Detecting Frozen Phrases in Open-Domain Question Answering

Mostafa Yadegari
yadegari@ualberta.ca
University of Alberta
Edmonton, Alberta, Canada

Ehsan Kamalloo
kamalloo@ualberta.ca
University of Alberta
Edmonton, Alberta, Canada

Davood Rafiei
drafiei@ualberta.ca
University of Alberta
Edmonton, Alberta, Canada

ABSTRACT

There is essential information in the underlying structure of words and phrases in natural language questions, and this structure has been extensively studied. In this paper, we study one particular structure, referred to as *frozen phrases*, that is highly expected to transfer as a whole from questions to answer passages. Frozen phrases, if detected, can be helpful in open-domain Question Answering (QA) where identifying the localized context of a given input question is crucial. An interesting question is if frozen phrases can be accurately detected. We cast the problem as a sequence-labeling task and create synthetic data from existing QA datasets to train a model. We further plug this model into a sparse retriever that is made aware of the detected phrases. Our experiments reveal that detecting frozen phrases whose presence in answer documents are highly plausible yields significant improvements in retrievals as well as in the end-to-end accuracy of open-domain QA models.

CCS CONCEPTS

• **Information systems** → **Question answering**; **Query reformulation**.

KEYWORDS

Open-domain question answering, Information retrieval, Frozen phrases, Sparse retriever, Question paraphrasing

ACM Reference Format:
Mostafa Yadegari, Ehsan Kamalloo, and Davood Rafiei. 2022. Detecting Frozen Phrases in Open-Domain Question Answering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3477495.3531793

1 INTRODUCTION

The structure of words and their relationships in a natural language text has been extensively studied in NLP, and those studies have led to tools and techniques for parsing and syntactic analysis that are in use inside many applications today. Phrase-based representation also has a long history in IR with both syntactic and statistical dependencies between words considered for querying and indexing [6, 41], text categorization [11, 20], etc. In this paper, we study a particular structure, referred to as "frozen phrases," which is highly expected to transfer *as a whole* from questions to answer passages. Detecting such structures has major implications in retrieval, especially in Open-domain Question Answering (OpenQA), which aims at answering factoid questions over an enormous collection of text documents. Recent OpenQA models often follow a two-stage framework that consists of a retriever to find candidate documents, and a reader to extract answers from retrieved candidates [2, 15].

Retrieval has undoubtedly a profound role in OpenQA mainly because the overall performance of the pipeline is arguably bounded by the performance of the retriever component [18, 27, 46]. Sparse retrieval models such as BM25 have been a popular choice for retriever [2, 3, 39, 44]. However, sparse representations are not designed to reflect the importance of phrases in the question vector. Consider the question "*Who wrote the country song I Can Only Imagine?*", taken from a well-known OpenQA dataset, Natural Questions-open [19]. "*I Can Only Imagine*" is the name of a song that has a high chance of matching verbatim in an answer document although its terms might even have lower IDFs than that of the rest of the question "*Who wrote the country song*."

In this paper, we characterize such phrases that are expected to appear in the target document as exactly as they are written in the question as Frozen Phrases. The terms of a frozen phrase are extremely likely to be seen together in the target document, no matter what their TF-IDF scores are in the question vector, and the ordering of the terms is expected to match closely. For instance, in the question "*who said one man's vulgarity is another's lyric*" [17], the phrase "*one man's vulgarity is another's lyric*" is a famous quote, which is a frozen phrase in the question. However, not all frozen phrases are expected to be helpful in retrieval. In the above example, the phrase "*who sings*" may also be a frozen phrase, but that phrase is less likely to be helpful because it is likely to appear in any arbitrary document.

Detecting frozen phrases can be helpful in many applications including query expansion [5, 25, 47] and question clustering [8, 14]. In query expansion, adding more terms to a question generally shrinks the weights of the terms in the original question including those of a frozen phrase, and this can negatively impact the retrievals. If the frozen phrases can be detected, query expansion can be better guided to leave the invariant parts of a question unchanged. Detecting frozen phrases and their types (e.g. a song lyric) in a question can also provide more insight about the focal point of the question, which can help with further question classification.

