# Phrase Retrieval Learns Passage Retrieval, Too

**Anonymous authors**

## Abstract

Dense retrieval methods have shown great promise over sparse retrieval methods in a wide range of NLP problems. Among them, dense phrase retrieval[1]—the most fine-grained retrieval unit—is appealing because phrases can be directly used as the output for question answering and slot filling tasks. In this work, we follow the intuition that retrieving phrases naturally entails retrieving larger text blocks and study whether phrase retrieval can serve as the basis for coarse-level retrieval including passages and documents. We first observe that a dense phrase-retrieval system, without any retraining, already achieves better passage retrieval accuracy (+3-5% in top-5 accuracy on QA tasks) compared to passage retrievers, which also helps achieve superior end-to-end QA accuracy with fewer passages. Then, we provide an interpretation for why phrase-level supervision can help learn better fine-grained entailment compared to passage-level supervision, and also show that phrase retrieval can be improved to achieve competitive performance in document-retrieval tasks for entity linking and knowledge-grounded dialogue. Finally, we demonstrate how phrase filtering and vector quantization can reduce the size of our index by 4-10x, making dense phrase retrieval a practical and versatile solution in multi-granularity retrieval.

## 1. Introduction

Dense retrieval aims to retrieve relevant contexts from a large corpus, by learning dense representations of queries and text segments. Recently, dense retrieval of passages [Lee et al., 2019, Karpukhin et al., 2020] has been shown to outperform traditional sparse retrieval methods such as TF-IDF and BM25 in a wide range of knowledge-intensive NLP tasks [Petroni et al., 2021], including open-domain question answering (QA) [Chen et al., 2017], entity linking [Wu et al., 2020], and knowledge-grounded dialogue [Dinan et al., 2019].

One natural design choice of these dense retrieval methods is the retrieval unit. For instance, passage retrievers [Karpukhin et al., 2020] encode a fixed-size text block of 100 words as the basic retrieval unit. On the other extreme, recent work [Seo et al., 2019, Lee et al., 2021] demonstrates that phrases can be used as a retrieval unit. In particular,

1. Following previous work [Seo et al., 2018, 2019], we use the term 'phrase' to denote any contiguous text segment up to $L$ words (including single words), which is not necessarily a linguistic phrase.

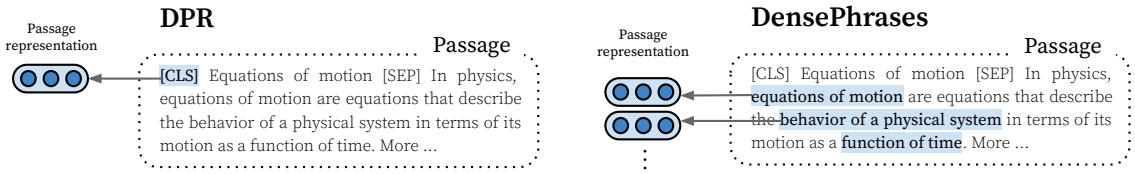


Figure 1: Comparison of passage representations from DPR [Karpukhin et al., 2020] and DensePhrases [Lee et al., 2021]. Unlike using a single vector for each passage, DensePhrases represents each passage with multiple phrase vectors and the score of a passage can be computed by the maximum score of phrases within it.

Lee et al. [2021] show that learning dense representations of phrases alone can achieve competitive performance in a range of open-domain QA and slot filling tasks. This is particularly appealing since the phrases can directly serve as the output, without relying on an additional reader model to process text passages.

In this work, we draw on an intuitive motivation that every single phrase is embedded within a larger text context and ask the following question: If a retriever is able to locate phrases, can we directly make use of it for passage retrieval as well? We formulate *phrase-based passage retrieval*, in which the score of a passage is determined by the maximum score of phrases within it (see Figure 1 for an illustration). By evaluating DensePhrases [Lee et al., 2021] on popular QA datasets, we observe that it achieves competitive or even better passage retrieval accuracy compared to the dense passage retriever (DPR) [Karpukhin et al., 2020], without any re-training or modification to the original model (Table 1). The gains are especially pronounced for top-$k$ accuracy when $k$ is smaller (e.g., 5), which also helps achieve superior open-domain QA accuracy with a much smaller number of passages as input to a generative reader model [Izacard and Grave, 2021a].

To better understand the nature of dense retrieval methods, we carefully analyze the training objectives of phrase and passage retrieval methods. While the in-batch negative losses in both models encourage them to retrieve topically relevant passages, we find that phrase-level supervision in DensePhrases provides a stronger training signal than using hard negatives from BM25, and helps DensePhrases retrieve correct phrases, and hence passages. Following this positive finding, we further explore whether phrase retrieval can be extended to retrieval of coarser granularities, or other NLP tasks. Through fine-tuning of the query encoder with document-level supervision, we are able to obtain competitive performance on entity linking [Hoffart et al., 2011] and knowledge-grounded dialogue retrieval [Dinan et al., 2019] in the KILT benchmark [Petroni et al., 2021].

