.2
EAR RD + FiD	**52.1**	**71.5**
Top-10 passages as input to FiD		
GAR + FiD	30.5	48.9
EAR RI + FiD	35.5	56.7
EAR RD + FiD	**39.6**	**60.0**

Table 4: End-to-end QA exact-match scores on the test sets of NQ and TriviaQA. Numbers for prior work are cited from Liu et al. (2022).

NQ/TriviaQA and directly test on our retrieval results without further fine-tuning. The exact match scores using the top-100 retrieved passages as input to FiD is shown at the top of Table 4. We observe that EAR consistently outperforms previous work, including DPR, GAR, SEAL, and Liu et al. (2022), on both NQ and TriviaQA. Although these gains may appear relatively small, however, this is primarily due to FiD's ability to take the top-100 retrieved passages as input and generate answers using cross-attention across all passages. Thus, even with low-ranked answer passages (say the answer is in the 99th passage), it is still possible that FiD could produce correct answers.

As there are many methods where relatively smaller context windows compared to FiD are used, especially when models are scaled up and cross-attention becomes much more expensive, improving retrieval accuracy for smaller k may be beneficial. For example, GPT-3 (Brown et al., 2020) only has a context window size of 2048, which can only support 10-20 passages as input. We explore this setting by selecting only the top-10 retrieved passages as input to FiD, and show the results at the bottom of Table 4. EAR achieve significant improvement over GAR, roughly 10% in exact match on both datasets, showing potential benefits for methods with limited context window size.

5 Query Reranking vs Passage Reranking

EAR shares similarities with passage reranking (PR). EAR reranks the queries before retrieving the passages, while PR reranks the retrieved list of passages after the retrieval process is completed. To better understand the relationship between EAR and PR, we implement a BERT-based passage reranker, following the method outlined in Nogueira and Cho (2019), to rerank the retrieval results of GAR. The implementation details can be found in Appendix C. From the experiments we aim to answer three questions: 1) Is EAR better than PR? 2) Are the contributions of EAR and PR

Model	Natural Questions			TriviaQA		
	Top-5	**Top-20**	**Top-100**	**Top-5**	**Top-20**	**Top-100**
BM25	43.8	62.9	78.3	67.7	77.3	83.9
GAR	60.8	73.9	84.7	71.8	79.5	85.3
GAR + Passage Rerank (k=25/k=32)	68.8	75.7	84.7	77.6	81.0	85.3
GAR + Passage Rerank (k=100)	71.7	80.2	84.7	79.2	83.3	85.3
EAR-RD	69.3	78.6	**86.5**	77.6	82.1	**86.4**
EAR-RD + Passage Rerank (k=100)	**73.7**	**82.1**	**86.5**	**80.6**	**84.5**	**86.4**

Table 5: Top-k retrieval accuracy (%) on the Natural Questions and TriviaQA test sets for comparison of query reranking and passage reranking.

complementary? Can their performance gains be aggregated if we apply both? 3) What are the extra advantages of EAR compared to PR?

Is EAR better than PR? We focus on comparing EAR-RD with PR, as EAR-RI is limited by its input, being able to see only the short expanded queries. On the other hand, EAR-RD has access to the top-1 passage retrieved by each query candidate, providing it with the same level of information as PR. In Table 5, we first present the performance of PR when reranking the same number of passages as the average number of query candidates considered by EAR (25 for NQ; 32 for TriviaQA), which can be found in row 3. The result of EAR-RD (shown in row 5) is better than row 3, indicating that when considering the same amount of information as inputs, EAR-RD outperforms PR. However, when PR is able to rerank a larger number of passages, such as the top-100 passages shown in row 4, it achieves better performance than EAR-RD. This implies that EAR-RD is more effective when PR can only access to the same level of information.