In this paper, we tackle the task of detecting frozen phrases in questions using a transformer based model [38]. A major challenge in training one such classifier is the absence of large enough annotated training data; hence, we propose an algorithm to automatically generate the training data based on existing QA corpora where questions, answer documents, and corpus statistics — e.g., TF, and IDF — are readily available.

SIGIR '22, July 11–15, 2022, Madrid, Spain

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3531793

To evaluate the performance of our proposed model and the quality of our generated training set, we use our model to predict frozen phrases in an unseen test set. The queries in the test set are expanded by the detected frozen phrases and are retrieved by a sparse retriever. Our empirical results show a significant performance boost that effectively underscores the usefulness of our devised strategy in identifying frozen phrases. Our code and data are released at https://github.com/Aashena/Frozen-Phrases.

Our contributions can be summarized as follows:

(1) We introduce frozen phrases to capture the invariant parts of a question and show its importance in question answering.
(2) We propose an algorithm to extract frozen phrases from natural questions that have an answer document, allowing us to create a training data from a QA corpus.
(3) Through our evaluation, we show that frozen phrases can be successfully detected using our generated training data, and our model is able to improve upon retrieval in question answering.

2 RELATED WORK

Retrieval in OpenQA. Sparse retrieval models such as TF-IDF and BM25 have been widely adopted in both early multi-stage OpenQA pipelines [4] and modern retriever-reader models [2, 3, 39, 40, 44]. However, they often suffer from the so-called *vocabulary mismatch* bottleneck [23]. As a remedy, several models [18, 27] offer an intermediate stage to re-rank the initial retrieved results via a neural model. Doc2query [29, 30] generates potential relevant queries for each passage in the corpus and stores them along with the passages. More recently, dense retrieval models [15, 16, 19, 33, 43] and retrieval-augmented models [10, 21] have become popular. However, these models often struggle with entity-centric questions for which sparse retrievers usually work well [1] and also, generalize poorly to new domains without supervision [12, 37].

Query Expansion/Reformulation. Another strategy to tackle vocabulary mismatch is query expansion [35] where the question is augmented with supplementary text to boost the matching likelihood. One strategy is to generate question paraphrases [7], which we also exploit in our model, but we ensure that frozen phrases are preserved. Similarly, other useful content, if available, may be added too. GAR [25] expands questions with automatically generated sentences to enrich question with clues—e.g., expected answers or sentences containing an answer—that are fetched from a pretrained language model. Alternatively, Nogueira and Cho propose a reinforcement learning approach that uses post-retrieval signals as a reward function to reformulate the query [28].

Question Decomposition. For complex questions such as multihop questions [45], decomposing them into simpler sub-questions is a known technique [26, 31, 42]. Our method is analogous to these methods in that we also detect subsequences in questions. However, our objective is inherently different as we find sequences that are expected to match answer documents verbatim. Also, Qi et al. propose an iterative strategy that uses a semantic overlap method to find potentially relevant documents based on the retrieval context at each step [32]. Specifically, the overlap is computed via a longest common subsequence/substring algorithm between a target document and the current context. We also employ a similar approach to align questions with their corresponding answer document.

3 METHODOLOGY

The problem of detecting frozen phrases in questions can be cast as a sequence labelling problem. Given a word sequence $w_1, \ldots, w_n$, denoting a question, we seek to find sub-sequences $w_i, \ldots, w_j$ of consecutive terms such that $w_i, \ldots, w_j$ is expected to transfer as a whole to the answer document.

For example, given the question "*Who sang I ran all the way home*," from NQ-open [17], we want to identify the song title "*I ran all the way home*" as a frozen phrase. Clearly, detecting phrases with a low selectivity is more desirable since these phrases are less likely to appear in arbitrary non-answer documents and they can be more effective in retrievals.