Finally, we draw connections to multi-vector passage encoding models [Khattab and Zaharia, 2020, Luan et al., 2021], where phrase retrieval models can be viewed as learning a dynamic set of vectors for each passage. We show that a simple phrase filtering strategy learned from QA datasets gives us a control over the trade-off between the number of vectors per passage and the retrieval accuracy. Since phrase retrievers encode a larger number of vectors, we also introduce using Optimized Product Quantization [Ge et al., 2013] and quantization-aware fine-tuning, reducing the size of the phrase index from 307GB to 69GB (or even $<$ 30GB with more aggressive filtering) for full English Wikipedia, without any performance degradation. This matches the index size of passage retrieval and makes dense phrase retrieval a practical and versatile solution for multi-granularity retrieval.

## 2. Background

**Passage retrieval.** Given a set of documents $\mathcal{D}$, passage retrieval aims to provide a set of relevant passages for a question $q$. Typically, each document in $\mathcal{D}$ is split into a set of passages and we denote the entire set of passages in $\mathcal{D}$ as $\mathcal{P} = \{p_1, \ldots, p_M\}$, where each passage can be a natural paragraph or a fixed-length text block. A passage retriever is designed to return top-$k$ passages $\mathcal{P}_k \subset \mathcal{P}$ with the goal of retrieving passages that are relevant to the question. In open-domain QA, passages are considered relevant if they contain answers to the question. However, many other knowledge-intensive NLP tasks (e.g., knowledge-grounded dialogue) provide human-annotated evidence passages or documents.

While traditional passage retrieval models rely on sparse representations such as BM25 [Robertson and Zaragoza, 2009], recent methods show promising results with dense representations of passages and questions, and enable retrieving passages that may have low lexical overlap with questions. Specifically, Karpukhin et al. [2020] introduce a dense passage retriever (DPR) that has a *passage encoder* $E_p(\cdot)$ and a *question encoder* $E_q(\cdot)$ trained on QA datasets and retrieves passages by using the inner product as a similarity function:

$$f(p, q) = E_p(p)^\top E_q(q). \tag{1}$$

For open-domain QA where a system is required to provide an exact answer string $a$, the retrieved top-$k$ passages $\mathcal{P}_k$ are subsequently fed into a reading comprehension model such as a BERT-QA model [Devlin et al., 2019], and this is called the retriever-reader approach.

**Phrase retrieval.** While passage retrievers require another reader model to find an answer, Seo et al. [2019] introduce the phrase retrieval approach that encodes phrases in each document and performs similarity search over all phrase vectors to directly locate the answer. Given a phrase $s^{(p)}$ from a passage $p$, their similarity function $f$ is computed as:

$$f(s^{(p)}, q) = E_s(s^{(p)})^\top E_q(q), \tag{2}$$

where $E_s(\cdot)$ and $E_q(\cdot)$ denote the *phrase encoder* and the *question encoder*, respectively. Since this formulates open-domain QA purely as a maximum inner product search (MIPS), it can drastically improve end-to-end efficiency. While previous approaches [Seo et al., 2019, Lee et al., 2020] relied on the combination of dense and sparse phrase vectors, Lee et al. [2021] demonstrate that dense representations of phrases alone are sufficient to close the performance gap with retriever-reader systems.

## 3. Phrase Retrieval Learns Passage Retrieval

Phrases naturally have their source texts from which they are extracted. Based on this fact, we define a very simple phrase-based passage retrieval strategy, where we retrieve passages based on the phrase-retrieval score:

$$\tilde{f}(p, q) := \max_{s^{(p)} \in \mathcal{S}(p)} E_s(s^{(p)})^\top E_q(q), \tag{3}$$

where $\mathcal{S}(p)$ denotes the set of phrases in the passage $p$. In practice, we first retrieve top-$k_2$ phrases, compute the score for each passage, and return top-$k$ unique passages.[2] Based