Are EAR and PR complementary? We found that the effects of EAR-RD and PR can be effectively combined for even better performance. When applying PR on the retrieval results of EAR-RD (shown in row 6), we see a significant improvement compared to both row 4 and row 5. This suggests that the contributions of EAR-RD and PR are complementary: EAR strengthens first-pass retrieval by selecting good queries, while PR rescores all the retrieved passages and generates an entirely new order for these passages. The distinction between these two mechanisms makes improvements accumulative and leads to superior results.

Extra advantages of EAR? An advantage of EAR is that it improves retrieval results beyond the top-k passages. In row 4, the top-100 accuracy cannot be improved by PR as it reranks within the top-100 passages. In contrast, the improvements provided by EAR are not limited to top-100 passages. As long as EAR selects good query expansions, it can improve the whole list of retrieved passages; we can see EAR-RD improves the top-100 accuracy of GAR from 84.7 to 86.5.

Model	Top-5	Top-20	Top-100
EAR-RI	63.2	76.4	85.9
EAR-RI holdout	63.6	76.3	86.0
EAR-RD	69.3	78.6	86.5
EAR-RD w/ DPR	65.7	78.7	86.3

Table 6: Top-k retrieval accuracy (%) on NQ for comparison of the two different training example construction methods and for EAR with dense retrievers.

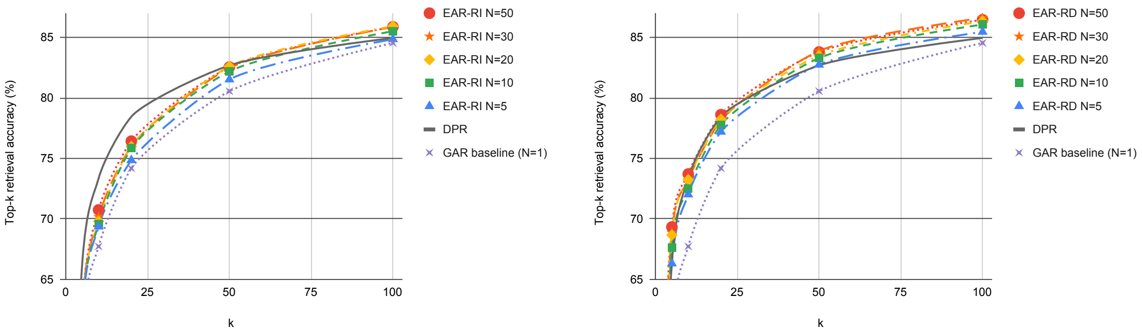
6 Discussions

6.1 Generating Training Examples with GAR

In Section 3.3, we discussed two methods to construct training examples for EAR. In our main experiments, we used T0-3B to randomly sample diverse query expansions. An alternative method was also explored, where we trained $K = 5$ different GAR models separately on $(K - 1)$ training subsets, then randomly sampled from the hold-out sets. The performance of this method, as shown in Table 6 (EAR-RI holdout), is slightly better than using T0-3B, but the difference is less than 0.5 points on Top-5/20/100 accuracy. Therefore, we continue to use T0-3B to generate training data in our main experiments as it eliminates the need to train K different GAR models separately.

6.2 EAR with Dense Retrievers

EAR is specifically optimized to work well with the BM25 retriever and hence its performance may be impacted when changing the retriever to DPR. As shown at the bottom of Table 6, when coupled with DPR, the top-5 accuracy of EAR-RD decreases,



(a) EAR-RI

(b) EAR-RD

Figure 2: Top-k performance curves on NQ for EAR-RI and EAR-RD with a reduced candidate size N.

Model	RI	RD
EAR N=50	63.16	69.34
EAR N=30	63.02	68.86
EAR N=20	62.96	68.67
EAR N=10	62.60	67.62
EAR N=5	62.44	66.32
GAR baseline (N=1)	60.80	

Table 7: Top-5 accuracy on NQ with different N. N stands for the maximum number of query expansions considered by the query reranker.

while the top-20/100 accuracy remains relatively unchanged. This suggests that EAR is heavily reliant on the retriever, and thus changing the retriever negatively impacts its performance. Making EAR work with DPR would require retraining with DPR retrieval results and significantly more compute. We leave this direction for future work.