A major challenge in training a model to detect such phrases is the lack of annotated data for this purpose. It may seem at first that frozen phrases can be annotated in a QA corpus by leveraging the Longest Common Sub-sequence (LCS) between a question and its answer document. However, given that answer documents are long, every question term is likely to appear somewhere in the answer passage and will be included in an LCS. Such sequences may not really form a phrase and are not the subject of our study.

The Longest Common Sub-string (LCStr) may be considered as an alternative for annotating frozen phrases. However, our experiments show that LCStr misses many phrases that do not *exactly* transfer to the answer due to minor differences such as misspelling (see Section 4.2.1 for our evaluation of LCStr).

Inspired by the Smith-Waterman (SW) local alignment algorithm [36], we align the words sequence of a question with its answer document and extract frozen phrases from the question.

Let Q and X respectively denote the word sequences of a question and its answer document. Given the word sequences of a question Q and its answer X, let $H_{i,j}$ be the maximum alignment score between their prefixes $Q_1, \ldots Q_j$ and $X_1, \ldots, X_i$. We define the scoring function as:

$$H_{0,0} = 0\ ,\ H_{i,0} = -i.W_X\ ,\ H_{0,j} = -j.W_Q,$$

$$H_{i,j} = \max \begin{cases} H_{i-1,j-1} + S_{i.j}(X, Q) \\ H_{i-1,j} - W_X \\ H_{i,j-1} - W_Q, \end{cases}$$

where W_X and W_Q are the gap penalties in the document text and the question respectively, and $S_{i.j}(X, Q)$ is the matching score of X_i and Q_j, which is defined as follows:

$$S_{i.j}(X, Q) = \begin{cases} IDF_{uni}(X_i) & \text{if } X_i{=}Q_i \wedge X_{i-1}{\neq}Q_{i-1}, \\ IDF_{uni}(X_i) + IDF_{bi}(X_{i-1}.X_i) & \text{if } X_i{=}Q_i \wedge X_{i-1}{=}Q_{i-1}, \\ -\infty & \text{if } X_i{\neq}Q_i, \end{cases} \tag{1}$$

where $IDF_{uni}(.)$ and $IDF_{bi}(.)$ respectively denote the IDFs of a unigram and a bigram. Using the IDF of a term helps us to assign higher scores to the terms with low selectivity, and using bigram IDF boosts the score of longer phrases. $S_{i.j}(X, Q)$ is set to $-\infty$ when X_i and Q_i are not equal to force the alignment to consider gaps.

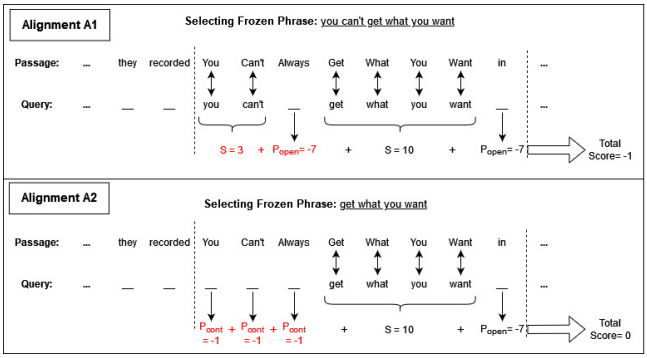

Figure 1: An illustration of a problem with the alignment algorithm. Some part of the frozen phrase are dropped when the opening gap penalty is not distributed over multiple gaps (Alignment A2 will be chosen over A1). S **denotes the matching score of a phrase, defined in Eq.** (1). P_{cont} **and** P_{open} **are the continuing gap penalty and the opening gap penalty, respectively. Here, we set** P_{cont} **to** -1 **and** P_{open} **to** -7**. The matching scores of "*you can't*" and "*get what you want*" are 3 and 10, respectively.**

The choice of a gap penalty is important on how the phrases are formed. A phrase that is perfectly transferred to an answer will not have gaps but often the wording of a question has extra terms, for example, due to misspellings, or misses out terms that appear in the answer. We are not expecting many extra terms in question phrases, hence we set W_Q to a constant.