2. In most cases, $k_2 = 2k$ is sufficient. If not, we re-try $2k_2$.

| Retriever | Natural Questions | | | | | TriviaQA | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Top-1 | Top-5 | Top-20 | MRR@20 | P@20 | Top-1 | Top-5 | Top-20 | MRR@20 | P@20 |
| DPR | 46.0 | 68.1 | **79.8** | 55.7 | 16.5 | 54.4[†] | - | 79.4[‡] | - | - |
| DPR-multi | 44.2 | 66.8 | 79.2 | 54.2 | 17.7 | 54.6 | 70.8 | 79.5 | 61.7 | 30.3 |
| DensePhrases-NQ | 50.1 | 69.5 | **79.8** | 58.7 | 20.5 | - | - | - | - | - |
| DensePhrases-multi | **51.1** | **69.9** | 78.7 | **59.3** | **22.7** | **62.7** | **75.0** | **80.9** | **68.2** | **38.4** |

Table 1: Open-domain QA passage retrieval results. DensePhrases retrieves passages following Eq. (3). We report top-$k$ passage retrieval accuracy (Top-$k$), mean reciprocal rank at $k$ (MRR@$k$), and precision at $k$ (P@$k$) in percent. See §3.1 for the details on each model. [†]: [Yang and Seo, 2020]. [‡]: [Karpukhin et al., 2020].

on our definition, phrases can act as a basic retrieval unit of any other granularity such as sentences or documents by simply changing $\mathcal{S}(p)$ (e.g., $s^{(d)} \in \mathcal{S}(d)$ for a document $d$). In this section, we evaluate the passage retrieval performance of Eq. (3) and also evaluate how phrase-based passage retrieval can contribute to open-domain QA systems.

### 3.1 Experiment: Passage Retrieval

**Datasets.** We use two open-domain QA datasets, Natural Questions [Kwiatkowski et al., 2019] and TriviaQA [Joshi et al., 2017], each of which follows the standard train/dev/test splits for the open-domain QA evaluation. For both models, we use the 2018-12-20 Wikipedia snapshot. To provide a fair comparison, we use Wikipedia articles pre-processed for DPR, which are split into 21M text blocks and each text block has exactly 100 words.

**Models.** For DPR, we use publicly available checkpoints[3] trained on each dataset (DPR) or multiple QA datasets (DPR-multi)[4], which we find to perform slightly better than the ones reported in Karpukhin et al. [2020]. For DensePhrases, we train the models with the code provided by the authors[5] on Natural Questions (DensePhrases-NQ) or multiple QA datasets (DensePhrases-multi). Note that we do not make any modification to the architecture or training methods of DensePhrases and achieve similar open-domain QA accuracy as reported. For phrase-based passage retrieval, we compute Eq. (3) with DensePhrases and return top-$k$ passages.

**Metrics.** Following previous work on passage retrieval for open-domain QA, we measure the top-$k$ passage retrieval accuracy (Top-$k$), which denotes the proportion of questions whose top-$k$ retrieved passages contain at least one of the gold answers. To further characterize the behavior of each system, we also include the following evaluation metrics: mean reciprocal rank at $k$ (MRR@$k$) and precision at $k$ (P@$k$). Given the top-$k$ passages $\mathcal{P}_k$ for a question, MRR@$k$ is the average reciprocal rank of the first relevant passage (that contains an answer) in the top-$k$ passages. Higher MRR@$k$ means relevant passages appear at higher ranks. Meanwhile, P@$k$ is the average proportion of relevant passages in the top-$k$ passages. Higher P@$k$ denotes that a larger proportion of top-$k$ passages contains the answers.

3. https://github.com/facebookresearch/DPR
4. DPR-multi is trained on NaturalQuestions, TriviaQA, CuratedTrec [Baudiš and Šedivỳ, 2015], and WebQuestions [Berant et al., 2013]. DensePhrases-multi additionally includes SQuAD [Rajpurkar et al., 2016], although it does not contribute to Natural Questions and TriviaQA much.
5. https://github.com/princeton-nlp/DensePhrases

| Retriever | Reader | NQ Dev | NQ Test | TQA Test | Retriever | Reader | NQ Dev | NQ Test | TQA Test |
|---|---|---|---|---|---|---|---|---|---|
| Results from Izacard and Grave [2021a] **Training resource**: 64 Tesla V100 32GB | | | | | Results with DensePhrases [Lee et al., 2021] **Training resource**: 4 RTX 24GB | | | | |
| DPR-multi | T5-base ($k = 5$) | 37.8 | - | - | DensePhrases-multi | None | - | 41.3 | 53.5 |
| | T5-base ($k = 10$) | 42.3 | - | - | | T5-base ($k = 5$) | 44.2 | 45.9 | 59.5 |
| | T5-base ($k = 25$) | 45.3 | - | - | | T5-base ($k = 10$) | 45.5 | 45.9 | 61.0 |
| | T5-base ($k = 50$) | 45.7 | - | - | | T5-base ($k = 25$) | 46.4 | 47.2 | 63.4 |
| | T5-base ($k = 100$) | 46.5 | **48.2** | **65.0** | | T5-base ($k = 50$) | **47.2** | 47.9 | 64.5 |

Table 2: Open-domain QA results. We report exact match (EM) of each model by feeding top-$k$ passages into a T5-base model. DensePhrases can reduce the computational cost of running generative reader models while having competitive performance.