6.3 Reducing the Query Candidate Size

In our experiments, we generate 50 query expansions per question and then de-duplicate the repeated ones. However, we can also limit the maximum query expansions considered by our reranker to trade off between efficiency and performance. In Table 7 we show the top-5 accuracy of lowering the maximum candidate size N from 50 to 30/20/10/5. We observe that the performance drops gradually as N decreases. However, we still see improvement over GAR even when $N = 5$, showing that EAR still works with a small candidate size. We also show the curves of the top-k accuracy in Figure 2, where we observe a big gap between DPR (solid line) and GAR (dotted line with x mark). EAR-RI gradually eliminates the gap as N increase, while EAR-RD even matches DPR for $k < 50$ and outperforms DPR for $k \geq 50$ with a small $N = 5$.

Model	Build Index	Query Expand	Query Rerank	Retrieval	Index Size	Top-5 (NQ)
DPR	3.5hr	-	-	22.4s	64GB	68.3
+HNSW	8.5hr	-	-	0.04s	142GB	68.0
BM25	0.5hr	-	-	0.15s	2.4GB	43.8
GAR	0.5hr	0.58s	-	0.56s	2.4GB	60.8
EAR-RI	0.5hr	1.29s	0.04s	0.50s	2.4GB	63.2
EAR-RD	0.5hr	1.29s	0.84s	0.54s	2.4GB	69.3

Table 8: Latency per query for DPR/BM25/GAR/EAR.

7 Computational Cost and Latency

We report the latency of DPR, GAR, and EAR in Table 8. Inference details can be found in Appendix D.

Dense Retrieval We first generate DPR document embeddings on 4 GPUs for ~3.5 hours on 21M documents. Standard indexing takes ~10 minutes with a 64GB index size. Indexing with the more advanced Hierarchical Navigable Small World (HNSW) (Malkov and Yashunin, 2018) takes ~5 hours and results in a huge index size of 142GB. For retrieval, standard indexing takes 22.3s per query, while the highly optimized HNSW can shorten it to 0.04s per query.

Sparse Retrieval For BM25 with Pyserini, indexing only takes 0.5 hours, with a very small index size of 2.4GB. Retrieval for BM25 takes 0.15s per query. For GAR, it needs an extra 0.58s to generate the query expansions, and retrieval time is 0.56s. For EAR, it needs 1.29s to batch sample 50 query expansions. EAR-RI only takes 0.04s to rerank queries. EAR-RD needs extra time to retrieve the top-1 passages for each expansion, which takes an extra 0.70s, and then run the actual reranking process, taking 0.14s, giving a total of 0.84s for query reranking. For retrieval, the time needed for both EAR-RI and EAR-RD is similar to GAR.

To conclude, EAR inherits the advantage of

BM25: *fast indexing time* and *small index size*. This makes it possible to index large collections of documents in a relatively short amount of time, which is important for tasks where documents are frequently added or updated. The main cost for EAR is the time for sampling query expansions. However, this can potentially be reduced by speed-up toolkits that optimize the inference time of transformers, such as FasterTransformer [4] (3.8~13× speedup for decoding) or FastSeq (Yan et al., 2021; 7.7× speedup for BART decoding). Moreover, we can leverage model distillation (Shleifer and Rush, 2020) and quantization (Li et al., 2022) for transformers. We leave these directions for future work.