However, the gap structure in the answer documents is a bit more complex. For example, a question that refers to a song title can miss out a few words. Similar observations are made in detecting molecular sub-sequences where the gap penalty is divided into an opening gap penalty P_{open} and a continuing gap penalty P_{cont}, with $P_{cont} < P_{open}$. For example, Smith et al. set P_{open} to 1.33 and P_{cont} to 0.33 [36].

Generally, as the length of a frozen phrase increases, the chance that a user misses out words or writes them inaccurately in the question also increases. That inaccuracy breaks the chain of the matching words in the frozen phrase. For example, consider the phrase "*You Can't Always Get What You Want*," the name of a song by The Rolling Stones rock band, that is mentioned in a document, but a question refers to it as "*you can't get what you want.*" There is a mismatch in the middle of the phrase, with *always* dropped in the question. Figure 1 illustrates two possible alignments of the phrases mentioned above. A1: (`"You Can't"`, `"you can't"`), (`"Always",-`), (`"Get What You Want","get what you want"`), and A2: (`"You Can't Always",-`), (`"Get What You Want","get what you want"`). The terms "*you*" and "*can't*" and the phrase "*you can't*" are expected to have low IDF since they are common terms, and it is likely that the matching score of "*you can't*", Eq. (1), minus the opening gap penalty (applied after) to be less than the gap penalty $-3.P_{cont}$. That means A2 will be selected over A1, and the first part of the phrase will be ignored.

To avoid such cases, we need to distribute the opening penalty in the first few gaps instead of applying it all at once. We need the opening gap to increase proportional to the likelihood of having longer gaps inside a phrase.

Consider a Question $Q = \{Q_1, \ldots, Q_l\}$ of length l and let X denote the term sequence of the answer document. Let $Q_{i+1}, \ldots, Q_{i+k}$ be a sub-sequence of Q of length k such that the preceding term Q_i and the following term Q_{i+k+1} are matched with terms in X but the terms in $Q_{i+1}, \ldots, Q_{i+k}$ are not matched. Suppose π_k denotes the probability of not having Q_i and Q_{i+k+1} in the same frozen phrase; we want our opening gap penalty to increase proportional to π_k. If we denote with t the smallest positive integer where $\pi_t \approx 1$, then the general formula for W_X can be expressed as:

$$W_X(k) = \begin{cases} P_{cont} & (k > t) \\ \frac{\pi_k . P_{open}}{\sum_{i=1}^{t} \pi_i} & (0 < k \leq t). \end{cases}$$

As for setting the values of P_{cont} and P_{open}, one can leverage the following formula:

$$\sum_{k=0}^{n} S(X_{i+k}, Q_{j+k}) - P_{open} > -(n+t) \times P_{cont}, \tag{2}$$

where $Q_j, Q_{j+1}, ..., Q_{j+n}$ and $X_i, X_{i+1}, ..., X_{i+n}$ are respectively question and document phrases that are matched. For good phrases, we want the above inequality to be satisfied and those phrases to be selected, and for bad phrases, we want the inequality not to be satisfied and the phrases to be ignored.

The annotation task, implemented using a dynamic programming algorithm, maps each question to a sequence of labels "`SEQ`" and "`O`", with "`SEQ`" indicating the terms that are part of a frozen phrase and "`O`" indicating the terms that are not. Using this procedure, we can create automatically annotated silver data on top of existing QA datasets.

Finally, we train a sequence-labelling model on the silver data. The model is employed in predicting frozen phrases for arbitrary questions and retrieving answer documents from a corpus. Table 1 provides examples of the terms extracted by the trained model and the alignment algorithm.

4 EXPERIMENTS

In our evaluation, we seek to answer the following questions:

(1) how informative the frozen phrases extracted by our alignment algorithm are in terms of their recall in open-domain QA and how much information is lost by only keeping such phrases,
(2) how effective the predicted frozen phrases are in improving the performance of sparse/dense retrievers and what role they play in query expansion, and
(3) if the end-to-end performance of open-domain QA is improved by leveraging the predicted frozen phrases during retrieval.