**Results.** As shown in Table 1, DensePhrases achieves competitive passage retrieval accuracy with DPR, while having a clear advantage on top-1 or top-5 accuracy for both Natural Questions (+6.7% Top-1) and TriviaQA (+7.5% Top-1). Although the top-20 (and top-100, which is not shown) accuracy is similar across different models, MRR@20 and P@20 reveal interesting aspects of DensePhrases—it ranks relevant passages higher and provides a larger number of correct passages. Our results suggest that DensePhrases can also retrieve passages very accurately, although it was not explicitly trained for that purpose. For the rest of the paper, we mainly compare DPR-multi and DensePhrases-multi.

### 3.2 Experiment: Open-domain QA

Recently, Izacard and Grave [2021a] proposed the Fusion-in-Decoder (FiD) approach where top-100 passages from DPR are fed into a generative model such as T5 [Raffel et al., 2020]. Since their generative model computes the hidden states of all tokens in 100 passages, this takes vast computational resources such as 64 Tesla V100 32GB GPUs. We use our phrase-based passage retrieval with DensePhrases-multi to replace DPR-multi in FiD, and see if we can use a much smaller number of passages to achieve comparable performance, which can greatly reduce the computational requirements. Since we train our model with 4 24GB RTX GPUs, we use up to 50 passages for training T5-base. Note that training T5-base with 5 or 10 passages can also be done with 11GB GPUs. We keep all the hyperparameters the same except the batch size (reduced from 64 to 4) and accumulate gradients for 16 steps to match the batch size of the original work.

**Results.** As shown in Table 2, using DensePhrases as a passage retriever achieves competitive performance with FiD (DPR-multi + T5-base) and significantly improves upon the performance of original DensePhrases (without a reader). Its better retrieval quality at top-$k$ for smaller $k$ indeed translates to better open-domain QA accuracy, achieving +6.4% gain compared to DPR + T5-base when $k = 5$. To obtain similar performance with using 100 passages in FiD, DensePhrases needs fewer passages, which can fit in smaller GPUs.

## 4. A Unified View of Phrase and Passage Retrieval

As shown in the previous section, phrase-based passage retrieval can achieve competitive passage retrieval accuracy, despite the fact that phrase retrieval was not explicitly trained

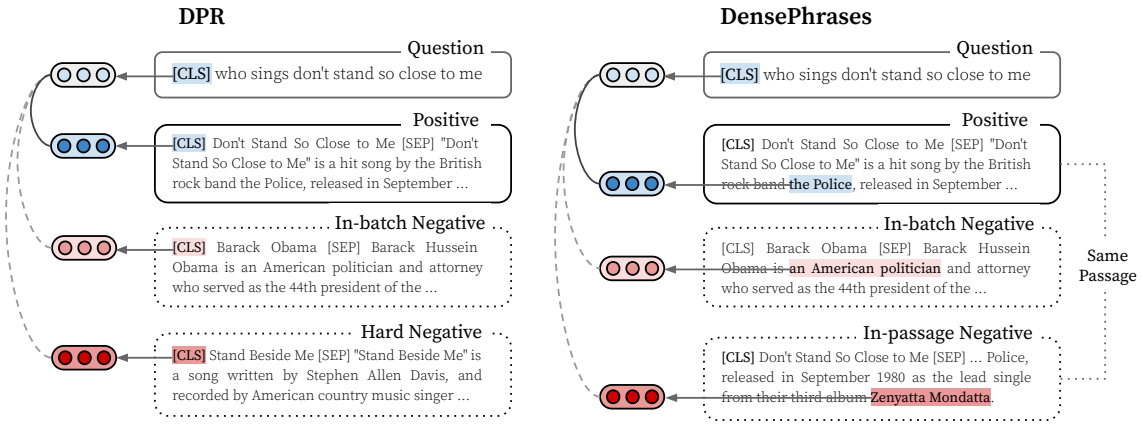


Figure 2: Comparison of training objectives of DPR and DensePhrases. While both models use in-batch negatives, DensePhrases use in-passage negatives (phrases) compared to BM25 hard-negative passages in DPR. Note that each phrase in DensePhrases can directly serve as an answer to open-domain questions.

for that. In this section, we compare the training objectives of DPR and DensePhrases in detail and explain how DensePhrases learns passage retrieval.