8 Related Work

Query Expansion and Reformulation Traditionally, query expansion methods based on pseudo relevance feedback utilize relevant context without external resources to expand queries (Rocchio, 1971; Jaleel et al., 2004; Lv and Zhai, 2010; Yu et al., 2021). Recent studies attempt to reformulate queries using generative models, relying on external resources such as search sessions (Yu et al., 2020) or conversational contexts (Lin et al., 2020; Vakulenko et al., 2021), or involve sample-inefficient reinforcement learning training (Nogueira and Cho, 2017). More recently, GAR (Mao et al., 2021a) explored the use PLMs for query expansion instead of external resources. A concurrent study (Liu et al., 2022) generates multiple expansions with beam search and filters and fuses the results, but EAR is aware of the BM25 retriever and could select more promising query expansions and run fewer BM25 retrievals.

Retrieval for OpenQA Sparse retrieval with lexical features such as BM25 was first explored for OpenQA (Chen et al., 2017). Dense retrieval methods were shown to outperform sparse methods (Karpukhin et al., 2020; Guu et al., 2020), while requiring large amounts of annotated data and much more compute. Although powerful, dense retrievers often fall short in the scenarios of 1) requiring lexically exact matching for rare entities (Sciavolino et al., 2021) and 2) out-of-domain generalization (Reddy et al., 2021). For 1), Luan et al. (2021) proposed a sparse-dense hybrid model, and Chen et al. (2021) trained a dense retriever to imitate a sparse one. For 2), Ram et al. (2022) created a pre-training task for dense retrievers to improve zero-shot retrieval and out-of-domain generalization. Another recent line of research explores passage reranking with PLMs to improve performance for both sparse and dense methods. Nogueira and Cho (2019) first explored BERT-based supervised rerankers for standard retrieval tasks and Mao et al. (2021b) proposed reranking by reader predictions without any training. Sachan et al. (2022) attempt to use an LLM directly as the reranker, but it requires huge amounts of computation at inference time and underperforms fine-tuned rerankers.

9 Conclusion

We propose EAR, which couples GAR and BM25 together with a query reranker to unlock the potential of sparse retrievers. EAR significantly outperforms DPR while inheriting the advantage of BM25: *fast indexing time* and *small index size* compared to the compute-heavy DPR. Cross-dataset evaluation also shows that EAR is very good at generalizing to out-of-domain examples. Furthermore, we demonstrate that contributions of EAR and passage reranking are complementary, and using both methods together leads to superior results. Overall, EAR is a promising alternative to existing dense retrieval models, providing a new way to achieve high performance with less computing resources.

Limitations

First, as EAR largely relies on GAR generators, the performance of the method is closely tied to the quality of the generator used. We have attempted to use large language models such as T0-3B without fine-tuning as a replacement for the GAR generator during testing, but the performance becomes worse. The main reason is that the quality of query expansions generated by T0-3B is too diverse, which makes EAR has a higher chance to select from terrible expansions. In contrast, the output quality of GAR is more stable. We may need a more complex mechanism that can exclude terrible query expansion if we want to directly use the query expansions generated by T0-3B during inference. Second, EAR has demonstrated a strong generalization ability to out-of-domain data, but the method may still face challenges when transferring to other languages without any supervised QA data, which GAR and EAR are trained on. Although challenging, we are still trying to train the EAR system without supervised QA data.

[4] https://github.com/nvidia/fastertransformer

Ethics Statement

In this research, we used publicly available datasets and we did not collect any personal information. Our method is designed to improve the performance of information retrieval systems, which can have a positive impact on various applications, such as search engines, QA systems, and other applications that rely on text retrieval. When deployed, however, our approach also poses the ethical risk typical of pre-trained language models, for instance, producing retrieval results that contain human biases which could potentially exacerbate discrimination. Therefore, caution should be exercised before implementing our approach in real-world situations and thorough audit of training data and testing of model outputs should be conducted.