4.1 Experimental Setup

4.1.1 Dataset. To generate our training data, we ran our alignment algorithm, discussed in Section 3, on the training set of the Natural Questions (NQ) dataset [17]. Our testing was done on the development set of NQ-open dataset [19] that consists of 3,610 questions. We excluded the questions that NQ-open did not have their answer

Table 1: Examples of the frozen phrases derived by our alignment algorithm (Silver Standard) vis-a-vis the extracted phrases by the model (Prediction), trained on the silver data.

Silver Standard (our proposed alignment)	Prediction
hazels boyfriend in the fault in our stars	hazels boyfriend in the fault in our stars
when does the day of the dead end	when does the day of the dead end
where is the citrus bowl held this year	where is the citrus bowl held this year
what year does the quiet man take place	what year does the quiet man take place
how many seasons of rules of engagement is there	how many seasons of rules of engagement is there
who plays dusty in the movie pure country	who plays dusty in the movie pure country
how tall is the actor who plays hagrid in harry potter	how tall is the actor who plays hagrid in harry potter

document annotated in NQ dataset, reducing the test set to 3072 questions.

4.1.2 Alignment Hyperparameters. We randomly selected a small subset of our training data, on which we manually observed the alignment algorithm output. Since π_t is the probability at which two adjacent phrases do not form a frozen phrase, we can estimate it based on our statistical observation of the subset. We start with a sequence length 1 and increase the length to t where $\pi_t \approx 1$ ($\pi_k < \pi_{k+1}$ for every k). For setting P_{opent} and P_{cont}, we leverage some negative and positive samples from the selected subset and find a setting where Eq. (2) is likely to hold for positive examples and it is less likely to hold for negative examples.

We want to assign a small value for W_Q (0 or close to 0) because we want the gap penalty in the question to be small to encourage the algorithm to ignore the terms that are not in a frozen phrase. A high value for W_Q forces the algorithm to select as many terms as it can from the question, thus reducing our algorithm to LCS. Throughout our experiments, we used the following hyperparameters for the alignment algorithm:

$$P_{cont} = 1,\ P_{open} = 7,\ W_Q = 0.1,\ t = 3,\ \pi_1 = 0.25,\ \pi_2 = 0.5,\ \pi_3 = 1$$

4.1.3 Sequence-labelling Model. To predict frozen phrases, we fine-tuned a pre-trained $\text{RoBERTa}_{\text{base}}$ model [24] with a token classification head[1] on our silver training data. The output of the model is a sequence of frozen phrases with possible gaps. Those gaps are replaced with an out-of-vocabulary term to avoid forming bigrams that are not in the original question. We refer to these generated questions as Frozen Phrase Questions (FPQ). We trained our model with a learning rate of $1.0e^{-4}$, a batch size of 64, and early stopping for 70 epochs.

4.1.4 OpenQA Pipeline. For sparse retrieval, we adopted the BM25 implementation from Pyserini [22]. Retrieval is done over passages of 100 words that are derived from Wikipedia, following [15]. As reader, the base model of Fusion-in-Decoder [13] was used. The top 100 retrieved passages are fed into the reader, as done in [13]. For our query expansion, a T5 transformer [34], fine-tuned on the Quora quesion paraphrase dataset, was used to generate up to 10 paraphrases for each question[2]. The paraphrases are concatenated with the original question with an out-of-vocabulary term added between the concatenated questions to avoid undesirable bigrams. We refer to the new expanded queries as *Orig10Par*.

4.1.5 Evaluation Metrics. As a measure of the informativeness of the frozen phrases, we introduce a metric based on the overlap between the terms in extracted frozen phrases and the title of answer document. Retrievers often assign higher weights to a match in the title than a match in the document body [9]. Moreover, since the document title is already prepended to the passages in the corpus, retaining the title terms becomes important in questions. Hence, we measure title recall, defined as the fraction of title terms that are preserved in a question (either FPQ or original question), i.e.,

$$Recall_{title} = \frac{\sum_{m=1}^{M} |Q^m \cap T^m|}{\sum_{m=1}^{M} |T^m|},$$

where M, Q^m, and T^m are the number of samples in the dataset, the question of the m-th sample, and the title of the m-th sample respectively. We consider an FPQ informative if its $Recall_{title}$ is close to that of the original question, meaning any loss is minimal.