### 4.1 Training Objectives

Both DPR and DensePhrases set out to learn a similarity function $f$ between a passage or phrase and a question. Passages and phrases are conceptually not so different and vary primarily in characteristic length, so we refer to either as a retrieval unit $x$.[6] DPR and DensePhrases both adopt a dual-encoder approach with inner product similarity (Eq. (1) and (2)), using BERT [Devlin et al., 2019] and SpanBERT [Joshi et al., 2020], respectively.

These dual-encoder models are then trained with a negative log-likelihood loss for discriminating positive retrieval units from negative ones:

$$\mathcal{L} = -\log \frac{e^{f(x^+,q)}}{e^{f(x^+,q)} + \sum_{x^- \in \mathcal{X}^-} e^{f(x^-,q)}}, \tag{4}$$

where $x^+$ is the positive phrase or passage corresponding to question $q$, and $\mathcal{X}^-$ is a set of negative examples. The choice of negatives is critical in this setting and both DPR and DensePhrases make important adjustments.

**In-batch negatives.** In-batch negatives are a common way to define $\mathcal{X}^-$, since they are available at no extra cost when encoding a mini-batch of examples. Specifically, in a mini-batch of $B$ examples, we can add $B-1$ in-batch negatives for each positive example. Since each mini-batch is randomly sampled from the set of all training passages, in-batch negative passages are usually *topically negative*, i.e., models can discriminate between $x^+$ and $\mathcal{X}^-$ based on their topic only.

6. Another detail is that phrases may overlap, whereas passages are usually segmented.

**Hard negatives.** Although topic-related features are useful in identifying broadly relevant passages, they often lack the precision to locate the exact passage containing the answer in a large corpus. Karpukhin et al. [2020] propose to use additional hard negatives which have a high BM25 lexical overlap with a given question but do not contain the answer. These hard negatives are likely to share a similar topic and encourage DPR to learn more fine-grained features to rank $x^+$ over the hard negatives. Figure 2 (left) shows an illustrating example.

**In-passage negatives.** While DPR is limited to use positive passages $x^+$ which merely contain the answer, DensePhrases is trained to predict that the positive phrase $x^+$ *is* the answer. Thus, the fine-grained structure of phrases allows for another source of negatives, *in-passage negatives*. In particular, DensePhrases augments the set of negatives $\mathcal{X}^-$ to encompass all phrases within the same passage that do not express the answer. See Figure 2 (right) for an example. We hypothesize that these in-passage negatives achieve a similar effect as DPR's hard negatives: They require the model to go beyond simple topic modeling, since they share not only the same topic but also the same context. Thus, phrase-based passage retriever might benefit from this phrase-level supervision, which has already been shown to be useful in the context of distilling knowledge from reader to retriever [Izacard and Grave, 2021b, Yang and Seo, 2020].

### 4.2 Topical vs. Hard Negatives

To address our hypothesis, we would like to study how the different kinds of negatives used by DPR and DensePhrases affect their reliance on topical and fine-grained entailment cues. We characterize their passage retrieval based on two metrics (losses), $\mathcal{L}_{\text{topic}}$ and $\mathcal{L}_{\text{hard}}$. We use Eq. (4) to define both $\mathcal{L}_{\text{topic}}$ and $\mathcal{L}_{\text{hard}}$, but use different sets of negatives $\mathcal{X}^-$. For $\mathcal{L}_{\text{topic}}$, $\mathcal{X}^-$ contains passages that are topically different from the gold passage; in practice, a single random Wikipedia passage. For $\mathcal{L}_{\text{hard}}$, $\mathcal{X}^-$ uses negatives containing topically similar passages, such that $\mathcal{L}_{\text{hard}}$ estimates how accurately models locate a passage that contains the exact answer among topically similar passages. We create a single hard negative by removing the sentence that contains the answer from the positive passage.

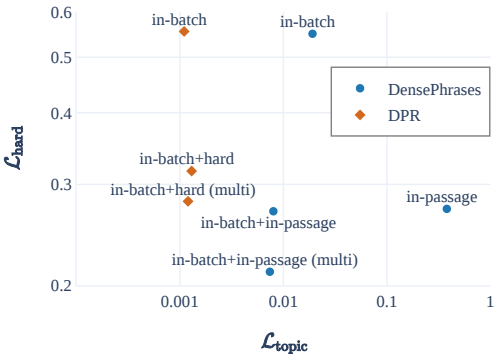


Figure 3: Comparison of DPR and DensePhrases on NQ (dev) with two losses: $\mathcal{L}_{\text{topic}}$ and $\mathcal{L}_{\text{hard}}$. For both losses, the lower the better. We show the effect of training with in-batch negatives (in-batch), hard negatives (hard), and in-passage negatives (in-passage), as well as training on multiple QA datasets (multi).