Acknowledgements

We thank Yuning Mao for his helpful feedback. We thank Ori Ram for providing detailed results of the experiments from Ram et al. (2022). This research was sponsored by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Petr Baudiš and Jan Šedivỳ. 2015. Modeling of the question answering task in the yodaqa system. In *International Conference of the cross-language evaluation Forum for European languages*, pages 222–228. Springer.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Michele Bevilacqua, Giuseppe Ottaviano, Patrick Lewis, Scott Yih, Sebastian Riedel, and Fabio Petroni. 2022. Autoregressive search engines: Generating substrings as document identifiers. In *Advances in Neural Information Processing Systems*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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.

Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37, Online. Association for Computational Linguistics.

Xilun Chen, Kushal Lakhotia, Barlas Oğuz, Anchit Gupta, Patrick Lewis, Stan Peshterliev, Yashar Mehdad, Sonal Gupta, and Wen-tau Yih. 2021. Salient phrase aware dense retrieval: Can a dense retriever imitate a sparse one? *arXiv preprint arXiv:2110.06918*.

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.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Nasreen Abdul Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard. In *Text Retrieval Conference*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

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 Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

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, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Zheng Li, Zijian Wang, Ming Tan, Ramesh Nallapati, Parminder Bhatia, Andrew Arnold, Bing Xiang, and Dan Roth. 2022. Dq-bart: Efficient sequence-to-sequence model via joint distillation and quantization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 203–211.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, pages 2356–2362.

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

Linqing Liu, Minghan Li, Jimmy Lin, Sebastian Riedel, and Pontus Stenetorp. 2022. Query expansion using contextual clue sampling with language models. *arXiv preprint arXiv:2210.07093*.

Yixin Liu and Pengfei Liu. 2021. SimCLS: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345.

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*, SIGIR '10, page 579–586, New York, NY, USA. Association for Computing Machinery.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021a. Generation-augmented retrieval for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4089–4100, Online. Association for Computational Linguistics.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021b. Reader-guided passage reranking for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 344–350, Online. Association for Computational Linguistics.

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.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Ori Ram, Gal Shachaf, Omer Levy, Jonathan Berant, and Amir Globerson. 2022. Learning to retrieve passages without supervision. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2687–2700, Seattle, United States. Association for Computational Linguistics.

Revanth Gangi Reddy, Vikas Yadav, Md Arafat Sultan, Martin Franz, Vittorio Castelli, Heng Ji, and Avirup Sil. 2021. Towards robust neural retrieval models with synthetic pre-training. *arXiv preprint arXiv:2104.07800*.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

J. J. Rocchio. 1971. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval system - experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall.

Devendra Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen-tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2022. Improving passage retrieval with zero-shot question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3781–3797, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

G. Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, et al. 2021. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Christopher Sciavolino, Zexuan Zhong, Jinhyuk Lee, and Danqi Chen. 2021. Simple entity-centric questions challenge dense retrievers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6138–6148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sam Shleifer and Alexander M Rush. 2020. Pretrained summarization distillation. *arXiv preprint arXiv:2010.13002*.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question rewriting for conversational question answering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 355–363.

Ellen M Voorhees et al. 1999. The trec-8 question answering track report. In *Trec*, volume 99, pages 77–82. Citeseer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yu Yan, Fei Hu, Jiusheng Chen, Nikhil Bhendawade, Ting Ye, Yeyun Gong, Nan Duan, Desheng Cui, Bingyu Chi, and Ruofei Zhang. 2021. Fastseq: Make sequence generation faster. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 218–226.

HongChien Yu, Chenyan Xiong, and Jamie Callan. 2021. Improving query representations for dense retrieval with pseudo relevance feedback. In *Proceedings of the 30th ACM International Conference on Information &; Knowledge Management*, CIKM '21, page 3592–3596, New York, NY, USA. Association for Computing Machinery.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, page 1933–1936, New York, NY, USA. Association for Computing Machinery.

A Dataset Statistics

We show the number of train/dev/test examples in each dataset in Table 9.