In addition to $Recall_{title}$, we use top-k retrieval accuracy and Exact match (EM) to evaluate our retriever and reader, respectively. The two metrics are widely used in prior work [15, 25].

4.2 Results and Discussions

4.2.1 Information Loss. To evaluate the informativeness of frozen phrases, we converted the set of questions in our test set to FPQs using our proposed alignment algorithm. The length of an FPQ is only 54% of the length of original questions on average. As a baseline for comparison, we also generated another dataset by replacing each question with its corresponding LCStr. The information loss is measured for both FPQ and LCStr using $Recall_{title}$.

As shown in Table 2, $Recall_{title}$ of our alignment algorithm (FPQ) is very close to that of the original questions (Orig). Even though the algorithm drops 46.3% of the question terms, it only drops 1% of the title terms. Moreover, the average IDF of the 1% dropped terms is quite low — i.e., 2.76 for unigrams and 6.2 for bigrams in our dataset.

[1] https://github.com/ThilinaRajapakse/simpletransformers

[2] https://github.com/ramsrigouthamg/Paraphrase-any-question-with-T5-Text-To-Text-Transfer-Transformer-

Table 2: $Recall_{title}$ and Passage level Accuracy to evaluate the Information loss

Dataset	$Recall_{title}$	Acc@1	Acc@10	Acc@100
Orig	0.48	**24.38**	**57.97**	**80.73**
LCStr	0.38	11.91	33.76	59.05
FPQ	0.47	24.12	56.35	78.91

As another measure of a possible information loss in retrieval, we also evaluated the performance of passage retrieval over the three query sets Orig, FPQ, and LCStr. As presented in Table 2, the retrieval accuracy for FPQ is higher than LCStr by a large margin, which shows how the algorithm is successfully selecting important terms. Furthermore, while we used almost half of the question terms, the performance declines only by around 2%. This confirms that the extracted frozen phrases play a significant role in retrieval.

4.2.2 Retriever Performance. To evaluate the performance of our frozen phrase prediction, we trained our transformer model on the data annotated by our alignment algorithm. The trained model was applied to the questions in our test set to generate FPQs.

To evaluate if the use of frozen phrases can improve the retrievals, we concatenated each FPQ to its original question in our test set to increase the weight of the predicted frozen phrases in the question vector. As shown in Table 3, adding frozen phrases to the questions (this is referred to as Orig+FPQ) significantly improves the accuracy. As a baseline for comparison, we also did a similar experiment but, instead of adding frozen phrases, we added named entities (Orig+NER) and singular nouns (Orig+NN) to increase their weight. Unlike frozen phrases, adding named entities (Orig+NER) and singular nouns does not improve the retrieval.

To evaluate the performance of our model prediction in query expansion, we generated up to 10 paraphrases for each question in our test set, using the method described in Section 4.1, and added those paraphrases to the original question (Orig10Par). Then, we added 10 copies of FPQs to the expanded query set and created a new set of questions (Orig10Par+10FPQ). We further added 10 copies of the original test questions to the expanded query (Orig10Par+10Orig) to see if our predictions are more helpful than the original questions for the expanded query. The results of the BM25 retriever on the aforementioned question sets as well as the original questions (Orig) are reported in Table 3.

The best result is achieved when 10 paraphrases and 10 copies of the FPQ are added to the original questions (Orig10Par+10FPQ), and the questions that only have the FPQ in addition to the original question (Orig+FPQ) stand second. These results show that we are increasing the weight of the right terms, and our frozen phrase extraction method improves the retrieval.

We also combine our enhanced BM25 retrievers with DPR [15], a prominent dense retriever, by taking a weighted mean of their retrieval scores, following [15]. The results are consistent with the previous results where we used only sparse retrieval.