**Results.** Figure 3 shows the comparison of DPR and DensePhrases trained on NQ or multiple datasets with the two losses. For DensePhrases, we compute the passage score using $\tilde{f}(p, q)$ as described in Eq. (3). We observe that in-batch negatives are highly effective at reducing $\mathcal{L}_{\text{topic}}$, while DensePhrases trained without in-batch negatives has a relatively high $\mathcal{L}_{\text{topic}}$. Furthermore, we observe that DensePhrases with in-passage negatives achieves an even lower $\mathcal{L}_{\text{hard}}$ than DPR, despite not using any hard negatives. Using multiple

datasets (denoted as 'multi') further improves $\mathcal{L}_{\text{hard}}$ for both models. The results suggest that DensePhrases relies less on topical features and is better at retrieving passages based on local entailment cues.

**Hard negatives for DensePhrases?** We test two different kinds of hard negatives in DensePhrases to see whether its performance can further improve in the presence of in-passage negatives. For each question, we mine for a hard negative passage, either by BM25 similarity or by finding another passage that contains the gold-answer phrase, but possibly with a wrong context. Then we use all phrases from the hard negative passage as additional hard negatives in $\mathcal{X}^-$. As shown in Appendix A (Table 5), DensePhrases obtains no substantial improvements from additional hard negatives, indicating that in-passage negatives are already highly effective at producing good phrase (or passage) representations.

## 5. Improving Coarse-grained Retrieval

While we showed that DensePhrases implicitly learns passage retrieval, Figure 3 indicates that DensePhrases might not be very good for retrieval tasks where topic matters more than fine-grained entailment, for instance, the retrieval of a single evidence document for entity linking. In this section, we propose a simple method that can adapt DensePhrases to larger retrieval units, especially when the topical relevance is more important.

**Method.** We modify the query-side fine-tuning proposed by Lee et al. [2021], which drastically improves the performance of DensePhrases by reducing the discrepancy between training and inference time. Since it is prohibitive to update the large number of phrase representations after indexing, only the query encoder is fine-tuned over the entire set of phrases in Wikipedia. Given a question $q$ and an annotated document set $\mathcal{D}^*$, we minimize:

$$\mathcal{L}_{\text{doc}} = -\log \frac{\sum_{s \in \tilde{\mathcal{S}}(q), d(s) \in \mathcal{D}^*} e^{f(s,q)}}{\sum_{s \in \tilde{\mathcal{S}}(q)} e^{f(s,q)}}, \tag{5}$$

where $\tilde{\mathcal{S}}(q)$ denotes top-$k$ phrases for the question, out of the entire set of phrase vectors. To retrieve coarse-grained text better, we simply check the condition $d(s) \in \mathcal{D}^*$, which means $d(s)$, the source document of $s$, is included in the set of annotated gold documents $\mathcal{D}^*$ for the question. With $\mathcal{L}_{\text{doc}}$, the model is trained to retrieve any phrases that are contained in a relevant document. Note that $d(s)$ can be changed to reflect any desired level of granularity.

**Experimental setup.** We train DensePhrases with $\mathcal{L}_{\text{doc}}$ for entity linking [Hoffart et al., 2011] or knowledge-grounded dialogue [Dinan et al., 2019] tasks from KILT [Petroni et al., 2021], where a gold document for each question is given. On these tasks, each model is expected to provide a small set of articles related to the query. We include DensePhrases trained with the original query-side fine-tuning loss (denoted as $\mathcal{L}_{\text{phrase}}$) as a baseline, which labels any phrase that matches the answer string (here, titles of gold documents are used as answer strings) as positive. For retrieval evaluation, we adopt KILT's official document-level metrics based on the gold annotated documents: R-precision and Recall@5. See the caption of Table 3 for the details on the metrics.

**Results.** Table 3 shows the results on three entity linking tasks and a knowledge-grounded dialogue task with DensePhrases trained with $\mathcal{L}_{\text{phrase}}$ or $\mathcal{L}_{\text{doc}}$. We also include results of