Dataset	Train	Dev	Test
Natural Questions	58,880	8,757	3,610
TriviaQA	60,413	8,837	11,313
WebQuestions	-	-	2,032
TREC	-	-	694
EntityQs	-	-	22,075

Table 9: Number of train/dev/test examples in each dataset.

B Training Details

For the training set, we use T0-3B [5] to randomly sample 50 query expansions per query. For the dev set and test set, we use the three GAR generators (answer/sentence/title), which are BART-large seq2seq models (Lewis et al., 2020) to generate 50 query expansions per query. We use the DeBERTa V3 base model[6], which has 86M parameters that are the same as BERT-base (Devlin et al., 2019), as EAR-RI and EAR-RD rerankers. For the implementation of rerankers, we reference the implementation of SimCLS (Liu and Liu, 2021)[7], which also does reranking for sequences. We start from the code of SimCLS and change the loss function to our ranking loss $\mathcal{L}_{\text{Rank}}$. During training, we use the dev set generated from three GAR generators to pick the best checkpoints, resulting in three different reranker models corresponding to the answer/sentence/title generators.

The ranges we search for our hyperparameters are shown in Table 10. Each training example in our dataset contains 50 sequences (generated by T0-3B). To prevent memory issues of GPU, we used gradient accumulation to simulate a batch size of 4 or 8, which effectively consists of 200 or 400 sequences, respectively.

The training time on a single NVIDIA V100 GPU is around 12 hours for EAR-RI and 1 to 2 days for EAR-RD. The best hyperparameters according to the dev set are shown in Table 11. However, in our experiments, the variance between different hyperparameters is actually quite small.

Hyperparams	Range
MAX_RANK	[101, 250]
Batch size	[4, 8]
Learning rate	[2e-3, 5e-3]
Epochs (EAR-RI)	2
Epochs (EAR-RD)	3
Max length (EAR-RI)	64
Max length (EAR-RD)	256

Table 10: The range for hyperparameter search. The definition of MAX_RANK is shown in Section 3.1.

Hyperparams	answer	sentence	title
NQ: EAR-RI			
MAX_RANK	101	250	101
Batch size	8	8	4
Learning rate	5e-3	2e-3	2e-3
NQ: EAR-RD			
MAX_RANK	101	101	250
Batch size	4	4	8
Learning rate	2e-3	2e-3	2e-3
TriviaQA: EAR-RI			
MAX_RANK	101	101	101
Batch size	8	8	8
Learning rate	2e-3	2e-3	2e-3
TriviaQA: EAR-RD			
MAX_RANK	101	101	101
Batch size	8	8	8
Learning rate	2e-3	2e-3	2e-3

Table 11: The best hyperparameters for NQ and TriviaQA dev sets.

C Passage Reranking

For the implementation of a BERT-based passage reranker, we generally follow the setting of (Nogueira and Cho, 2019) for training. We separately fine-tuned two `bert-base-uncased` models on the NQ training set and the TriviaQA training set. Each pre-trained BERT model is fine-tuned for reranking using cross-entropy loss on the binary classification head on top of the hidden state corresponding to the [CLS] token. We use the top-10 outputs of BM25 ran on the training sets as the training examples, which contains both positive and negative examples. We fine-tune the models using 2 GPUs with mixed precision (fp16) with a batch size of 128 for 3 epochs. AdamW (Loshchilov and Hutter, 2018) is used for optimization with a learning rate of 5e-5, linear warmup over the first 10k steps and linear decay afterwards, and a weight decay of 0.01.

[5] https://huggingface.co/bigscience/T0_3B

[6] https://huggingface.co/microsoft/deberta-v3-base

[7] https://github.com/yixinL7/SimCLS

D Inference Details

For inference of GAR retrieval results, we follow GAR to retrieve with three queries generated by three context generators (answer/sentence/title), and then fuse the three retrieved passages lists in the order of sentence, answer, title. In other words, given the three retrieved lists of passages: $(a_1, a_2, ..., a_{100})$, $(s_1, s_2, ..., s_{100})$, $(t_1, t_2, ..., t_{100})$, we fuse the results as $(s_1, a_1, t_1, s_2, a_2, t_2, ..., s_{33}, a_{33}, t_{33}, s_{34})$. We skip all the duplicated passages that exist twice during the fusion process.