4.2.3 End-to-End Performance. Finally, Table 4 shows the end-to-end exact-match accuracy of our models where the reader is applied to the two question sets that have the best retrieval performance as well as to the original set of test questions. We can observe that the performance boost in the retrieval translates to a better performance of the reader, and that the end-to-end improvement is statistically significant.

Table 3: Retrieval accuracy at top-k on different question sets. † denotes statistical significance (p-value < 0.01) over Orig for each retriever, BM25 and DPR+BM25.

Question Set	Acc@1	Acc@10	Acc@100
BM25 on Orig	24.38	57.97	80.73
BM25 on Orig+NER	24.61	58.69	80.37
BM25 on Orig+NN	23.568	58.07	80.18
BM25 on Orig+FPQ	25.65†	60.06†	82.13†
BM25 on Orig10Par	25.29	57.45	80.99
BM25 on Orig10Par+10Orig	25.16	58.89	81.61
BM25 on Orig10Par+10FPQ	**26.43**†	**60.25**†	**82.52**†
DPR on Orig	46.87	77.25	88.54
DPR+BM25 on Orig	48.80	78.81	89.20
DPR+BM25 on Orig+FPQ	**50.01**†	**79.07**	89.40
DPR+BM25 on Orig10Par+10FPQ	49.64	78.66	**90.10**†

Table 4: End-to-end exact-match accuracy based on the best question sets for two retrievers. † denotes statistical significance (p-value < 0.01) over Orig.

Question Set	EM for BM25	EM for DPR+BM25
Orig	42.84	48.57
Orig+FPQ	43.72†	49.15
Orig10Par+10FPQ	**44.40**†	**49.97**†

5 CONCLUSIONS

In this paper, we examine the importance of contiguous term spans, namely frozen phrases, that are expected to appear verbatim in answer passages. Frozen phrases embody locality that is crucial in finding potential answer passages in OpenQA. However, detecting them is challenging due to an absence of existing annotated data. We address this problem by introducing a strategy to construct synthetic silver data from existing QA datasets. We show that by incorporating frozen phrases, the retrieval accuracy as well as the end-to-end performance substantially improves. Frozen phrases can also be integrated into the backbone of dense retrieval models, which is an interesting direction for future work.

ACKNOWLEDGMENTS

This research is supported by the Natural Sciences and Engineering Research Council and by a grant from Huawei.

REFERENCES

[1] Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to Retrieve Reasoning Paths over Wikipedia Graph for Question Answering. In *ICLR*.

[2] 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)*.

Association for Computational Linguistics, Vancouver, Canada, 1870–1879. https://doi.org/10.18653/v1/P17-1171

[3] Christopher Clark and Matt Gardner. 2018. Simple and Effective Multi-Paragraph Reading Comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 845–855. https://doi.org/10.18653/v1/P18-1078

[4] Charles LA Clarke, Gordon V Cormack, and Thomas R Lynam. 2001. Exploiting redundancy in question answering. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. 358–365.

[5] Vincent Claveau. 2020. Query expansion with artificially generated texts. *arXiv preprint arXiv:2012.08787* (2020).

[6] W Bruce Croft, Howard R Turtle, and David D Lewis. 1991. The use of phrases and structured queries in information retrieval. In *Proc. of the SIGIR Conference*. 32–45.

[7] Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to Paraphrase for Question Answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 875–886. https://doi.org/10.18653/v1/D17-1091

[8] Pablo Duboue and Jennifer Chu-Carroll. 2006. Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. 33–36.

[9] Venkat N Gudivada, Vijay V Raghavan, William I Grosky, and Rajesh Kasanagottu. 1997. Information retrieval on the world wide web. *IEEE Internet Computing* 1, 5 (1997), 58–68.

[10] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*. PMLR, 3929–3938.

[11] Khaled M Hammouda and Mohamed S Kamel. 2004. Efficient phrase-based document indexing for web document clustering. *IEEE Transactions on knowledge and data engineering* 16, 10 (2004), 1279–1296.

[12] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Towards Unsupervised Dense Information Retrieval with Contrastive Learning. *arXiv preprint arXiv:2112.09118* (2021).