| Model | Entity Linking AY2 | Entity Linking WnWi | Entity Linking WnCw | Dialogue WoW | Model | Entity Linking AY2 | Entity Linking WnWi | Entity Linking WnCw | Dialogue WoW |
|---|---|---|---|---|---|---|---|---|---|
| | *R-precision* | | | | | *Recall@5* | | | |
| RAG | **72.6** | **48.1** | **47.6** | **57.8** | RAG | 72.6 | 48.1 | 47.6 | **74.6** |
| DensePhrases w/ $\mathcal{L}_{\text{phrase}}$ | 7.7 | 12.5 | 6.4 | - | DensePhrases w/ $\mathcal{L}_{\text{phrase}}$ | 20.0 | 30.5 | 17.5 | - |
| DensePhrases w/ $\mathcal{L}_{\text{doc}}$ | 61.6 | 32.1 | 37.4 | 47.0 | DensePhrases w/ $\mathcal{L}_{\text{doc}}$ | **76.2** | **52.4** | **54.6** | 70.8 |
| Multi-task DPR | 26.5 | 4.9 | 1.9 | 41.1 | Multi-task DPR | 39.5 | 12.0 | 3.6 | 67.2 |
| Multi-task DensePhrases | **68.4** | **47.5** | **47.5** | **55.7** | Multi-task DensePhrases | **84.3** | **68.4** | **65.4** | **75.6** |

Table 3: Results on the KILT test set. We report page-level R-precision and Recall@5 on each task. R-precision is the proportion of successfully retrieved pages in the top-R results, where R is the number of distinct pages in a provenance set. Recall@5 is defined as the proportion of distinct provenance sets that were retrieved in the top-5.

DensePhrases trained with $\mathcal{L}_{\text{doc}}$ on the KILT datasets in a multi-task setting, similar to the multi-task DPR baseline provided by Petroni et al. [2021]. Since $\mathcal{L}_{\text{phrase}}$ requires the answer string to be a short phrase, we only apply it on entity linking tasks where we use document titles as gold answers. On all tasks, we find that DensePhrases with $\mathcal{L}_{\text{doc}}$ performs comparably well with DPR-based models. Using $\mathcal{L}_{\text{phrase}}$ (i.e., original DensePhrases) does very poorly since it focuses on phrase-level entailment, rather than document-level relevance.

## 6. DensePhrases as a Multi-Vector Passage Encoder

In this section, we demonstrate that DensePhrases can be interpreted as a multi-vector passage encoder, which has recently been shown to be very effective for passage retrieval [Luan et al., 2021, Khattab and Zaharia, 2020]. Since this type of multi-vector encoding models requires a large disk footprint, we show that we can control the number of vectors per passage (and hence the index size) through filtering. We also introduce quantization techniques to build more efficient phrase retrieval models without a significant performance drop.

**Multi-vector encodings.** Since we represent passages not by a single vector, but by a set of phrase vectors (decomposed as token-level start and end vectors, see Lee et al. [2021]), we notice similarities to previous work, which addresses the capacity limitations of dense, fixed-length passage encodings. While these approaches store a fixed number of vectors per passage [Luan et al., 2021, Humeau et al., 2020] or all token-level vectors [Khattab and Zaharia, 2020], phrase retrieval models store a dynamic number of phrase vectors per passage, where many phrases are filtered by a model trained on QA datasets.

Specifically, Lee et al. [2021] train a binary classifier (or a phrase filter) to filter phrases based on their phrase representations. This phrase filter is supervised by the answer annotations in QA datasets, hence denotes candidate answer phrases. In our experiment, we tune the filter threshold to control the number of vectors per passage for passage retrieval.

**Optimized product quantization.** Since the multi-vector encoding models are prohibitively large due to their multiple representations, we further introduce a vector quantization based method that can safely reduce the size of our phrase index, without performance degradation. We use Product Quantization (PQ) [Jegou et al., 2010] where the original vector space is decomposed into the Cartesian product of subspaces. Using PQ, the memory usage of using $N$ number of $d$-dimensional centroid vectors reduces from $Nd$ to

| **Model** | $\tau$ | # vec / $p$ | Quant. | Size (GB) | Top-5 |
|---|---|---|---|---|---|
| DensePhrases | 1.0 | 28.0 | SQ4 | 307 | - |
| | 1.0 | 28.0 | OPQ96 | 69 | **69.0** |
| | 2.0 | 17.9 | OPQ96 | 49 | 68.5 |
| | 2.5 | 13.2 | OPQ96 | 40 | 68.1 |
| | 3.0 | 8.8 | OPQ96 | 30 | 67.0 |
| | 3.5 | 5.1 | OPQ96 | 23 | 64.8 |
| w/o QSFT | 1.0 | 28.0 | OPQ96 | 69 | 59.3 |
| DPR | - | 1 | None | 62 | 66.3 |

Table 4: Top-5 passage retrieval accuracy on Natural Questions (dev) for different index sizes (GB) of DensePhrases. The index size of DensePhrases is controlled by the filtering threshold $\tau$. # vec / $p$: average number of saved vectors per passage. Quant.: quantization method. QSFT: quantization-aware query-side fine-tuning.