For EAR, we use the same pipeline of GAR, while the only difference is that instead of greedy decoding, now the three generators of GAR can do random sampling, and three different query rerankers (answer/sentence/title) are applied to select the best queries. After that, the pipeline to obtain retrieval results is exactly the same as GAR.

To fairly compare the latency of these methods, we run the 3610 queries in NQ test set one-by-one without batching (batch size = 1) and compute the average the latency per query, where document encoding, query expansion and reranking are run on NVIDIA RTX A5000 GPUs and indexing and retrieval are run on fifty Intel Xeon Gold 5318Y CPUs @ 2.10GHz, for both FAISS (Johnson et al., 2019) (DPR) and Pyserini (Lin et al., 2021) (BM25).

DPR document indexing We used 4 GPUs to encode 21M Wikipedia passages in parallel with mixed precision (fp16), which takes around 3.5 hours.

GAR and EAR For inference of GAR and EAR, answer/sentence/title generators/rerankers are run in parallel on three GPUs.

FiD We take the public checkpoints of FiD[8], which are trained from T5-Large (?) with NQ/TriviaQA, to directly evaluate the end-to-end QA performance.

E Qualitative Study

In this section, we aim to investigate the differences between queries generated by GAR and EAR. We first look at the lengths of the expanded queries for GAR, EAR-RI, EAR-RD. In general, the lengths of queries from EAR are slightly shorter than that of GAR, but the trends are not very obvious. Thus, we conduct a qualitative study to see what is the difference between these queries.

Model	answer	sentence	title
Original Query		9.2	
GAR	13.3	38.8	32.3
EAR-RI	13.1	36.2	29.3
EAR-RD	13.2	38.2	28.8

Table 12: Lengths of the expanded queries in words for different methods on NQ test set.

As shown in Table 13, we provide three examples to demonstrate how our method EAR works. In the first example, the initial query only includes two keywords, "Deadpool" and "released," that can match the answer passage. As a result, the BM25 algorithm is unable to retrieve the correct passage within the top results until the 77th passage. The greedy decoding output for GAR also fails to retrieve the correct passage, as it includes many irrelevant name entities. However, both EAR-RI and EAR-RD are able to select useful outputs from GAR, which contain keywords such as “scheduled,” “2018,” “Leitch,” and “in the United States.” Although none of these keywords contains the real answer *May 18, 2018*, these keywords already provide enough lexical overlap with the answer passage, allowing BM25 to correctly retrieve the answer passage in the top-1 result.

For the second example, the original query only contains three keywords “India’s,” “next,” and “star” that can match the answer passage, so BM25 with the original query cannot retrieve the correct passage within the top retrieved results until the 96th passage. For GAR, the greedy decoding output for GAR is also not effective, as it is a misleading answer and only includes one useful keyword “winner” and thus cannot retrieve the correct passage within the top-100 results. For EAR-RI and EAR-RD, they are able to select a sentence that, while not containing the correct answer “Natasha Bharadwaj” or “Aman Gandotra,” does include useful keywords such as “winner,” “Superstar,” “Season,” and “2018.” These keywords provide enough lexical overlap with the answer passage, allowing EAR-RI and EAR-RD to correctly retrieve the answer passage in the top-1 result.