[13] 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*. Association for Computational Linguistics, Online, 874–880. https://doi.org/10.18653/v1/2021.eacl-main.74

[14] Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. 84–90.

[15] 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)*. Association for Computational Linguistics, Online, 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[16] Omar Khattab, Christopher Potts, and Matei Zaharia. 2021. Relevance-guided Supervision for OpenQA with ColBERT. *Transactions of the Association for Computational Linguistics* 9 (2021), 929–944. https://doi.org/10.1162/tacl_a_00405

[17] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* 7 (2019), 453–466.

[18] Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking Paragraphs for Improving Answer Recall in Open-Domain Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 565–569. https://doi.org/10.18653/v1/D18-1053

[19] 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*. Association for Computational Linguistics, Florence, Italy, 6086–6096. https://doi.org/10.18653/v1/P19-1612

[20] David D Lewis. 1992. An evaluation of phrasal and clustered representations on a text categorization task. In *Proc. of the SIGIR Conference*. 37–50.

[21] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, Vol. 33. 9459–9474.

[22] 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 *SIGIR*.

[23] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. 2021. Pretrained Transformers for Text Ranking: BERT and Beyond. *Synthesis Lectures on Human Language Technologies* 14, 4 (2021), 1–325. https://doi.org/10.2200/S01123ED1V01Y202108HLT053

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[25] Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021. Generation-Augmented Retrieval for Open-Domain Question Answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 4089–4100. https://doi.org/10.18653/v1/2021.acl-long.316

[26] Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Multi-hop Reading Comprehension through Question Decomposition and Rescoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 6097–6109. https://doi.org/10.18653/v1/P19-1613

[27] Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. Revealing the Importance of Semantic Retrieval for Machine Reading at Scale. 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)*. Association for Computational Linguistics, Hong Kong, China, 2553–2566. https://doi.org/10.18653/v1/D19-1258

[28] 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*. Association for Computational Linguistics, Copenhagen, Denmark, 574–583. https://doi.org/10.18653/v1/D17-1061

[29] Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document Ranking with a Pretrained Sequence-to-Sequence Model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Online, 708–718. https://doi.org/10.18653/v1/2020.findings-emnlp.63

[30] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375* (2019).

[31] Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised Question Decomposition for Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 8864–8880. https://doi.org/10.18653/v1/2020.emnlp-main.713

[32] Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering Complex Open-domain Questions Through Iterative Query Generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 2590–2602. https://doi.org/10.18653/v1/D19-1261

[33] Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, 5835–5847. https://doi.org/10.18653/v1/2021.naacl-main.466

[34] 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 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[35] Joseph Rocchio. 1971. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing* (1971), 313–323.

[36] Temple F Smith, Michael S Waterman, et al. 1981. Identification of common molecular subsequences. *Journal of molecular biology* 147, 1 (1981), 195–197.

[37] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[39] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R^3: Reinforced ranker-reader for open-domain question answering. In *AAAI*.

[40] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 5878–5882. https://doi.org/10.18653/v1/D19-1599

[41] Hugh E Williams, Justin Zobel, and Dirk Bahle. 2004. Fast phrase querying with combined indexes. *ACM transactions on information systems (TOIS)* 22, 4 (2004), 573–594.

[42] Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break It Down: A Question Understanding Benchmark. *Transactions of the Association for Computational Linguistics* 8 (2020), 183–198. https://doi.org/10.1162/tacl_a_00309

[43] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. In *ICLR*.

[44] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-End Open-Domain Question Answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Association for Computational Linguistics, Minneapolis, Minnesota, 72–77. https://doi.org/10.18653/v1/N19-4013

[45] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 2369–2380. https://doi.org/10.18653/v1/D18-1259

[46] Xuchen Yao, Benjamin Van Durme, and Peter Clark. 2013. Automatic coupling of answer extraction and information retrieval. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 159–165.

[47] Zhi Zheng, Kai Hui, Ben He, Xianpei Han, Le Sun, and Andrew Yates. 2020. BERT-QE: contextualized query expansion for document re-ranking. *arXiv preprint arXiv:2009.07258* (2020).