$N^{1/M}d$ with $M$ subspaces while each database vector requires $\log_2 N$ bits. Among different variants of PQ, we use Optimized Product Quantization (OPQ) [Ge et al., 2013], which uses an additional orthogonal matrix $R$ to better decompose the original vector space. See Ge et al. [2013] for more details on OPQ.

**Quantization-aware training.** While this type of aggressive vector quantization can significantly reduce the memory usage, it often comes at the cost of performance degradation due to the quantization loss. To mitigate this problem, we use quantization-aware query-side fine-tuning motivated by the recent successes on quantization-aware training [Jacob et al., 2018]. Specifically, during query-side fine-tuning, we reconstruct the phrase vectors using the trained (optimized) product quantizer, which are then used to minimize Eq. (5).

**Results.** In Table 4, we present the top-5 passage retrieval accuracy with respect to the size of the phrase index in DensePhrases. By tuning the threshold $\tau$ for the phrase filter, the number of vectors per each passage can be reduced without hurting the performance significantly. The performance improves with a larger number of vectors per passage, which aligns with the findings of multi-vector encoding models [Khattab and Zaharia, 2020, Luan et al., 2021]. Our results show that having 8.8 number of vectors per passage in DensePhrases has similar accuracy with DPR. While applying OPQ can reduce the index size of DensePhrases from 307GB to 69GB, it should be done with the quantization-aware query-side fine-tuning.

## 7. Conclusion

In this paper, we show that phrase retrieval models also learn passage retrieval without any modification. By drawing connections between the objectives of DPR and DensePhrases, we provide a better understanding of how phrase retrieval learns passage retrieval, which is also supported by several empirical evaluations on multiple benchmarks. Specifically, phrase-based passage retrieval has better retrieval quality on top-$k$ passages when $k$ is small, and this translates to an efficient use of passages for open-domain QA. We also show that DensePhrases can be fine-tuned for more coarse-grained retrieval units, serving as a basis for any retrieval unit. We plan to further evaluate phrase-based passage retrieval on standard information retrieval tasks such as MS MARCO.

## References

Petr Baudiš and Jan Šedivỳ. Modeling of the question answering task in the YodaQA system. In *International Conference of the Cross-Language Evaluation Forum for European Languages (CLEF)*, 2015.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. In *Association for Computational Linguistics (ACL)*, pages 1870–1879, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186, 2019.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of Wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations (ICLR)*, 2019.

Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization. *IEEE transactions on pattern analysis and machine intelligence*, 36(4):744–755, 2013.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust disambiguation of named entities in text. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 782–792, 2011.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *International Conference on Learning Representations*, 2020.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *European Chapter of the Association for Computational Linguistics (EACL)*, 2021a.

Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering. In *International Conference on Learning Representations*, 2021b.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2704–2713, 2018.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1): 117–128, 2010.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Association for Computational Linguistics (ACL)*, pages 1601–1611, 2017.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association of Computational Linguistics (TACL)*, 8:64–77, 2020.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48, 2020.

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

Jinhyuk Lee, Minjoon Seo, Hannaneh Hajishirzi, and Jaewoo Kang. Contextualized sparse representations for real-time open-domain question answering. In *Association for Computational Linguistics (ACL)*, pages 912–919, 2020.

Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. Learning dense representations of phrases at scale. In *Association for Computational Linguistics (ACL)*, 2021.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Association for Computational Linguistics (ACL)*, pages 6086–6096, 2019.

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

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, et al. Kilt: a benchmark for knowledge intensive language tasks. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392, 2016.

Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc, 2009.

Minjoon Seo, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. Phrase-indexed question answering: A new challenge for scalable document comprehension. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 559–564, 2018.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. Real-time open-domain question answering with dense-sparse phrase index. In *Association for Computational Linguistics (ACL)*, pages 4430–4441, 2019.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. Scalable zero-shot entity linking with dense entity retrieval. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 6397–6407, 2020.

Sohee Yang and Minjoon Seo. Is retriever merely an approximator of reader? *arXiv preprint arXiv:2010.10999*, 2020.

## Appendix A. Hard Negatives for DensePhrases

| Type | $\mathcal{D} = \{p\}$ | $\mathcal{D} = \mathcal{D}_{\text{small}}$ |
|---|---|---|
| DensePhrases | 71.8 | **61.3** |
| + BM25 neg. | 71.8 | 60.6 |
| + Same-phrase neg. | **72.1** | 60.9 |

Table 5: Effect of using hard negatives in DensePhrases on the NQ development set. We report EM when a single gold passage is given ($\mathcal{D} = \{p\}$) or 6K passages are given by gathering all the gold passages from NQ development set ($\mathcal{D} = \mathcal{D}_{\text{small}}$). The two hard negatives do not give any noticeable improvement in DensePhrases.