The third example presents a challenging scenario. The initial query only includes two common keywords, “method” and “writer,” which is difficult to match the answer passage. While BM25 is able to correctly retrieve the answer at the 92nd

[8] https://github.com/facebookresearch/FiD

Model	Query [*Answer = May 18, 2018*]	Answer Rank
BM25	*When is the next* ***Deadpool*** *movie being* ***released****?*	77
GAR	*When is the next* ***Deadpool*** *movie being* ***released****? Miller Brianna Hildebrand Jack Kesy Music by Tyler Bates Cinematography Jonathan Sela Edited by Dirk Westervelt...*	>100
EAR-RI	*When is the next* ***Deadpool*** *movie being* ***released****? Deadpool 2 is scheduled to be released on May 26, 2018, with Leitch directing.*	1
EAR-RD	*When is the next* ***Deadpool*** *movie being* ***released****? The film is scheduled to be released on March 7, 2018, in the United States.*	1
	Answer Passage	
	*"****Deadpool*** *2" premiered at Leicester Square in London on May 10, 2018. It was* ***released*** *in the United States on May 18, 2018, having been previously scheduled for release on June 1 of that year. Leitch ś initial cut of the film was around two hours and twelve minutes, ...*	

Model	Query [*Answer = Natasha Bharadwaj, Aman Gandotra*]	Answer Rank
BM25	*Who has won* ***India's next*** *super* ***star****?*	96
GAR	*Who has won* ***India's next*** *super* ***star****? The winner of the competition is 18 year-old Mahesh Manjrekar from Mumbai.*	>100
EAR-RI	*Who has won* ***India's next*** *super* ***star****? The winner of the Superstar Season 2018 is Siddharth Shukla.*	1
EAR-RD	*Who has won* ***India's next*** *super* ***star****? The winner of the Superstar Season 2018 is Siddharth Shukla.*	1
	Answer Passage	
	India's Next *Superstars (INS) is an Indian talent-search reality TV show, which premiered on* ***Star*** *Plus and is streamed on Hotstar. Karan Johar and Rohit Shetty are the judges for the show. Aman Gandotra and Natasha Bharadwaj were declared winners of the 2018 season...*	

Model	Query [*Answer = Anthropomorphism, Pathetic fallacy, Hamartia, Personification*]	Answer Rank
BM25	***Method*** *used by a* ***writer*** *to develop a character?*	92
GAR	***Method*** *used by a* ***writer*** *to develop a character? Developing a character is a technique employed by writers in the creation of a narrative.*	>100
EAR-RI	***Method*** *used by a* ***writer*** *to develop a character? Developing a character is the primary method employed by writers in the creation of a fictional character.*	>100
EAR-RD	***Method*** *used by a* ***writer*** *to develop a character? Developing a character is a technique employed by writers in terms of establishing a persona and building a relationship between the reader and the character.*	>100
	Answer Passage	
	The intensive journal ***method*** *is a psychotherapeutic technique largely developed in 1966 at Drew University and popularized by Ira Progoff (1921-1998). It consists of a series of writing exercises using loose leaf notebook paper in a simple ring binder, divided into sections to helping accessing various areas of the* ***writer****'s life. These include a dialogue section for the personification of things, a "depth dimension" to aid in accessing the subconscious and other places for ....*	

Table 13: Examples that show the difference between BM25/GAR/EAR-RI/EAR-RD. Words in blue are query expansions generated by GAR. **Bold words** are useful keywords from the original query. Words highlighted in green are useful keywords generated by GAR. **Answer Rank** shows the ranking of the answer passage in the retrieval results.

passage, the generated query expansions are not helpful and instead are misleading, resulting in GAR and EAR-RI/EAR-RD all unable to retrieve the correct passage within the top-100 results due to the distracting query expansions. This example illustrates the importance of the GAR generators. If all of the generated query expansions are not useful, EAR is unable to improve the results.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations before Reference

☑ A2. Did you discuss any potential risks of your work?
Ethics Statement before Reference

☑ A3. Do the abstract and introduction summarize the paper's main claims?
In the Abstract

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

□ B1. Did you cite the creators of artifacts you used?
No response.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

□ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix B

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix B

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Appendix B

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix B

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.